Mathematics of Data: From Theory to Computation

Prof. Volkan Cevher
volkan.cevher@epfl.ch

Lecture 8: From variance reduction to deep learning...

Laboratory for Information and Inference Systems (LIONS)
École Polytechnique Fédérale de Lausanne (EPFL)

EE-556 (Fall 2023)

License Information for Mathematics of Data Slides

- This work is released under a Creative Commons License with the following terms:
- **Attribution**
  - The licensor permits others to copy, distribute, display, and perform the work. In return, licensees must give the original authors credit.
- **Non-Commercial**
  - The licensor permits others to copy, distribute, display, and perform the work. In return, licensees may not use the work for commercial purposes – unless they get the licensor's permission.
- **Share Alike**
  - The licensor permits others to distribute derivative works only under a license identical to the one that governs the licensor's work.
- Full Text of the License

An observation of GD vs. SGD step

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \gamma_k \nabla f(\mathbf{x}^k) \quad \text{(GD)}$$

Lemma

Assume f is Lipschitz smooth with constant L. Then,

$$f(\mathbf{x}^{k+1}) - f(\mathbf{x}^k) \leq \left(\frac{\gamma_k^2 L}{2} - \gamma_k \right) \|\nabla f(\mathbf{x}^k)\|^2.$$

An observation of GD vs. SGD step

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \gamma_k G(\mathbf{x}^k, \theta_k) \quad \text{(SGD)}$$

Lemma

Assume f is Lipschitz smooth with constant L. Then,

$$\mathbb{E}[f(\mathbf{x}^{k+1}) - f(\mathbf{x}^k)] \leq \left(\frac{\gamma_k^2 L}{2} - \gamma_k\right)\mathbb{E}[\|\nabla f(\mathbf{x}^k)\|^2] + \frac{L\gamma_k^2}{2}\mathbb{E}[\|G(\mathbf{x}^k, \theta_k) - \nabla f(\mathbf{x}^k)\|^2]$$

An observation of GD vs. SGD step

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \gamma_k G(\mathbf{x}^k, \theta_k) \quad \text{(SGD)}$$

Lemma

Assume f is Lipschitz smooth with constant L. Then,

$$\mathbb{E}[f(\mathbf{x}^{k+1}) - f(\mathbf{x}^k)] \leq \left(\frac{\gamma_k^2 L}{2} - \gamma_k\right) \mathbb{E}[\|\nabla f(\mathbf{x}^k)\|^2] + \frac{L\gamma_k^2}{2} \mathbb{E}[\|G(\mathbf{x}^k, \theta_k) - \nabla f(\mathbf{x}^k)\|^2]$$

Observations:
- ◦ The variance of gradient estimate dominates as $\nabla f(\mathbf{x}^k) \to 0$.
- ◦ To ensure convergence we need to control variance.

$$\gamma_k \to 0 \Longrightarrow \text{Slow convergence!}$$

Can we decrease the variance while using a constant step-size?

An observation of GD vs. SGD step

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \gamma_k G(\mathbf{x}^k, \theta_k) \quad \text{(SGD)}$$

Lemma

Assume f is Lipschitz smooth with constant L. Then,

$$\mathbb{E}[f(\mathbf{x}^{k+1}) - f(\mathbf{x}^k)] \leq \left(\frac{\gamma_k^2 L}{2} - \gamma_k\right)\mathbb{E}[\|\nabla f(\mathbf{x}^k)\|^2] + \frac{L\gamma_k^2}{2}\mathbb{E}[\|G(\mathbf{x}^k, \theta_k) - \nabla f(\mathbf{x}^k)\|^2]$$

Observations:
- The variance of gradient estimate dominates as $\nabla f(\mathbf{x}^k) \to 0$.
- To ensure convergence we need to control variance.

$$\gamma_k \to 0 \Longrightarrow \text{Slow convergence!}$$

Can we decrease the variance while using a constant step-size?

Choose a stochastic gradient, s.t. $\mathbb{E}\left[\|G(\mathbf{x}^k; \theta_k)\|^2\right] \to 0.$

A simple approach: Mini-batch SGD

◦ More samples imply a better estimate for full gradient.

SGD with mini batches

Let $G(\mathbf{x}, \theta)$ be an unbiased gradient estimate $(\mathbb{E}[G(\mathbf{x}, \theta)] = \nabla f(\mathbf{x}))$ and B_k be the batch size. Then, we have

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \alpha_k \frac{1}{B_k} \sum_{j=1}^{B_k} G(\mathbf{x}^k, \theta_{k,j}).$$

Theorem

Let $B_k > 0$ be the batch size and $G(\mathbf{x}, \theta)$ be an unbiased gradient estimate with bounded variance, i.e., $\mathbb{E}[\| G(\mathbf{x}, \theta) - \nabla f(\mathbf{x}) \|^2 \mid \mathbf{x}] \leq \sigma^2$. Then, the mini-batch estimate has the following properties:

$$\mathbb{E}\left[\frac{1}{B_k} \sum_{j=1}^{B_k} G(\mathbf{x}, \theta_{k,j})\right] = \nabla f(\mathbf{x}) \qquad \text{and} \qquad \mathbb{E}\left[\left\| \frac{1}{B_k} \sum_{j=1}^{B_k} G(\mathbf{x}, \theta_{k,j}) - \nabla f(\mathbf{x}) \right\|^2 \mid \mathbf{x}\right] \leq \frac{\sigma^2}{B_k}.$$

Remarks:

◦ We might need to increase the batch size over time to take variance to 0.

◦ We can come up with a “smarter” estimate for $\nabla f(\mathbf{x})$.

How to construct a new estimate $G(\mathbf{x}^k; \theta_k)$? [6]

Finite sum structure:	SGD update rule:
$f^\star := \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) := \frac{1}{n}\sum_{j=1}^n f_j(\mathbf{x}) \right\}$	$\mathbf{x}^{k+1} = \mathbf{x}^k - \gamma_k \nabla f_j(\mathbf{x}^k)$

- Let $X = \nabla f_j(\mathbf{x}^k)$ be a random variable (due to $j \sim \text{Uniform}(\{1, \cdots, n\})$).
- Let $Y = \nabla f_j(\tilde{\mathbf{x}})$ be another random variable, and $\tilde{\mathbf{x}}$ is a particularly selected point.

Remarks:
- We want X and Y to be correlated (we will see why!).
- Given Y, we should be able to estimate $\mathbb{E}[X]$ with more confidence.

Observations:
- Choice of $\tilde{\mathbf{x}}$ affects how correlated X and Y are.
- We can compute $\mathbb{E}[Y] = \frac{1}{n}\sum_{j=1}^n \nabla f_j(\tilde{\mathbf{x}}) = \nabla f(\tilde{\mathbf{x}})$.

Goal:
- Find a **good** estimate of $\mathbb{E}[X] = \frac{1}{n}\sum_{j=1}^n \nabla f_j(\mathbf{x}^k) = \nabla f(\mathbf{x}^k)$.

How to construct a new estimate $G(\mathbf{x}^k; \theta_k)$? [6]

Finite sum structure:	SGD update rule:
$f^\star := \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) := \frac{1}{n}\sum_{j=1}^n f_j(\mathbf{x}) \right\}$	$\mathbf{x}^{k+1} = \mathbf{x}^k - \gamma_k \nabla f_j(\mathbf{x}^k)$

◦ Let $X = \nabla f_j(\mathbf{x}^k)$ be a random variable (due to $j \sim$ Uniform$(\{1, \cdots, n\})$).

◦ Let $Y = \nabla f_j(\tilde{\mathbf{x}})$ be another random variable, and $\tilde{\mathbf{x}}$ is a particularly selected point.

A generalized estimator: $R_\alpha = \alpha(X - Y) + \mathbb{E}[Y]$

- $\mathbb{E}[R_\alpha] = \alpha\mathbb{E}[X] + (1-\alpha)\mathbb{E}[Y]$
- $\mathrm{Var}(R_\alpha) = \alpha^2(\mathrm{Var}(X) + \mathrm{Var}(Y) - 2\mathrm{Cov}(X, Y))$

Observations:
◦ When $\alpha = 1$, R_α becomes unbiased, i.e., $\mathbb{E}[R_\alpha] = \mathbb{E}[X]$.
◦ If $\mathrm{Cov}(X, Y)$ is large enough (X and Y are correlated enough), $\mathrm{Var}(R_\alpha) \le \mathrm{Var}(X)$.

How could we use this information to construct our estimate?

Variance reduction techniques: SVRG

- Select the stochastic gradient ∇f_{i_k}, and compute a gradient estimate

$$\mathbf{r}_k = \nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\tilde{\mathbf{x}}) + \nabla f(\tilde{\mathbf{x}}).$$

- As $\tilde{\mathbf{x}} \to \mathbf{x}^\star$ and $\mathbf{x}^k \to \mathbf{x}^\star$, we have

$$\nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\tilde{\mathbf{x}}) + \nabla f(\tilde{\mathbf{x}}) \to 0.$$

- As a result, we can ensure the following

$$\mathbb{E}\left[\|\nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\tilde{\mathbf{x}}) + \nabla f(\tilde{\mathbf{x}})\|^2\right] \to 0.$$

Remarks:

- Remember the generalized estimator: $R_\alpha = \alpha(X - Y) + \mathbb{E}[Y]$.
- For SVRG, $\alpha = 1$, $X = \nabla f_{i_k}(\mathbf{x}^k)$ and $Y = \nabla f_{i_k}(\tilde{\mathbf{x}})$.
- We will see how $\tilde{\mathbf{x}}$ is computed!

Stochastic gradient algorithm with variance reduction

Stochastic gradient with variance reduction (SVRG) [11, 21]

1. Choose $\widetilde{\mathbf{x}}^0 \in \mathbb{R}^p$ as a starting point and $\gamma > 0$ and $q \in \mathbb{N}_+$.
2. For $s = 0, 1, 2 \cdots$, perform:
   - 2a. $\widetilde{\mathbf{x}} = \widetilde{\mathbf{x}}^s, \quad \widetilde{\mathbf{v}} = \nabla f(\widetilde{\mathbf{x}}), \quad \mathbf{x}^0 = \widetilde{\mathbf{x}}$.
   - 2b. For $k = 0, 1, \cdots q-1$, perform:

$$\begin{cases} \text{Pick } i_k \in \{1, \ldots, n\} \text{ uniformly at random} \\ \mathbf{r}_k = \nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\widetilde{\mathbf{x}}) + \widetilde{\mathbf{v}} \\ \mathbf{x}^{k+1} := \mathbf{x}^k - \gamma \mathbf{r}_k, \end{cases} \qquad (1)$$

   - 2c. Update $\widetilde{\mathbf{x}}^{s+1} = \frac{1}{m} \sum_{j=0}^{q-1} \mathbf{x}^j$.

Features

- The SVRG method uses a multistage scheme to reduce the variance of the stochastic gradient $\mathbf{r}_k$.
- Learning rate γ does not necessarily tend to 0 while $\mathbf{x}^k$ and $\widetilde{\mathbf{x}}^s$ tend to $\mathbf{x}_\star$.
- Each stage, SVRG uses $n + 2q$ component gradient evaluations.
- n for the full gradient at the beginning of each stage, and $2q$ for each of the q stochastic gradient steps.

Convergence analysis

Assumption A5.

(i) f is μ-strongly convex

(ii) The learning rate $0 < \gamma < 1/(4L_{\max})$, where $L_{\max} = \max_{1\leq j\leq n} L_j$.

(iii) q is large enough such that

$$\kappa = \frac{1}{\mu\gamma(1-4\gamma L_{\max})q} + \frac{4\gamma L_{\max}(q+1)}{(1-4\gamma L_{\max})q} < 1.$$

Theorem

Assumptions:

- *The sequence $\{\widetilde{\mathbf{x}}^s\}_{k\geq 0}$ is generated by SVRG.*
- *Assumption A5 is satisfied.*

Conclusion: *Linear convergence is obtained:*

$$\mathbb{E}f(\widetilde{\mathbf{x}}^s) - f(\mathbf{x}^\star) \leq \kappa^s (f(\widetilde{\mathbf{x}}^0) - f(\mathbf{x}^\star)).$$

Choice of γ and q, and complexity

Chose γ and q such that $\kappa \in (0,1)$:

For example

$$\gamma = 0.1/L_{\max}, q = 100(L_{\max}/\mu) \Longrightarrow \kappa \approx 5/6.$$

Complexity

$$\mathbb{E}f(\widetilde{\mathbf{x}}^s) - f(\mathbf{x}^\star) \leq \varepsilon, \quad \text{when } s \geq \log((f(\widetilde{\mathbf{x}}^0) - f(\mathbf{x}^\star))/\epsilon)/\log(\kappa^{-1})$$

- Each stage needs $n + 2q$ component gradient evaluations
- With $q = \mathcal{O}(L_{\max}/\mu)$, we obtain an overall complexity of

$$\mathcal{O}\Big((n + L_{\max}/\mu)\log(1/\epsilon)\Big).$$

Comparison: GD vs. SGD vs. SVRG

- GD update:

$$\left\{\ \mathbf{x}^{k+1} := \mathbf{x}^k - \gamma \nabla f(\mathbf{x}^k),\right.$$

- SGD update:

$$\left\{\ \mathbf{x}^{k+1} := \mathbf{x}^k - \gamma \nabla f_{i_k}(\mathbf{x}^k),\right.$$

- SVRG update:

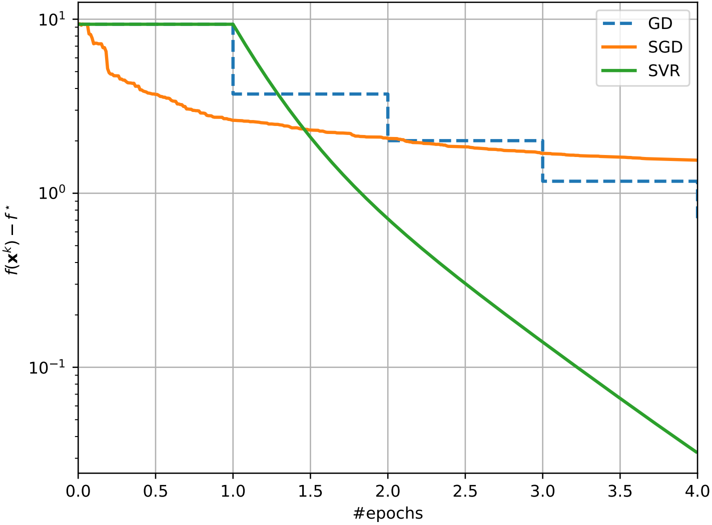
$$\begin{cases} \mathbf{r}_k = \nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\widetilde{\mathbf{x}}) + \nabla f(\tilde{\mathbf{x}}) \\ \mathbf{x}^{k+1} := \mathbf{x}^k - \gamma \mathbf{r}_k, \end{cases}$$

	SGD	SVRG	GD
Requires gradient storage?	no	no	no
Epoch-based	no	yes	no
Parameters	stepsize	stepsize & epoch length	stepsize
Gradient evaluations	1 per iteration	$n + 2q$ per epoch	n per iteration

Table: Comparisons of SGD, SVRG and GD [6]

- Recall that $q = \mathcal{O}(L_{\max}/\mu)$ is the epoch length for SVRG.

Example: ℓ_2-regularized least squares with synthetic data



Taxonomy of algorithms

$$f^\star := \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) := \frac{1}{n}\sum_{j=1}^{n} f_j(\mathbf{x}) \right\}.$$

- $f(\mathbf{x}) = \frac{1}{n}\sum_{j=1}^{n} f_j(\mathbf{x})$: μ-strongly convex with L-Lipschitz continuous gradient.

SVRG	GD	SGD
Linear	Linear	Sublinear

Table: Rate of convergence.

- $\kappa = L/\mu$.

SVRG	GD	SGD
$\mathcal{O}((n+\kappa)\log(1/\varepsilon))$	$\mathcal{O}((n\kappa)\log(1/\varepsilon))$	$1/\varepsilon$

Table: Complexity to obtain ε-solution.

The variance reduction zoo: convex

Setting	Algorithm	Lower bound	Complexity bound
L-smooth f_i's with bounded variance	Gradient descent	$L\Delta_0 \min\{\sigma/\epsilon^3, \sqrt{n}/\epsilon^2\}$ [10]	$nL\Delta_0/\epsilon^2$
	SVRG ($B_k = 1$) [16]		$nL\Delta_0/\epsilon^2$
	SVRG ($B_k = \Omega(n^{2/3})$) [16]		$n^{2/3}L\Delta_0/\epsilon^2$
	SAGA ($B_k = 1$) [16]		$nL\Delta_0/\epsilon^2$
	SAGA ($B_k = \Omega(n^{2/3})$) [16]		$n^{2/3}L\Delta_0/\epsilon^2$
	SpiderBoost [19]		$\sqrt{n}L\Delta_0/\epsilon^2$
	SpiderBoost-M [19]		$\sqrt{n}L\Delta_0/\epsilon^2$
	Spider [10]		$L\Delta_0 \min\{\sigma/\epsilon^3, \sqrt{n}/\epsilon^2\}$
	PAGE [15]		$L\Delta_0 \min\{\sigma/\epsilon^3, \sqrt{n}/\epsilon^2\}$
f is μ-SCVX and L-smooth f_i's are average L-smooth	KatyushaX [3]	$(n + n^{3/4}\sqrt{\frac{L}{\mu}}) \log \frac{\Delta_0}{\epsilon}$ [22]	$(n + n^{3/4}\sqrt{\frac{L}{\mu}}) \log \frac{\Delta_0}{\epsilon}$
f is CVX and L-smooth f_i's are average L-smooth	KatyushaX [3]	$n + n^{3/4}\sqrt{\frac{LD_0^2}{\epsilon}}$ [23]	$n + n^{3/4}\sqrt{\frac{LD_0^2}{\epsilon}}$

Remarks:
- ◦ Complexity ((S)CVX f): total number of stochastic first-order oracle calls to find $\mathbf{x}^\star_\epsilon$ with $\mathbb{E}[f(\mathbf{x}^\star_\epsilon) - f(\mathbf{x}^\star)] \leq \epsilon$.
- ◦ $\Delta_0 = f(\mathbf{x}^0) - f^\star$, $D_0 = \|\mathbf{x}^0 - \mathbf{x}^\star\|$.
- ◦ Bounded variance: $\mathbb{E}_i[\|\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x})\|^2] \leq \sigma^2 \quad \forall \mathbf{x}$.
- ◦ Average L-smooth: $\mathbb{E}_i[\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\|^2] \leq L^2\|\mathbf{x} - \mathbf{y}\|^2 \ \forall \mathbf{x}, \mathbf{y}$.

Variance-reduction for non-convex problems

SVRG estimator vs. a recursive estimator

◦ SVRG update:

$$\begin{cases} \mathbf{r}_1 = \nabla f(\tilde{\mathbf{x}}) \\ \mathbf{r}_k := \nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\widetilde{\mathbf{x}}) + \nabla f(\tilde{\mathbf{x}}) \\ \mathbf{x}^{k+1} := \mathbf{x}^k - \gamma \mathbf{r}_k, \end{cases}$$

◦ Spider [10] update:

$$\begin{cases} \mathbf{r}_1 = \nabla f(\tilde{\mathbf{x}}) \\ \mathbf{r}_k := \nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\widetilde{\mathbf{x}}) + \mathbf{r}_{k-1} \\ \mathbf{x}^{k+1} := \mathbf{x}^k - \gamma \mathbf{r}_k, \end{cases}$$

Spider [10]

1. Choose $\mathbf{x}^0 \in \mathbb{R}^p$ as a starting point and $\gamma = \epsilon / L$.
2. For $k = 0, 1, 2, \ldots,$ perform:
   - 2a. If $k \bmod n = 0$, do:
     $\mathbf{r}_k = \nabla f(\mathbf{x}^k)$
     else:
     Pick $i_k \in \{1, \ldots, n\}$ uniformly at random
     $\mathbf{r}_k = \nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\mathbf{x}^{k-1}) + \mathbf{r}_{k-1}$
   - 2b. Update $\mathbf{x}^{k+1} := \mathbf{x}^k - \frac{\gamma}{\| \mathbf{r}_k \|} \mathbf{r}_k$
3. Return $\mathbf{x}^k$

Remarks:

◦ Sample complexity: $O\left(n + \sqrt{n}\frac{\Delta L}{\epsilon^2}\right)$.

◦ Sets the final accuracy apriori.

◦ Step-size depends on ϵ and L.

Adaptive variance-reduction for non-convex problems

AdaSpider [13]
1. Choose $\mathbf{x}^0 \in \mathbb{R}^p$ as a starting point.
2. For $k = 0, 1, 2 \cdots$, perform:
2a. If k mod $n = 0$, do:
$\mathbf{r}_k = \nabla f(\mathbf{x}^k)$
else:
Pick $i_k \in \{1, \ldots, n\}$ uniformly at random
$\mathbf{r}_k = \nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\mathbf{x}^{k-1}) + \mathbf{r}_{k-1}$
2b. Compute $\gamma_k := 1/\left(n^{1/4}\sqrt{n^{1/2} + \sum_{i=0}^{k} \Vert\mathbf{r}_i\Vert^2}\right)$
2c. Update $\mathbf{x}^{k+1} := \mathbf{x}^k - \gamma_k \mathbf{r}_k$
3. Return $\mathbf{x}^k$

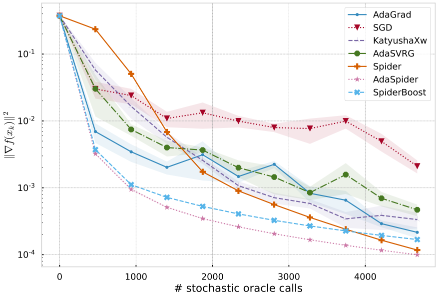
Theorem

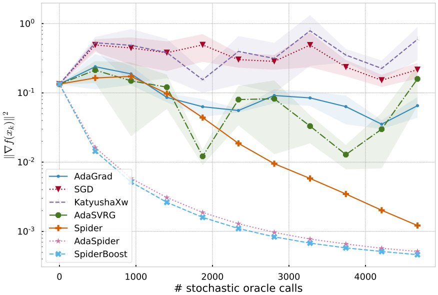
Let $\Delta_0 = f(\mathbf{x}^0) - \min_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x})$. The sequence $\mathbf{x}^0, \cdots, \mathbf{x}^k$ generated by AdaSpider satisfies:

$$\frac{1}{k}\sum_{i=0}^{k-1} \mathbb{E}[\Vert\nabla f(\mathbf{x}^i)\Vert] \leq O\left(n^{1/4}\frac{\Delta_0 + L^2}{\sqrt{k}}\log(k)\right), \quad \textit{with sample complexity } \tilde{O}\left(n + \sqrt{n}\frac{\Delta_0^2 + L^4}{\varepsilon^2}\right).$$

Performance of AdaSpider

- Image classification with neural networks (spoiler alert!) trained with cross entropy loss.
- AdaGrad [8], KatyushaXw [2], AdaSVRG[7], Spider [10], SpiderBoost [20].





The variance reduction zoo: non-convex

Setting	Algorithm	Lower bound	Complexity bound
f is α-weakly CVX and L-smooth f_i's are average L-smooth	Spider [10]	$\frac{\Delta_0}{\epsilon^2} \min\{n^{3/4}\sqrt{\alpha L}, \sqrt{n}L\}$ [23]	$\frac{\Delta_0}{\epsilon^2} \min\{n^{3/4}\sqrt{\alpha L}, \sqrt{n}L\}$
f_i's are α-weakly CVX and L-smooth	Natasha [1]	$\frac{\Delta_0}{\epsilon^2} \min\{\sqrt{n\alpha L}, L\}$ [23]	$\frac{\Delta_0}{\epsilon^2} \min\{\sqrt{n\alpha L}, \sqrt{n}L\}$
f is non-CVX f_i's are non-CVX and L-smooth	AdaSpider [13]	$\frac{\Delta_0 L}{\epsilon^2}\sqrt{n}$ [23, 10]	$\tilde{O}\left(n + \frac{\Delta_0^2 + L^4}{\epsilon^2}\sqrt{n}\right)$

Remarks:
- Complexity (nonCVX f): total number of stochastic first-order oracle calls to find $\mathbf{x}^\star_\epsilon$ with $\mathbb{E}[\|\nabla f(\mathbf{x}^\star_\epsilon)\|^2] \leq \epsilon^2$.
- $\Delta_0 = f(\mathbf{x}^0) - f^\star$, $D_0 = \|\mathbf{x}^0 - \mathbf{x}^\star\|$.
- Bounded variance: $\mathbb{E}_i[\|\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x})\|^2] \leq \sigma^2 \quad \forall \mathbf{x}$.
- L-smooth: $\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\| \; \forall \mathbf{x}, \mathbf{y}$.
- $f(\mathbf{x})$ is α-weakly convex if $f(\mathbf{x}) + \frac{\alpha}{2}\|\mathbf{x}\|^2$ is convex $\forall \mathbf{x}$.

Deep learning outline

- In the sequel,
  - Introduction to deep learning
  - The deep learning paradigm
  - Challenges in deep learning theory and applications
- Next class
  - Generalization in deep learning

Remark about notation

The Deep Learning literature might use a different notation:

	Our lectures	DL literature
data/sample	$\mathbf{a}$	$\mathbf{x}$
label	b	y
bias	μ	b
weight	$\mathbf{x}, \mathbf{X}$	$\mathbf{w}, \mathbf{W}$

Power of linear classifiers–I

Problem (Recall: Logistic regression)

Given a sample vector $\mathbf{a}_i \in \mathbb{R}^d$ *and a binary class label* $b_i \in \{-1, +1\}$ $(i = 1, \ldots, n)$, *we define the conditional probability of* b_i *given* $\mathbf{a}_i$ *as follows:*

$$\mathbb{P}(b_i|\mathbf{a}_i, \mathbf{x}) \propto 1/(1 + e^{-b_i\langle \mathbf{x}, \mathbf{a}_i\rangle}),$$

where $\mathbf{x} \in \mathbb{R}^d$ *is some weight vector.*

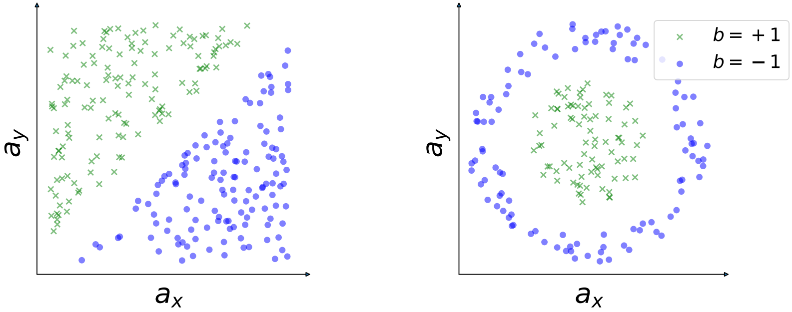


Figure: Linearly separable versus nonlinearly separable dataset

Power of linear classifiers–II

- Lifting dimensions to the rescue
  - Convex optimization objective
  - Side effect: The curse-of-dimensionality
  - Possible to avoid via kernel methods, such as SVMs

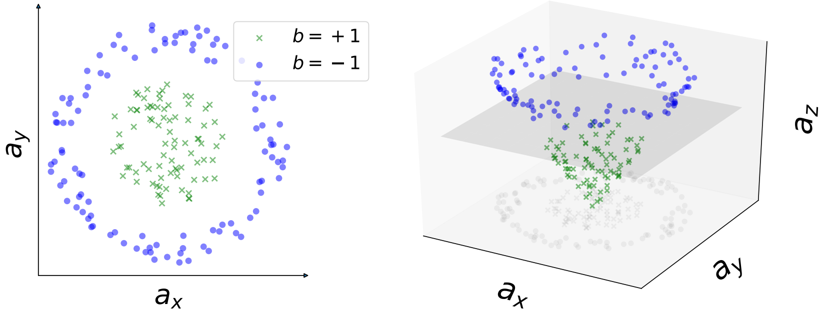


Figure: Non-linearly separable data (left). Linearly separable in $\mathbb{R}^3$ via $\mathbf{a}_z = \sqrt{\mathbf{a}_x^2 + \mathbf{a}_y^2}$ (right).

An important alternative for non-linearly separable data

1-hidden-layer neural network with m neurons (fully-connected architecture):

- Parameters: $\mathbf{X}_1 \in \mathbb{R}^{m \times d}$, $\mathbf{X}_2 \in \mathbb{R}^{c \times m}$ (weights), $\mu_1 \in \mathbb{R}^m, \mu_2 \in \mathbb{R}^c$ (biases)
- Activation function: $\sigma : \mathbb{R} \to \mathbb{R}$

$$h_{\mathbf{x}}(\mathbf{a}) := \underset{\substack{\uparrow \\ \text{input}}}{\begin{bmatrix} \mathbf{a} \end{bmatrix}}$$

An important alternative for non-linearly separable data

1-hidden-layer neural network with m neurons (fully-connected architecture):

- Parameters: $\mathbf{X}_1 \in \mathbb{R}^{m \times d}$, $\mathbf{X}_2 \in \mathbb{R}^{c \times m}$ (weights), $\mu_1 \in \mathbb{R}^m, \mu_2 \in \mathbb{R}^c$ (biases)
- Activation function: $\sigma : \mathbb{R} \to \mathbb{R}$

$$h_{\mathbf{x}}(\mathbf{a}) := \underset{\text{weight}}{\underset{\downarrow}{\begin{bmatrix} \mathbf{X}_1 \end{bmatrix}}} \underset{\text{input}}{\underset{\downarrow}{\begin{bmatrix} \mathbf{a} \end{bmatrix}}}$$

An important alternative for non-linearly separable data

1-hidden-layer neural network with m neurons (fully-connected architecture):

- Parameters: $\mathbf{X}_1 \in \mathbb{R}^{m \times d}$, $\mathbf{X}_2 \in \mathbb{R}^{c \times m}$ (weights), $\mu_1 \in \mathbb{R}^m, \mu_2 \in \mathbb{R}^c$ (biases)
- Activation function: $\sigma : \mathbb{R} \to \mathbb{R}$

$$h_{\mathbf{x}}(\mathbf{a}) := \underbrace{\begin{bmatrix} \quad \mathbf{X}_1 \quad \end{bmatrix}}_{\text{weight}} \underbrace{\begin{bmatrix} \mathbf{a} \end{bmatrix}}_{\text{input}} + \underbrace{\begin{bmatrix} \mu_1 \end{bmatrix}}_{\text{bias}}$$

An important alternative for non-linearly separable data

1-hidden-layer neural network with m neurons (fully-connected architecture):

- Parameters: $\mathbf{X}_1 \in \mathbb{R}^{m \times d}$, $\mathbf{X}_2 \in \mathbb{R}^{c \times m}$ (weights), $\mu_1 \in \mathbb{R}^m, \mu_2 \in \mathbb{R}^c$ (biases)
- Activation function: $\sigma : \mathbb{R} \to \mathbb{R}$

$$h_{\mathbf{x}}(\mathbf{a}) := \underbrace{\overset{\text{activation}}{\overset{\downarrow}{\color{red}\sigma}} \color{red}{\Bigg(} \color{black} \overset{\text{weight}}{\overset{\downarrow}{\Big[\ \mathbf{X}_1\ \Big]}} \overset{\text{input}}{\overset{\downarrow}{\color{green}\big[\mathbf{a}\big]}} + \overset{\text{bias}}{\overset{\downarrow}{\color{blue}\big[\mu_1\big]}} \color{red}{\Bigg)}}_{\text{hidden layer = learned features}}$$

An important alternative for non-linearly separable data

1-hidden-layer neural network with m neurons (fully-connected architecture):

- Parameters: $\mathbf{X}_1 \in \mathbb{R}^{m \times d}$, $\mathbf{X}_2 \in \mathbb{R}^{c \times m}$ (weights), $\mu_1 \in \mathbb{R}^m, \mu_2 \in \mathbb{R}^c$ (biases)
- Activation function: $\sigma : \mathbb{R} \to \mathbb{R}$

$$h_{\mathbf{x}}(\mathbf{a}) := \begin{bmatrix} \quad \mathbf{X}_2 \quad \end{bmatrix} \underbrace{\overset{\text{activation}}{\overset{\downarrow}{\sigma}} \left(\overset{\text{weight}}{\overset{\downarrow}{\begin{bmatrix} \quad \mathbf{X}_1 \quad \end{bmatrix}}} \overset{\text{input}}{\overset{\downarrow}{\begin{bmatrix} \mathbf{a} \end{bmatrix}}} + \overset{\text{bias}}{\overset{\downarrow}{\begin{bmatrix} \mu_1 \end{bmatrix}}} \right)}_{\text{hidden layer = learned features}}$$

An important alternative for non-linearly separable data

1-hidden-layer neural network with m neurons (fully-connected architecture):

∘ Parameters: $\mathbf{X}_1 \in \mathbb{R}^{m\times d}$, $\mathbf{X}_2 \in \mathbb{R}^{c\times m}$ (weights), $\mu_1 \in \mathbb{R}^m, \mu_2 \in \mathbb{R}^c$ (biases)
∘ Activation function: $\sigma : \mathbb{R} \to \mathbb{R}$

$$h_{\mathbf{x}}(\mathbf{a}) := \Big[\ \mathbf{X}_2\ \Big] \underbrace{\overset{\text{activation}}{\overset{\downarrow}{\sigma}} \left(\overset{\text{weight}}{\overset{\downarrow}{\Big[\ \mathbf{X}_1\ \Big]}} \overset{\text{input}}{\overset{\downarrow}{\Big[\ \mathbf{a}\ \Big]}} + \overset{\text{bias}}{\overset{\downarrow}{\Big[\ \mu_1\ \Big]}} \right)}_{\text{hidden layer} \,=\, \text{learned features}} + \overset{\text{bias}}{\overset{\downarrow}{\Big[\ \mu_2\ \Big]}}$$

An important alternative for non-linearly separable data

1-hidden-layer neural network with m neurons (fully-connected architecture):

- Parameters: $\mathbf{X}_1 \in \mathbb{R}^{m \times d}$, $\mathbf{X}_2 \in \mathbb{R}^{c \times m}$ (weights), $\mu_1 \in \mathbb{R}^m, \mu_2 \in \mathbb{R}^c$ (biases)
- Activation function: $\sigma : \mathbb{R} \to \mathbb{R}$

$$h_{\mathbf{x}}(\mathbf{a}) := \left[\ \mathbf{X}_2\ \right] \underbrace{\overset{\text{activation}}{\sigma}\left(\left[\ \overset{\text{weight}}{\mathbf{X}_1}\ \right] \left[\overset{\text{input}}{\mathbf{a}} \right] + \left[\overset{\text{bias}}{\mu_1} \right] \right)}_{\text{hidden layer = learned features}} + \left[\overset{\text{bias}}{\mu_2} \right], \qquad \mathbf{x} := [\mathbf{X}_1, \mathbf{X}_2, \mu_1, \mu_2]$$

An important alternative for non-linearly separable data

1-hidden-layer neural network with m neurons (fully-connected architecture):

- Parameters: $\mathbf{X}_1 \in \mathbb{R}^{m\times d}$, $\mathbf{X}_2 \in \mathbb{R}^{c\times m}$ (weights), $\mu_1 \in \mathbb{R}^m, \mu_2 \in \mathbb{R}^c$ (biases)
- Activation function: $\sigma : \mathbb{R} \to \mathbb{R}$

$$h_{\mathbf{x}}(\mathbf{a}) := \Big[\ \mathbf{X}_2\ \Big] \underbrace{\overset{\text{activation}}{\sigma} \left(\Big[\ \overset{\text{weight}}{\mathbf{X}_1}\ \Big] \Big[\overset{\text{input}}{\mathbf{a}}\Big] + \Big[\overset{\text{bias}}{\mu_1}\Big] \right)}_{\text{hidden layer} \,=\, \text{learned features}} + \Big[\overset{\text{bias}}{\mu_2}\Big], \qquad \mathbf{x} := [\mathbf{X}_1, \mathbf{X}_2, \mu_1, \mu_2]$$

recursively repeat activation $+$ affine transformation to obtain "deeper" networks.

Why neural networks?: An approximation theoretic motivation

Theorem (Universal approximation [5])

Let $\sigma(\cdot)$ be a nonconstant, bounded, and increasing continuous function. Let $I_d = [0,1]^d$. The space of continuous functions on I_d is denoted by $\mathcal{C}(I_d)$.

Given $\epsilon > 0$ and $g \in \mathcal{C}(I_d)$ there exists a 1-hidden-layer network h with m neurons such that h is an ϵ-approximation of g, i.e.,

$$\sup_{\mathbf{a} \in I_d} |g(\mathbf{a}) - h(\mathbf{a})| \leq \epsilon$$

Caveat

The number of neurons m needed to approximate some function g can be arbitrarily large!

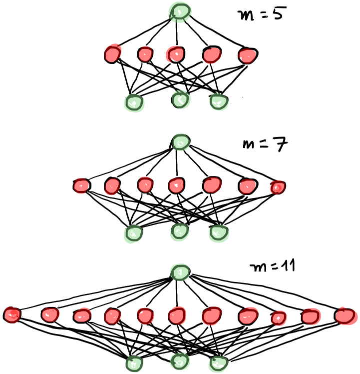


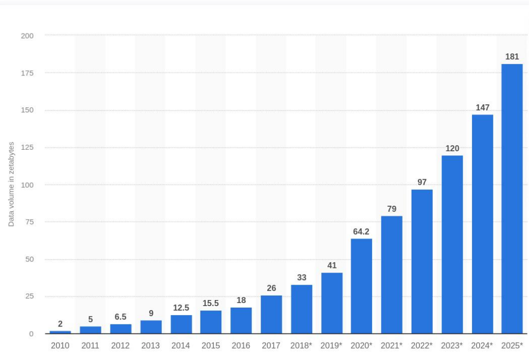
Figure: networks of increasing width

Why were NNs not popular before 2010?

- too big to optimize!
- did not have enough data
- could not find the optimum via algorithms

Why were NNs not popular before 2010?

- too big to optimize!
- did not have enough data
- could not find the optimum via algorithms



Supervised learning: Multi-class classification

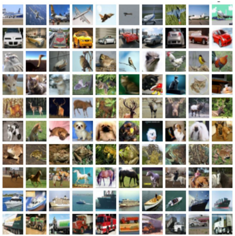


Figure: CIFAR10 dataset: 60000 32x32 color images (3 channels) from 10 classes

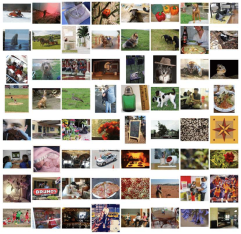


Figure: Imagenet dataset: 14 million color images (varying resolution, 3 channels) from 21K classes

Goal

Image-label pairs $(\mathbf{a}, b) \subseteq \mathbb{R}^d \times \{1, \ldots, c\}$ follow an unknown distibution $\mathbb{P}$. Find $h : \mathbb{R}^d \to \{1, \ldots, c\}$ with minimum *misclassification probability*

$$\min_{h \in \mathcal{H}} \mathbb{P}(h(\mathbf{a}) \neq b)$$

2010-today: Deep Learning becomes popular again

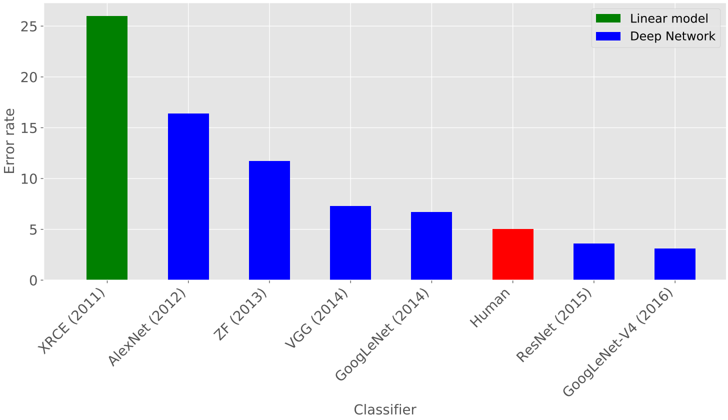


Figure: Error rate on the ImageNet challenge, for different network architectures.

2010-today: Deep Learning becomes popular again

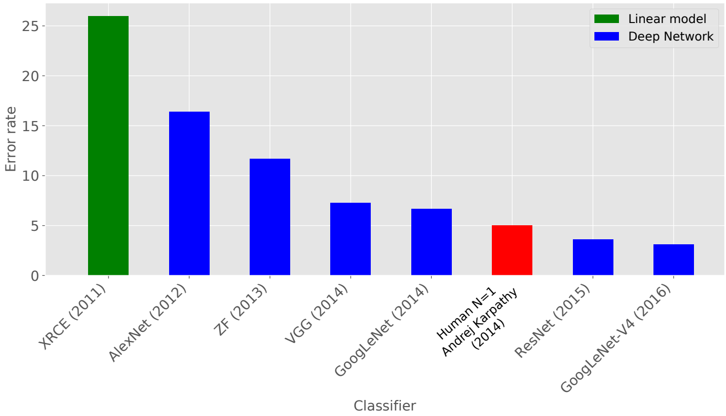


Figure: Error rate on the ImageNet challenge, for different network architectures [17, 12].

Convolutional architectures in Computer Vision tasks

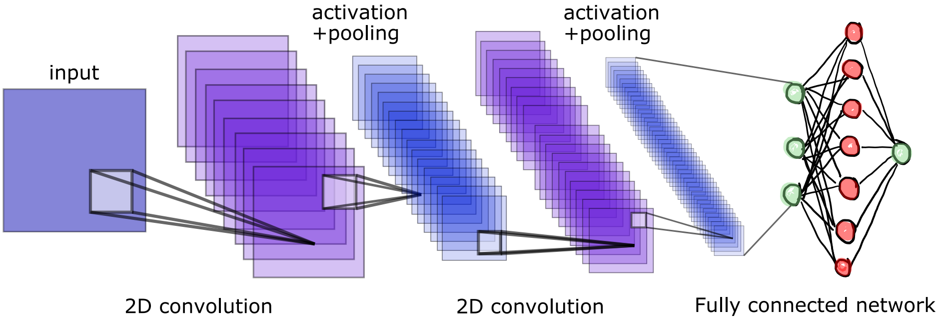
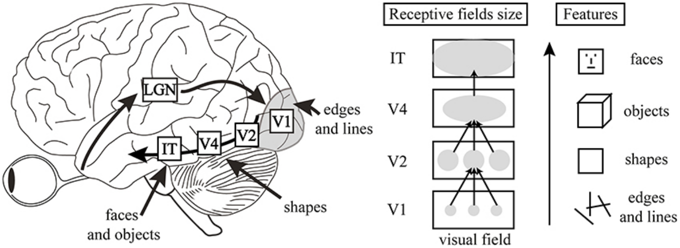
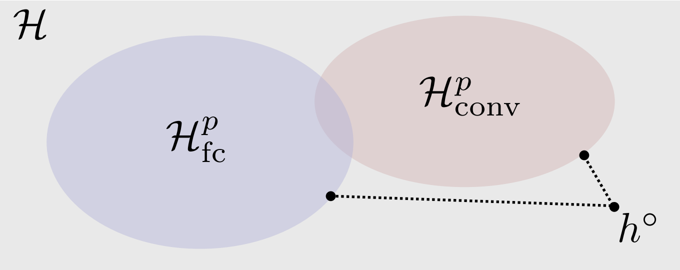


Figure: "Locality" structure of a 2D deep convolutional neural network.

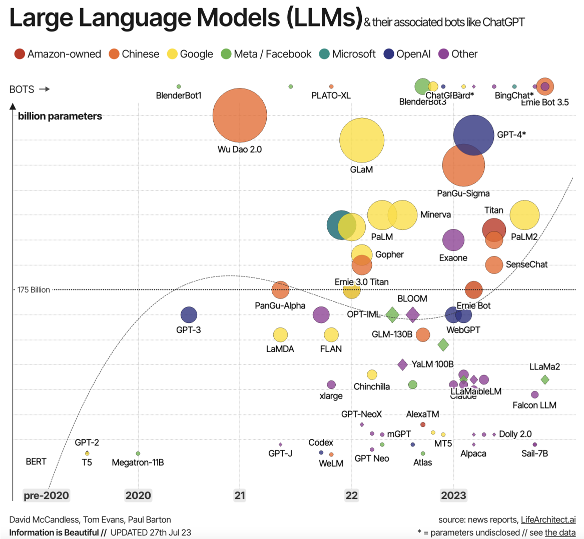
Inductive Bias: Why convolution works so well in Computer Vision tasks?



h°	true unknown function
$\mathcal{H}$	space of all functions
$\mathcal{H}^p_{\text{fc}}$	fully-connected networks with p parameters
$\mathcal{H}^p_{\text{conv}}$	convolutional networks with p parameters



The era of model scaling



From: `https://informationisbeautiful.net/visualizations/the-rise-of-generative-ai-large-language-models-llms-like-chatgpt/`

The landscape of ERM with multilayer networks

Recall: Empirical risk minimization (ERM)

Let $h_{\mathbf{x}} : \mathbb{R}^n \to \mathbb{R}$ be network and let $\{(\mathbf{a}_i, b_i)\}_{i=1}^n$ be a sample with $b_i \in \{-1, 1\}$ and $\mathbf{a}_i \in \mathbb{R}^n$. The *empirical risk minimization* (ERM) is defined as follows

$$\min_{\mathbf{x}} \left\{ R_n(\mathbf{x}) := \frac{1}{n} \sum_{i=1}^{n} L(h_{\mathbf{x}}(\mathbf{a}_i), b_i) \right\} \tag{2}$$

where $L(h_{\mathbf{x}}(\mathbf{a}_i), b_i)$ is the loss on the sample $(\mathbf{a}_i, b_i)$ and $\mathbf{x}$ are the parameters of the network.

Some frequently used loss functions

- $L(h_{\mathbf{x}}(\mathbf{a}), b) = \log(1 + \exp(-b \cdot h_{\mathbf{x}}(\mathbf{a})))$ (logistic loss)
- $L(h_{\mathbf{x}}(\mathbf{a}), b) = (b - h_{\mathbf{x}}(\mathbf{a}))^2$ (squared error)
- $L(h_{\mathbf{x}}(\mathbf{a}), b) = \max(0, 1 - b \cdot h_{\mathbf{x}}(\mathbf{a}))$ (hinge loss)

The landscape of ERM with multilayer networks

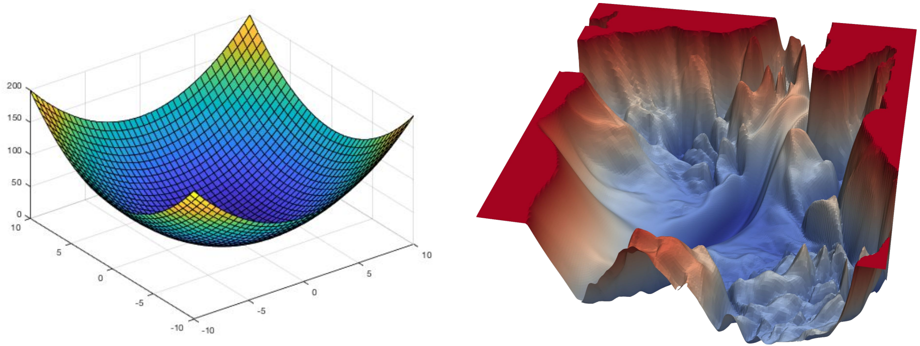


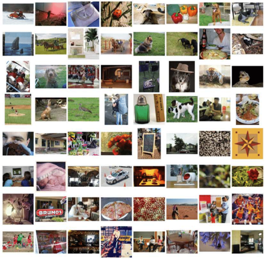
Figure: convex (left) vs non-convex (right) optimization landscape [14]

Conventional wisdom in ML until 2010:
Simple models + simple errors

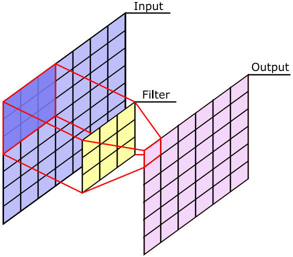
The landscape of ERM with multilayer networks

1. Liu, Chaoyue, et al. *Loss landscapes and optimization in over-parameterized non-linear systems and neural networks*, in Applied and Computational Harmonic Analysis, 2022.
2. Zhang, Yaoyu, et al. *Embedding Principle of Loss Landscape of Deep Neural Networks*, in Advances in Neural Information Processing Systems (NeurIPS), 2021.
3. Simsek, Berfin, et al. *Geometry of the loss landscape in overparameterized neural networks: Symmetries and invariances*, inInternational Conference on Machine Learning, 2021.
4. Foret, Pierre, et al. *Sharpness-aware minimization for efficiently improving generalization*. arXiv preprint arXiv:2010.01412, 2020.
5. Jiang, Yiding, et al. *Fantastic generalization measures and where to find them.* arXiv preprint arXiv:1912.02178, 2019.
6. Dziugaite, Gintare Karolina, and Daniel M. Roy. *Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data.* arXiv preprint arXiv:1703.11008, 2017.
7. Keskar, Nitish Shirish, et al. *On large-batch training for deep learning: Generalization gap and sharp minima.* arXiv preprint arXiv:1609.04836, 2016.

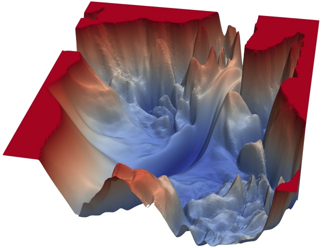
The deep learning paradigm



(a) Massive datasets



(b) Inductive bias from large and complex architectures



(c) ERM using stochastic non-convex first-order optimization algorithms (SGD)

Figure: Most common components in a Deep Learning Pipeline

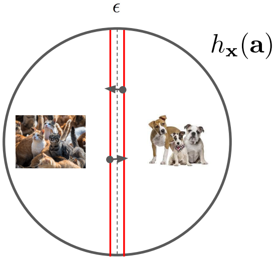
Challenges in DL/ML applications: Robustness (I)

(a) Turtle classified as rifle [4].

(b) Stop sign classified as 45 mph sign [9].

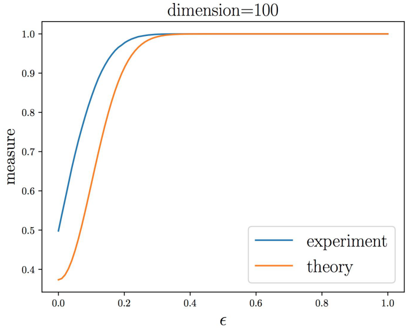
Figure: Natural or human-crafted modifications that trick neural networks used in computer vision tasks

Challenges in DL/ML applications: Robustness (II)



$$h_{\mathbf{x}}(\mathbf{a}+\epsilon) \approx h_{\mathbf{x}}(\mathbf{a}) + \langle \epsilon, \nabla h_{\mathbf{x}}(\mathbf{a}) \rangle$$
$$|h_{\mathbf{x}}(\mathbf{a}+\epsilon) - h_{\mathbf{x}}(\mathbf{a})| \leq \|\epsilon\| \|\nabla h_{\mathbf{x}}(\mathbf{a})\|$$

(a) Linear classifier on data distributed on a sphere



(b) Concentration of measure phenomenon on high dimensions

Figure: Understanding the robustness of a classifier in high-dimensional spaces [18]

Challenges in DL/ML applications: Robustness (References I)

1. Dominguez-Olmedo Ricardo et al., *On the Adversarial Robustness of Causal Algorithmic Recourse*, International Conference on Machine Learning, 2022.
2. Wang Yunjuan et al., *Adversarial Robustness is at Odds with Lazy Training*, in Advances in Neural Information Processing Systems (NeurIPS), 2022.
3. Zhu Zhenyu et al., *Robustness in deep learning: The good (width), the bad (depth), and the ugly (initialization)*, in Advances in Neural Information Processing Systems (NeurIPS), 2022.
4. Xing Yue et al., *Why Do Artificially Generated Data Help Adversarial Robustness*, Advances in Neural Information Processing Systems (NeurIPS), 2022.
5. D. Krueger et al., *Out-of-Distribution Generalization via Risk Extrapolation*, in International Conference on Machine Learning, 2021.
6. R. Bhattacharjee, S. Jha, and K. Chaudhuri, *Sample Complexity of Robust Linear Classification on Separated Data*, in International Conference on Machine Learning, 2021.

Challenges in DL/ML applications: Robustness (References II)

1. F. Tramèr, N. Carlini, W. Brendel, and A. Ma, *On Adaptive Attacks to Adversarial Example Defenses*, in Advances in Neural Information Processing Systems, 2020.
2. A. D'Amour et al., *Underspecification Presents Challenges for Credibility in Modern Machine Learning*, arXiv:2011.03395, 2020.
3. L. Chen, Y. Min, M. Zhang, and A. Karbasi, *More Data Can Expand the Generalization Gap Between Adversarially Robust and Standard Models*, in International Conference on Machine Learning, 2020.
4. F. Tramer, J. Behrmann, N. Carlini, N. Papernot, and J.-H. Jacobsen, *Fundamental Tradeoffs between Invariance and Sensitivity to Adversarial Perturbations*, in International Conference on Machine Learning, 2020.
5. E. Rosenfeld, P. K. Ravikumar, and A. Risteski, *The Risks of Invariant Risk Minimization*, in International Conference on Learning Representations, 2020.
6. T. Li, A. Beirami, M. Sanjabi, and V. Smith, *Tilted Empirical Risk Minimization*, in International Conference on Learning Representations, 2020.
7. T. Pang, K. Xu, Y. Dong, C. Du, N. Chen, and J. Zhu, *Rethinking Softmax Cross-Entropy Loss For Adversarial Robustness*, in International Conference on Learning Representations, 2020.
8. Y. Min, L. Chen, and A. Karbasi, *The Curious Case of Adversarially Robust Models: More Data Can Help, Double Descend, or Hurt Generalization*, arXiv:2002.11080, 2020.
9. R. Geirhos et al., *Shortcut Learning in Deep Neural Networks*, in Nature Machine Intelligence, 2020.

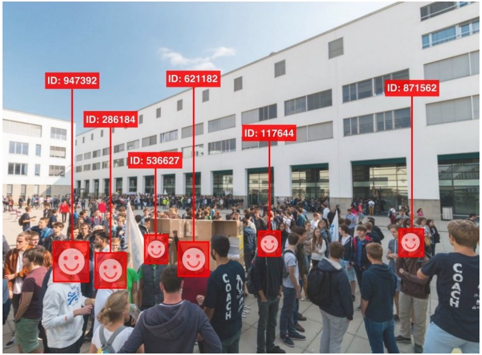
Challenges in DL/ML applications: Robustness (References III)

1. A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, *Adversarial Examples Are Not Bugs, They Are Features*, in Advances in Neural Information Processing Systems, 2019.
2. J. Gilmer, N. Ford, N. Carlini, and E. Cubuk, *Adversarial Examples Are a Natural Consequence of Test Error in Noise*, in International Conference on Machine Learning, 2019.
3. Shafahi A., Ronny Huang, W., Studer, C., Feizi, S. and Goldstein, T. *Are adversarial examples inevitable?*, in International Conference on Learning Representations. 2019.
4. C. Xie and A. Yuille, *Intriguing Properties of Adversarial Training at Scale*, in International Conference on Learning Representations, 2019.
5. Z. Charles, H. Rosenberg, and D. Papailiopoulos, *A Geometric Perspective on the Transferability of Adversarial Directions*, in International Conference on Artificial Intelligence and Statistics, 2019.

Challenges in DL/ML applications: Robustness (References IV)

1. A. Fawzi, H. Fawzi, and O. Fawzi, *Adversarial vulnerability for any classifier*, in Advances in Neural Information Processing Systems, 2018.
2. Raghunathan, A., Steinhardt, J., and Liang, P. S. *Semidefinite relaxations for certifying robustness to adversarial examples*, in Advances in Neural Information Processing Systems, 2018.
3. L. Schmidt, S. Santurkar, D. Tsipras, K. Talwar, and A. Ma, *Adversarially Robust Generalization Requires More Data*, in Advances in Neural Information Processing Systems, 2018.
4. Wong, E. and Kolter, Z. (2018). *Provable defenses against adversarial examples via the convex outer adversarial polytope*, in International Conference on Machine Learning, 2018.
5. Athalye, A., et al. *Synthesizing robust adversarial examples*, in International Conference on Machine Learning, 2018.
6. Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., and Song, D. *Robust physical-world attacks on deep learning visual classification*, in IEEE Conference on Computer Vision and Pattern Recognition, 2018.
7. J.-H. Jacobsen, J. Behrmann, R. Zemel, and M. Bethge, *Excessive Invariance Causes Adversarial Vulnerability*, in International Conference on Learning Representations, 2018.
8. Madry, Aleksander and Makelov, Aleksandar and Schmidt, Ludwig and Tsipras, Dimitris and Vladu, Adrian. *Towards Deep Learning Models Resistant to Adversarial Attacks*, in International Conference on Learning Representations, 2018.
9. Huang, X., Kwiatkowska, M., Wang, S., and Wu, M. *Safety verification of deep neural networks*, in Computer Aided Verification 2017.

Challenges in DL/ML applications: Surveillance/Privacy/Manipulation



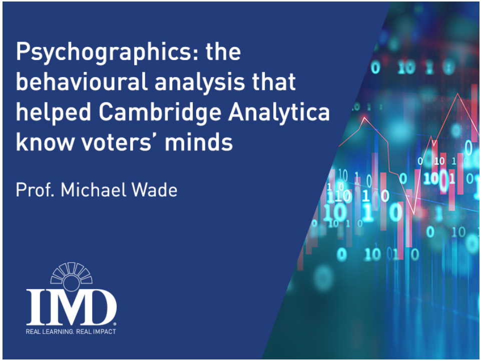
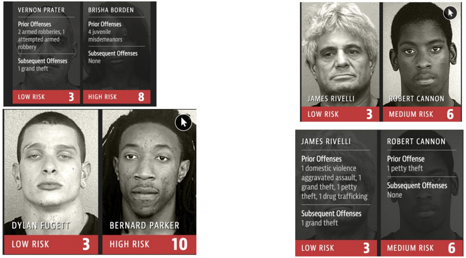


Figure: Political and societal concerns about some DL/ML applications

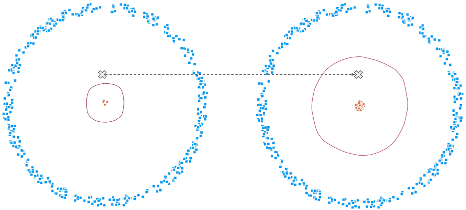
Challenges in DL/ML applications: Surveillance/Privacy/Manipulation (References)

1. Ganev Georgi et al. *Robin Hood and Matthew Effects: Differential Privacy Has Disparate Impact on Synthetic Data*, in International Conference on Machine Learning, 2022.
2. Xu, D., Du, W., Wu, X. *Removing disparate impact on model accuracy in differentially private stochastic gradient descent*, in Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2021.
3. Tran, C., Fioretto, F., Van Hentenryck, P. *Differentially private and fair deep learning: A lagrangian dual approach*, arXiv:2009.12562, 2020.
4. Pujol, D., McKenna, R., Kuppam, S., Hay, M., Machanavajjhala, A., Miklau, G. (textitFair decision making using privacy-protected data , in Proceedings of Fairness, Accountability, and Transparency (FAT), 2020.
5. Bu, Z., Dong, J., Long, Q., Su, W. J. *Deep learning with Gaussian differential privacy*, Harvard data science review, 2020.
6. Bagdasaryan, E., Poursaeed, O., Shmatikov, V. *Differential privacy has disparate impact on model accuracy*, in Neural Information Processing Systems (NeurIPS), 2019.
7. Sreenu, G., Saleem Durai, M.A. *Intelligent video surveillance: a review through deep learning techniques for crowd analysis*, J Big Data 6, 48. 2019.
8. Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. *Deep learning with differential privacy*, in Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, 2016.
9. O'Neil, C., *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*, Broadway Books, 2016.
10. Dwork, C., and Roth, A. *The Algorithmic Foundations of Differential Privacy*, in Foundations and Trends in Theoretical Computer Science, 9, 2013.

Challenges in DL/ML applications: Fairness



(a) Racist classifier



(b) Effect of unbalanced data

Figure: Unfair classifiers due to biased or unbalanced datasets/algorithms

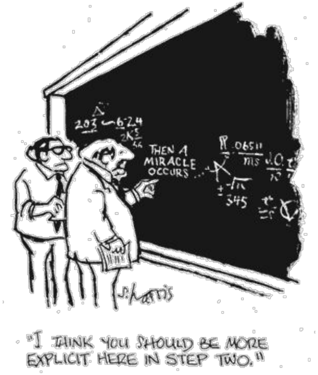
Challenges in DL/ML applications: Fairness (References I)

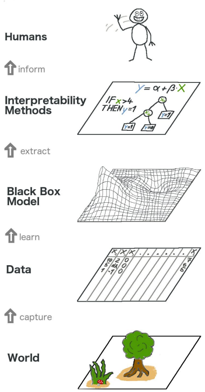
1. Nilforoshan Hamed et al. *Causal Conceptions of Fairness and their Consequences*, in International Conference on Machine Learning (ICML), 2022.
2. Jacobs, A. Z., Wallach, H. *Measurement and fairness*, in Proceedings of Fairness, Accountability, and Transparency (FAT), 2021.
3. Ramaswamy, V. V., Kim, S. S., Russakovsky, O. *Fair attribute classification through latent space de-biasing*, in Proceedings of the Conference on Computer Vision and Pattern Recognition, 2021.
4. Jalal, A., Karmalkar, S., Hoffmann, J., Dimakis, A., Price, E. *Fairness for Image Generation with Uncertain Sensitive Attributes*, in International Conference on Machine Learning (ICML), 2021.
5. Rolf, E., Simchowitz, M., Dean, S., Liu, L. T., Bjorkegren, D., Hardt, M., Blumenstock, J. *Balancing competing objectives with noisy data: Score-based classifiers for welfare-aware machine learning*, in International Conference on Machine Learning (ICML), 2021.
6. Wang, A., Russakovsky, O. *Directional bias amplification*, in International Conference on Machine Learning (ICML), 2021.
7. Jia, S., Meng, T., Zhao, J., Chang, K. W. *Mitigating gender bias amplification in distribution by posterior regularization*, in Annual Meeting of the Association for Computational Linguistics (ACL), 2020.
8. Mitchell, M., Baker, D., Moorosi, N., Denton, E., Hutchinson, B., Hanna, A., Morgenstern, J. *Diversity and inclusion metrics in subset selection*, in Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, 2020.
9. Yang, K., Qinami, K., Fei-Fei, L., Deng, J., Russakovsky, O. *Towards fairer datasets: Filtering and balancing the distribution of the people subtree in the imagenet hierarchy*, in Proceedings of Fairness, Accountability, and Transparency (FAT), 2020.
10. Hanna, A., Denton, E., Smart, A., Smith-Loud, J. *Towards a critical race methodology in algorithmic fairness*, in Proceedings of Fairness, Accountability, and Transparency (FAT), 2020.

Challenges in DL/ML applications: Fairness (References II)

1. Barocas, S. Hardt, M. Narayanan, Arvind. *Fairness in Machine Learning Limitations and Opportunities*. `https://fairmlbook.org/pdf/fairmlbook.pdf`, 2020.
2. Obermeyer, Z., Powers, B., Vogeli, C., Mullainathan, S. *Dissecting racial bias in an algorithm used to manage the health of populations*. Science, 2019.
3. Hoyle, A., Wallach, H., Augenstein, I., Cotterell, R. *Unsupervised discovery of gendered language through latent-variable modeling*, arXiv:1906.04760, 2019.
4. Noble, S.U. *Algorithms of Oppression: How Search Engines Reinforce Racism*, NYU Press, 2018.
5. Campolo, A., Sanfilippo, M., Whittaker, M., Crawford, K. *AI Now 2017 Report*. AI Now Institute at New York University, 2017.
6. Munoz, C., Smith, M., and Patil, D. *Big Data: A Report on Algorithmic Systems, Opportunity, and Civil Rights*. Executive Office of the President. The White House, 2016.
7. Hardt, M. *How Big Data Is Unfair*. `https://medium.com/@mrtz/how-big-data-is-unfair-9aa544d739de` 2014.
8. Pedreshi, D., Ruggieri, S. and Turini, F. *Discrimination-Aware Data Mining*. Proc. 14th SIGKDD. ACM 2008.
9. Friedman, B. and Nissenbaum, H. *Bias in Computer Systems*. ACM Transactions on Information Systems (TOIS), 1996.

Challenges in DL/ML applications: Interpretability





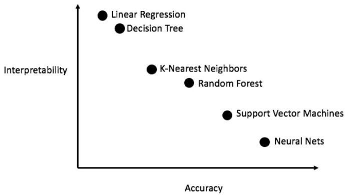


Figure: Performance vs Interpretability trade-offs in DL/ML

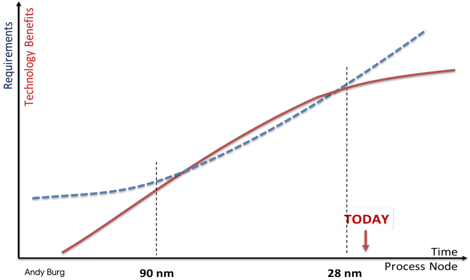
Challenges in DL/ML applications: Interpretability (References I)

1. Yin Kayo et al *Interpreting Language Models with Contrastive Explanations*, arXiv:2202.10419, 2022.
2. C. Rudin, C. Chen, Z. Chen, H. Huang, L. Semenova, and C. Zhong, *Interpretable Machine Learning: Fundamental Principles and 10 Grand Challenges*, arXiv:2103.11251, 2021.
3. F. Poursabzi-Sangdeh, D. G. Goldstein, J. M. Hofman, J. W. Wortman Vaughan, and H. Wallach, *Manipulating and Measuring Model Interpretability*, in Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems, New York, NY, USA: Association for Computing Machinery, 2021, pp. 1–52.
4. J. Lee, S. Park, and J. Shin, *Learning Bounds for Risk-sensitive Learning*, arXiv:2006.08138. 2020.
5. H. Kaur, H. Nori, S. Jenkins, R. Caruana, H. Wallach, and J. Wortman Vaughan, *Interpreting Interpretability: Understanding Data Scientists' Use of Interpretability Tools for Machine Learning*, in Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems, 2020.
6. P. Hase and M. Bansal, *Evaluating Explainable AI: Which Algorithmic Explanations Help Users Predict Model Behavior?*, in Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020.
7. L. Semenova, C. Rudin, and R. Parr, *A study in Rashomon curves and volumes: A new perspective on generalization and model simplicity in machine learning*, arXiv:1908.01755, 2019.
8. C. Rudin and J. Radin, *Why Are We Using Black Box Models in AI When We Don't Need To? A Lesson From An Explainable AI Competition*, in Harvard Data Science Review, 2019.

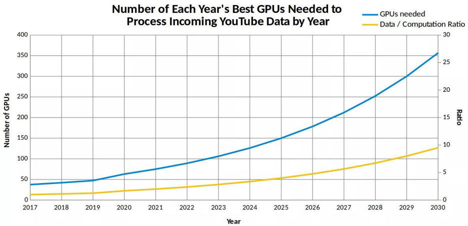
Challenges in DL/ML applications: Interpretability (References II)

1. C. Rudin, *Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead*, in Nat Mach Intell, 2019.
2. S. Hooker, D. Erhan, P.-J. Kindermans, and B. Kim, *A Benchmark for Interpretability Methods in Deep Neural Networks*, 2018.
3. S. Barocas, M. Hardt, and A. Narayanan, *Fairness in machine learning*, in Nips tutorial, 2017.
4. Sundararajan, Mukund and Taly, Ankur and Yan, Qiqi. *Axiomatic Attribution for Deep Networks*, in International Conference on Machine Learning (ICML), 2017.
5. Shrikumar, Avanti and Greenside, Peyton and Kundaje, Anshul. *Learning Important Features Through Propagating Activation Differences*, in International Conference on Machine Learning (ICML), 2017.
6. Ribeiro, Marco and Singh, Sameer and Guestrin, Carlos. *Why Should I Trust You?": Explaining the Predictions of Any Classifier.*, KDD 2016.
7. *Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps*, arXiv:1312.6034, 2013.
8. Baehrens, David and Schroeter, Timon and Harmeling, Stefan and Kawanabe, Motoaki and Hansen, Katja and Mueller, Klaus-Robert. Simonyan, Karen and Vedaldi, Andrea and Zisserman, Andrew. *How to Explain Individual Classification Decisions*, JMLR, 2010.

Challenges in DL/ML applications: Energy efficiency and cost



(a)



(b)

Figure: Efficiency and Scalability concerns in DL/ML

Challenges in DL/ML applications: Energy efficiency and cost (References)

1. Gutiérrez María et al. *Analysing the energy impact of different optimisations for machine learning models*, in International Conference on ICT for Sustainability (ICT4S), 2022.
2. E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell, *On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?*, in Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency, 2021.
3. J. Launay et al. *Hardware Beyond Backpropagation: a Photonic Co-Processor for Direct Feedback Alignment*, arXiv:2012.06373, 2020.
4. J. Launay, I. Poli, F. Boniface, and F. Krzakala, *Direct Feedback Alignment Scales to Modern Deep Learning Tasks and Architectures*, in Advances in Neural Information Processing Systems (NeurIPS), 2020.
5. Goel, A., Tung, C., Lu, Y. H., and Thiruvathukal, G. K. *A Survey of Methods for Low-Power Deep Learning and Computer Vision*, arXiv preprint arXiv:2003.11066, 2020.
6. Conti, F., Rusci, M., and Benini, L. *The Memory Challenge in Ultra-Low Power Deep Learning*, in NANO-CHIPS 2030, 2020.
7. García-Marín, E., Rodrigues, C. F., Riley, G., and Grahn, H. *Estimation of energy consumption in machine learning*, in Journal of Parallel and Distributed Computing, 2019.
8. Strubell, E., Ganesh, A., and McCallum, A. *Energy and policy considerations for deep learning in NLP*, arXiv preprint arXiv:1906.02243, 2019.

Wrap up!

- Learning deep continues!

References I

[1] Zeyuan Allen-Zhu.
Natasha: Faster non-convex stochastic optimization via strongly non-convex parameter.
In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 89–97. PMLR, 06–11 Aug 2017.
(Cited on page 21.)

[2] Zeyuan Allen-Zhu.
Katyusha x: Practical momentum method for stochastic sum-of-nonconvex optimization.
In *ICML*, 2018.
(Cited on page 20.)

[3] Zeyuan Allen-Zhu.
Katyusha x: Simple momentum method for stochastic sum-of-nonconvex optimization.
In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 179–185. PMLR, 10–15 Jul 2018.
(Cited on page 17.)

[4] Anish Athalye, Nicholas Carlini, and David Wagner.
Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples.
In *International Conference on Machine Learning*, pages 274–283. PMLR, 2018.
(Cited on page 47.)

References II

[5] George Cybenko.
Approximation by superpositions of a sigmoidal function.
Mathematics of control, signals and systems, 2(4):303–314, 1989.
(Cited on page 34.)

[6] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien.
Saga: A fast incremental gradient method with support for non-strongly convex composite objectives.
In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1646–1654. Curran Associates, Inc., 2014.
(Cited on pages 8, 9, and 14.)

[7] Benjamin Dubois-Taine, Sharan Vaswani, Reza Babanezhad, Mark Schmidt, and Simon Lacoste-Julien.
Svrg meets adagrad: Painless variance reduction, 2021.
(Cited on page 20.)

[8] John Duchi, Elad Hazan, and Yoram Singer.
Adaptive subgradient methods for online learning and stochastic optimization.
J. Mach. Learn. Res., 12:2121–2159, July 2011.
(Cited on page 20.)

References III

[9] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song.
Robust physical-world attacks on deep learning visual classification.
In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1625–1634, 2018.
(Cited on page 47.)

[10] Cong Fang, Chris Junchi Li, Zhouchen Lin, and Tong Zhang.
SPIDER: near-optimal non-convex optimization via stochastic path-integrated differential estimator.
In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 687–697, 2018.
(Cited on pages 17, 18, 20, and 21.)

[11] Rie Johnson and Tong Zhang.
Accelerating stochastic gradient descent using predictive variance reduction.
In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 315–323. Curran Associates, Inc., 2013.
(Cited on page 11.)

References IV

[12] Andrej Karpathy.
What I learned from competing against a ConvNet on ImageNet.
(Cited on page 39.)

[13] Ali Kavis, Stratis Skoulakis, Kimon Antonakopoulos, Leello Tadesse Dadi, and Volkan Cevher.
Adaptive stochastic variance reduction for non-convex finite-sum minimization.
Advances In Neural Information Processing Systems 36 (NeurIPS 2022), (CONF), 2022.
(Cited on pages 19 and 21.)

[14] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein.
Visualizing the Loss Landscape of Neural Nets.
In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.
(Cited on page 44.)

[15] Zhize Li, Hongyan Bao, Xiangliang Zhang, and Peter Richtarik.
Page: A simple and optimal probabilistic gradient estimator for nonconvex optimization.
In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 6286–6295. PMLR, 18–24 Jul 2021.
(Cited on page 17.)

References V

[16] Sashank J Reddi, Suvrit Sra, Barnabás Póczos, and Alex Smola.
Stochastic frank-wolfe methods for nonconvex optimization.
arXiv preprint arXiv:1607.08254, 2016.
(Cited on page 17.)

[17] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al.
Imagenet large scale visual recognition challenge.
115(3):211–252, 2015.
(Cited on page 39.)

[18] Ali Shafahi, W Ronny Huang, Christoph Studer, Soheil Feizi, and Tom Goldstein.
Are adversarial examples inevitable?
In *7th International Conference on Learning Representations (ICLR 2019)*, 2019.
(Cited on page 48.)

[19] Zhe Wang, Kaiyi Ji, Yi Zhou, Yingbin Liang, and Vahid Tarokh.
Spiderboost and momentum: Faster stochastic variance reduction algorithms.
In *Advances in Neural Information Processing Systems*, 2019.
(Cited on page 17.)

References VI

[20] Zhe Wang, Kaiyi Ji, Yi Zhou, Yingbin Liang, and Vahid Tarokh.
Spiderboost and momentum: Faster variance reduction algorithms.
In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alche Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.
(Cited on page 20.)

[21] Lin Xiao and Tong Zhang.
A proximal stochastic gradient method with progressive variance reduction.
SIAM Journal on Optimization, 24, 03 2014.
(Cited on page 11.)

[22] Guangzeng Xie, Luo Luo, and Zhihua Zhang.
A general analysis framework of lower complexity bounds for finite-sum optimization, 2019.
(Cited on page 17.)

[23] Dongruo Zhou and Quanquan Gu.
Lower bounds for smooth nonconvex finite-sum optimization.
In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7574–7583. PMLR, 09–15 Jun 2019.
(Cited on pages 17 and 21.)