Modern Techniques of Statistical Optimization for Machine Learning

Tong Zhang

Our Current Approach to AI

Simple Environment: Big Data+Big Computing+Deep Learning=AI

ImageNet Challenge

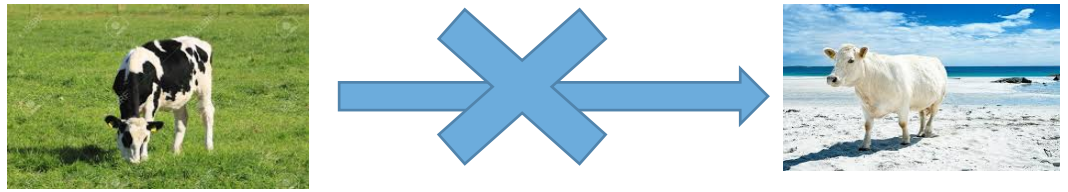
IMAGENET

- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.

Complex Environment & Small Data Learning

- Robustness

- Varying Environment



Research Theme 1: Big Data & System Machine Learning

Big Data + Big Computing

- Feature engineering
- Data augmentation

- Model specification
- Nonconvex optimization
- Distributed optimization

- Model compression
- On device learning

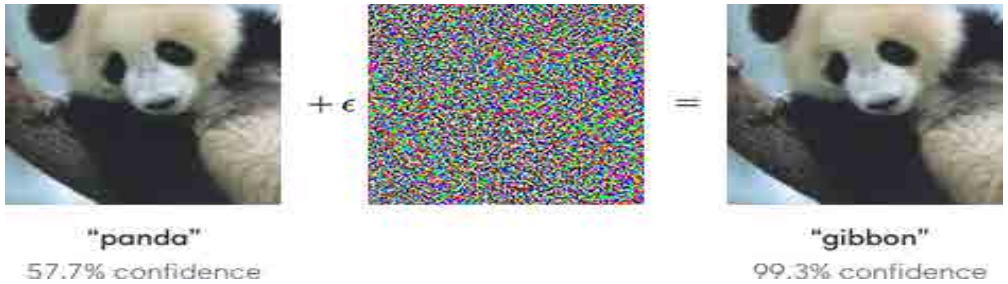
Automated Machine Learning (autoML)
feature engineering, hyperparameter tuning, architecture search …

Machine Learning Platform

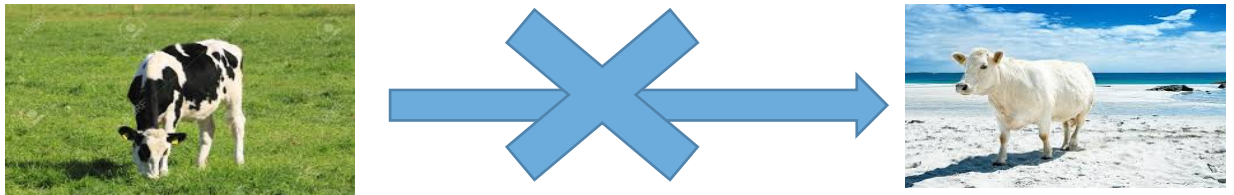
Research Theme 2: Robust and Adaptive Algorithms

Small Data + Complex Environment

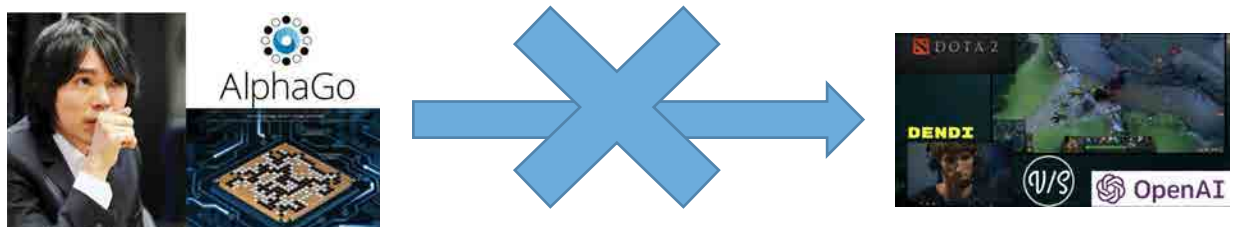
- Robustness



- Varying Environment



- Expansion to New Tasks



Adversarial Examples; Causality; Meta Learning; Continual Learning; Few-shots Learning

Some Statistical Optimization Related Recent Papers (and Techniques)

Big Data and System ML

Theory of Nonconvex Stochastic Optimization (variance reduction)

- Sharp Analysis for Nonconvex SGD Escaping from Saddle Points (COLT 2019)
- Near-Optimal Non-Convex Optimization via Stochastic Path-Integrated Differential Estimator (NIPS 2018)

Distributed Training (gradient compression)

- Gradient Sparsification for Communication-Efficient Distributed Optimization (NIPS 2018)
- Error Compensated Quantized SGD and its Applications to Large-scale Distributed Optimization (ICML 18)
- DoubleSqueeze: Parallel Stochastic Gradient Descent with Double-pass Error-Compensated Compression (ICML 19)

Robust and Adaptive Algorithms

Adversarial Attack (derivative free optimization)

- Efficient Decision-based Black-box Adversarial Attacks on Face Recognition (CVPR 2019)
- NATTACK: Improved Black-Box Adversarial Attack with Normal Distributions (ICML 2019)
- Hessian-Aware Zeroth-Order Optimization for Black-Box Adversarial Attack (arxiv)

Problem 1/3: Nonconvex Stochastic Optimization

Challenges: nonconvexity

Technique: variance reduction

Papers:

Sharp Analysis for Nonconvex SGD Escaping from Saddle Points (COLT 2019)

Near-Optimal Non-Convex Optimization via Stochastic Path-Integrated Differential Estimator (NIPS 18)

Mathematical Formulation

Training machine learning models

$$\min_x f(x) \qquad f(x) := \frac{1}{n}\sum_{i=1}^{n} f_i(x)$$

SGD algorithm:

for t=1, 2, …

randomly pick i from 1 to n:

$$x_t \leftarrow x_{t-1} - \eta_t \nabla f_i(x_{t-1})$$

In practice it works well for nonconvex functions such as neural networks

Simple Theoretical Question

Question: How many SGD steps are needed to find an approximate stationary point x

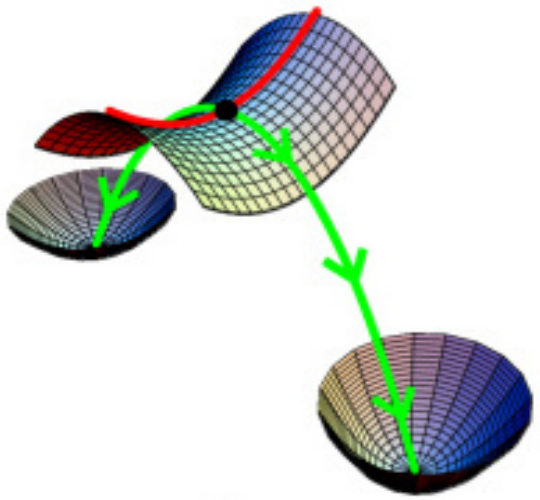
$$\|\nabla f(x)\|_2 \leq \epsilon$$

Partial answer: not difficult to show (using standard proof techniques)

$O\left(\epsilon^{-4}\right)$ number of steps

More Complex Theoretical Questions

Q 1
Does SGD escape saddle and
converge to an approximate local minimum?



Q2
How fast does it converge to local minimum?

Q3
Is there a better method?

Saddle Points and Approximate Local Minimum

Given $\epsilon > 0$, an ϵ-approximate local minimum is a point x such that

$$\|\nabla f(x)\|_2 \leq \epsilon$$

$$\nabla^2 f(x) \geq -\sqrt{\epsilon} I$$

Claim: SGD can escape saddle points and converge to an approximate local minimum

Number of iterations for SGD:

(Ge et al., 2015)	$\mathrm{poly}(d)\epsilon^{-8}$
(Daneshmand et al., 2018)	$d\epsilon^{-10}$
(Jin et al, arxiv 2019)	ϵ^{-4}
(Our result COLT 2019)	$\epsilon^{-3.5}$

Improving SGD: earlier attempts

Our new result shows SGD can converge to approximate local minimum in $\epsilon^{-3.5}$ steps.

Question: Can we do better?

Earlier attempts to improve the convergence of SGD: most are not better

SGD Variants	NEON+SGD	(Xu et al., 2018)	ϵ^{-4}
	NEON2+SGD	(Allen-Zhu & Li, 2018)	
	Stochastic Cubic	(Tripuraneni et al., 2018)	$\epsilon^{-3.5}$
	RSGD5	(Allen-Zhu, 2018a)	
	Natasha2$^{\Delta}$	(Allen-Zhu, 2018b)	$\epsilon^{-3.5}$
	NEON2+SNVRG$^{\Theta}$	(Zhou et al., 2018a)	
	SPIDER-SFO^{+}	(Fang et al., 2018)	ϵ^{-3}

Improving SGD: SPIDER

It can be shown best possible convergence result is: ϵ^{-3}

Can be achieved by SPIDER algorithm, using variance reduction technique

Proved in our NIPS 2018 paper [Fang, Li, Lin, Zhang]
Near-Optimal Non-Convex Optimization via Stochastic Path-Integrated Differential Estimator

Stochastic Path Integrated Differential Estimator

Given stochastic path $x_1, x_2, \ldots, x_t, \ldots$ estimate gradient $\nabla f(x_t)$

SG (Stochastic Gradient):

$$g_t = \nabla f_i(x_t) \quad \text{Unbiased estimator of} \quad \nabla f(x_t)$$

SPIDER:

$$\Delta g_t = \nabla f_i(x_t) - \nabla f_i(x_{t-1}) \quad \text{Unbiased estimator of} \quad \nabla f(x_t) - \nabla f(x_{t-1})$$

$$\nabla f(x_0) + \sum_{s=1}^{t} \Delta g_s \quad \text{Unbiased estimator of} \quad \nabla f(x_t)$$

Under Suitable Conditions, SPIDER is a better estimator than SG with smaller variance

The Benefit of SPIDER (Variance Reduction)

Variance of SGD: $\mathbf{E}_i \|f_i(x_t) - \nabla f(x_t)\|^2 = O(1)$

Variance of SPIDER: $O(\eta)$ with $t = O(1/\eta)$

Assume constant learning rate

$\|x_s - x_{s-1}\| = O(\eta)$

$\|\Delta g_s\| = \|\nabla f_i(x_s) - \nabla f_i(x_{s-1})\| = O(\eta)$

$\|\nabla f(x_s) - \nabla f(x_{s-1})\| = O(\eta)$

$$\mathbf{E} \| \underbrace{\sum_{s=1}^{t} (\Delta g_s - (\nabla f(x_s) - \nabla f(x_{s-1}))}_{\text{sum of martingale difference}} \|^2$$

$$= \sum_{s=1}^{t} \mathbf{E} \| \underbrace{(\Delta g_s - (\nabla f(x_s) - \nabla f(x_{s-1}))}_{O(\eta)} \|^2$$

$$= \mathbf{E} \sum_{s=1}^{t} O(\eta^2) = O(\eta).$$

The SPIDER Algorithm

Algorithm 2 SPIDER-SFO: Input x^0, $\mathcal{S}_1 = 2\sigma^2/\epsilon^2$, $q = 2\sigma^2/\epsilon^2$ (online case, in high probability)

for $k = 0$ to $K-1$ **do**
  if $\mod(k, q) = 0$ **then**
    Draw $\mathcal{S}_1$ samples and let $v^k = \nabla f_{S_1}(x^k)$
  else
    Draw $i \sim [n]$ uniformly at random, and let $v^k = \nabla f_i(x^k) - \nabla f_i(x^{k-1}) + v^{k-1}$
  end if
  if $\|v^k\| \le 2\epsilon$ **then**
    return x^k
  else
    $x^{k+1} = x^k - \eta \cdot (v^k/\|v^k\|)$ where $\eta = \epsilon^2/(2L\sigma)$.
  end if
end for
return x^K ◇ however, this line is *not* reached with high probability

Optimality of SPIDER Algorithm

Upper Bound

SPIDER can obtain an ϵ approximate local minimum in

$$O(\epsilon^{-3}) \qquad \text{steps}$$

Lower Bound

There exists a problem such that no algorithm can obtain an ϵ approximate local minimum in fewer than

$$\Omega(\epsilon^{-3}) \qquad \text{steps}$$

Summary

Answered a fundamental question on the complexity of nonconvex optimization

1. The complexity of SGD

2. The optimal complexity

Practical for some problems, but not for neural networks due to special structures

What's next?
Work in progress: improved practical training algorithm for neural network training

Problem 2/3: Distributed Training

Challenges: Network Communication Bandwidth

New Technique: gradient compression + error compensation

Papers:

Gradient Sparsification for Communication-Efficient Distributed Optimization (NIPS18)

DoubleSqueeze: Parallel Stochastic Gradient Descent with Double-pass Error-Compensated Compression (ICML 19)

Error Compensated Quantized SGD and its Applications to Large-scale Distributed Optimization (ICML 18)

Optimize Communication for Distributed Computing

Problem: training machine learning models

$$\min_{x} \quad \frac{1}{n}\sum_{i=1}^{n} f_i(x)$$

Data are distributed over multiple nodes

Question: How to reduce the communication cost?

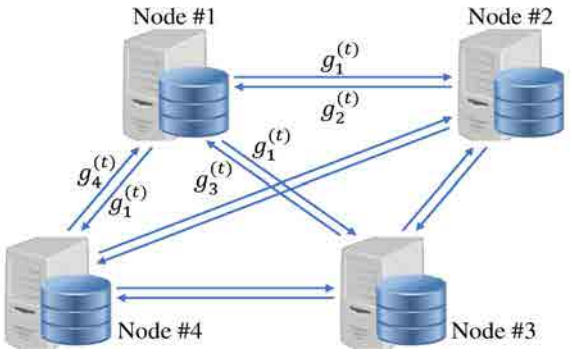
Answer:

Error Compensated Quantized SGD and its Applications to Large-scale Distributed Optimization （ICML 2018)

DoubleSqueeze: Parallel Stochastic Gradient Descent with Double-pass Error-Compensated Compression (ICML 19)

Other Communication Architectures

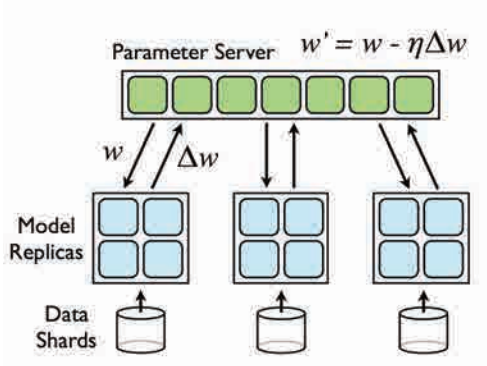
Simple One-Step Communication



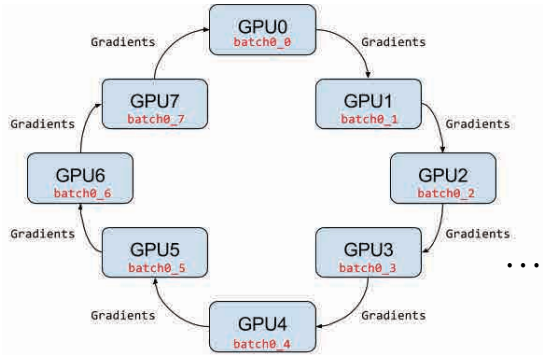
Fully Connected

Error Compensated Quantized SGD and its Applications to Large-scale Distributed Optimization (ICML 18)

Complex Multi-Step Communication



Parameter Server



Ring All Reduce ...

DoubleSqueeze: Parallel Stochastic Gradient Descent with Double-pass Error-Compensated Compression (ICML 19)

Optimization Method: SGD

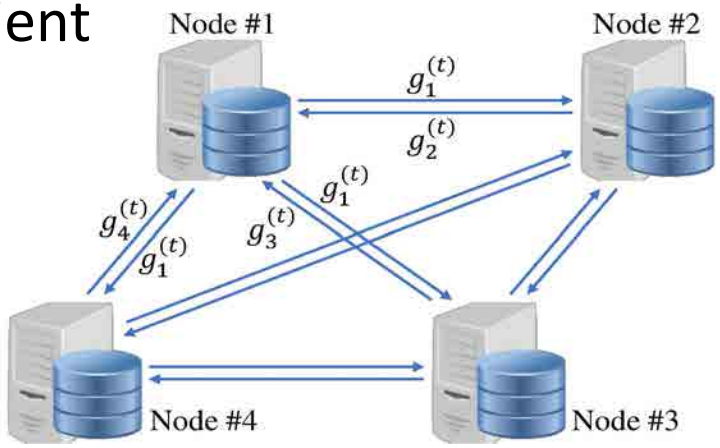
time step t model parameter $x^{(t)}$

Compute minibatch gradient using local data on each node

$$g_i^{(t)}(x^{(t)}) = \frac{1}{b}\sum_{j=1}^{b}\nabla f_j(x^{(t)})$$

Communicate across network and compute aggregated gradient

$$g^{(t)} = \frac{1}{m}\sum_{i=1}^{m} g_i^{(t)}(x^{(t)})$$



Communicate and update model parameter:

$$x^{(t+1)} = x^{(t)} - \eta g^{(t)}$$

Existing Algorithm: QSGD

Compute local gradient on each node $g^{(t)}$

Quantize $\tilde{g}^{(t)} = \mathrm{Quantize}(g^{(t)})$

Broadcast to other nodes $\tilde{g}^{(t)}$

Advantage: quantization reduces communication cost

Disadvantage: quantization error causes convergence slow down

How to Improve

Ideal iterate: $x^{(t)} = x^{(0)} - \eta[g^{(0)} + g^{(1)} + \cdots + g^{(t-1)}]$

Latest gradient: $g^{(t)}$

Transmitted gradient: $\tilde{g}^{(t)}$

One step error due to compression: untransmitted gradient $g^{(t)} - \tilde{g}^{(t)}$

Cummulative error: cumulative untransmitted gradient

$$h^{(t+1)} = h^{(t)} + \underbrace{(g^{(t)} - \tilde{g}^{(t)})}_{\text{untransmitted gradient}}$$

Key Idea: error compensation

- Use cumulative untramsmitted gradients to compensate the current gradient
- Effect is similar to delayed gradients in asynchronous distributed optimization

Our Approach: Error Compensated Quantized SGD

Compute local gradient on each node $g^{(t)}$

Quantize $\tilde{g}^{(t)} = \mathrm{Quantize}(g^{(t)} + h^{(t)})$

Broadcast to other nodes $\tilde{g}^{(t)}$

Update on each node $h^{(t+1)} = h^{(t)} + \underbrace{(g^{(t)} - \tilde{g}^{(t)})}_{\mathrm{error}}$

Theory of Error Compensation

Claim 1:
QSGD has slower convergence rate than that of SGD

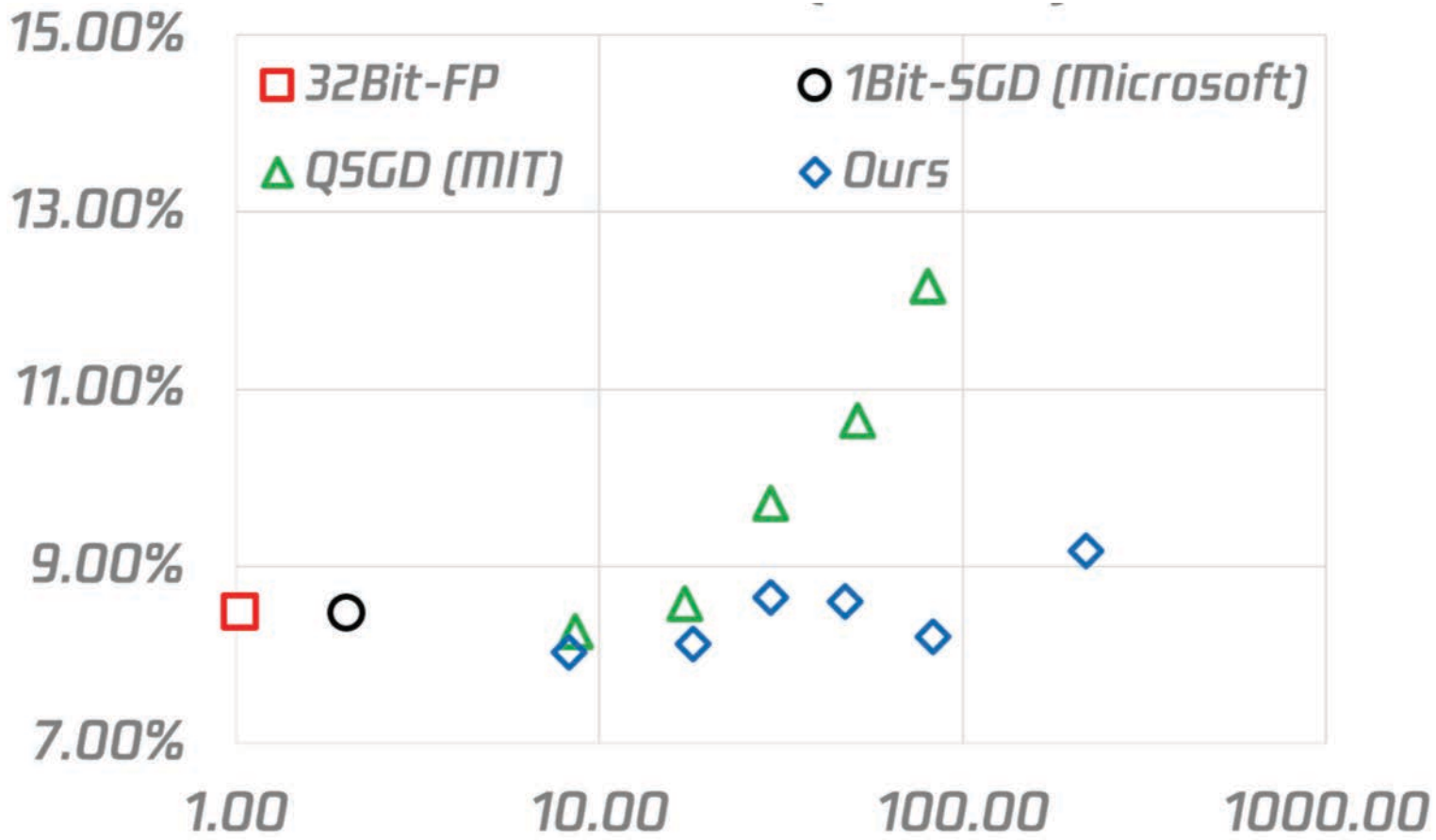
Claim 2:
Asymptotically EC-QSGD has the same convergence rate as that of SGD up to the leading order, under appropriate conditions, and in particular:

$$\|\mathrm{Quantize}(u) - u\| \leq \gamma \|u\| \quad \text{with } \gamma < 1.$$

Conclusion:
EC-QSGD behaves significantly better than QSGD

Numerical Experiments

Classification Error



Compression Ratio

Summary

New Technique for Distributed Optimization under Limited Network Bandwidth

What's Next?

Other System-Aware Machine Learning Training and Deployment

- Decentralized On-Device Training
- Cloud-Device Cooperative Training
- Privacy-Aware Cooperative Training
- Low-bit Training
- Automated System Performance Optimization
- Automated Machine Learning
- On-Device Model Compression
- …

Problem 3/3: Adversarial Attack

Challenges: Limited Number of Queries

New Technique: 2nd order derivative free optimization

Papers:

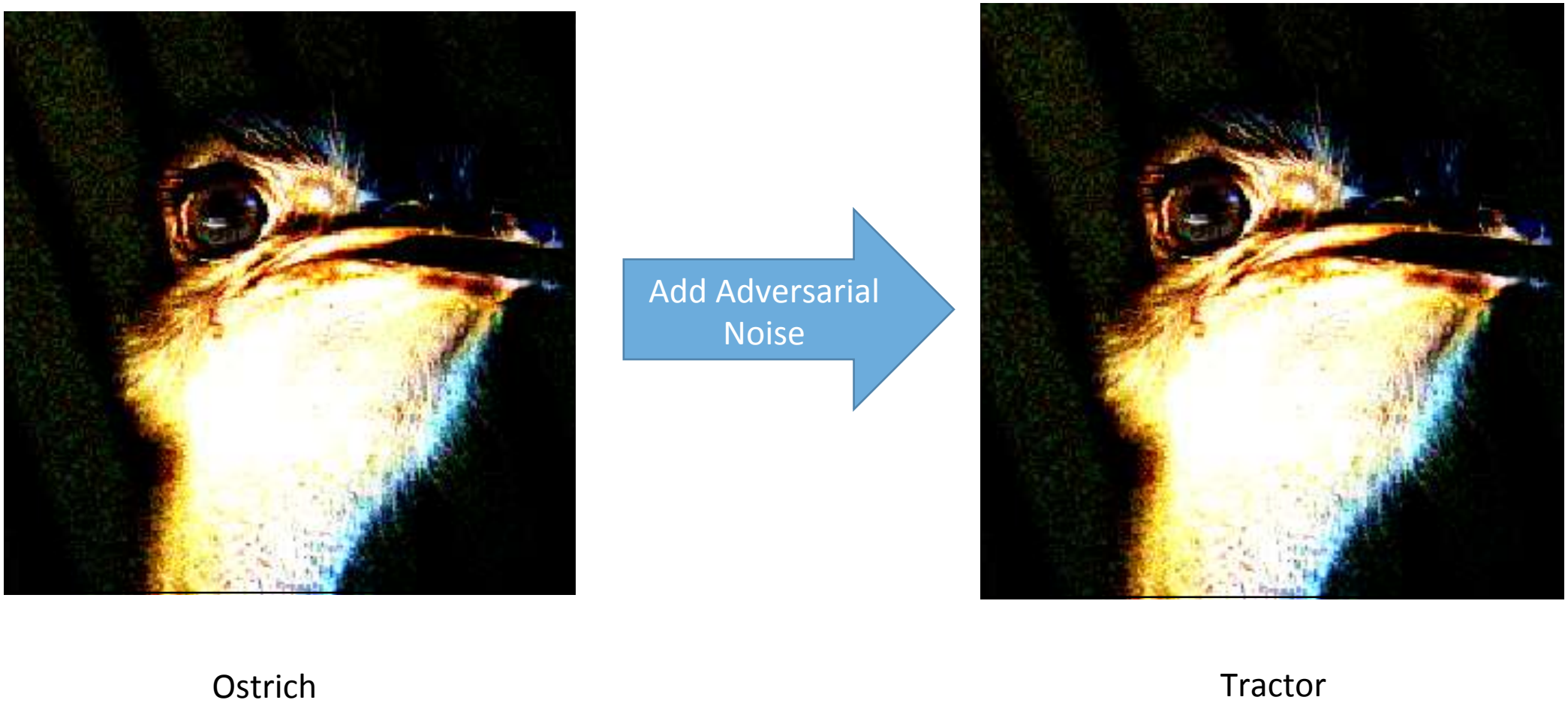
Efficient Decision-based Black-box Adversarial Attacks on Face Recognition (CVPR 19)

NATTACK: Improved Black-Box Adversarial Attack with Normal Distributions (ICML 19)

Hessian-Aware Zeroth-Order Optimization for Black-Box Adversarial Attack (arxiv)

Problem: Deep models are not robust under adversarial attack

- How to construct efficient black-box Adversarial Attack procedure



Ostrich

Tractor

Existing Solutions

- Mathematical Question:
  - Given model $g(\theta, x)$
  - Given example x
  - Find a small perturbation $\delta : \|\delta\| \leq \epsilon$

$$g(\theta, x) \quad \text{and} \quad g(\theta, x + \delta) \quad \text{have different labels}$$

- White Box
  - Need to know the model parameter θ
  - Technique: first order gradients of the model

- Black Box
  - No need to know the model parameter θ : can evaluate function $g(x) = g(\theta, x)$
  - Technique: zeroth order approximation of first order gradients

Black Box Attack Mathematical Problem Statement

Given x_0 with label y_0, want to find $x \approx x_0$ so that

$$\min_{\|x - x_0\| \leq \epsilon} f(x)$$

where

$$f(x) = L(g(x), y_0)$$

is a loss function measuring how different is the model output $g(x)$ and label y_0

Derivative free: We can evaluate $f(x)$ but not $\nabla f(x)$

Black Box Attack using 1st order derivative free algorithm

Approximate derivative using function value

$$\nabla f(x) = \mathbf{E}_{u \sim N(0,I)}\, \mu^{-1}(f(x+\mu u) - f(x))u + O(\mu)$$

The following derivative free algorithm works as approximate SGD

Iterate t=0, 1, …,

Draw $u \sim N(0, I)$

$$x_{t+1} = x_t - (f(x_t + \mu u) - f(x_t))u$$

Our Method: Incorporate Hessian information into black box attack

Algorithm 1 Algorithm `ZO-HessAware`

Input: $x^{(0)}$ is an initial point sufficient close to x^*. And b is the batch size and p is an integer. Parameter η is the step size.
for $t = 0, \ldots, T$ **do**
  if t mod $p == 0$ **then**
    Compute an approximate Hessian $\tilde{H}_t$ satisfies Eqn. (3.1).
  end if
  Generate b samples with $u_i \sim N(0, I_d)$ and construct $\tilde{g}_\mu(x_t) = \frac{1}{b}\sum_{i=1}^{b} \frac{f(x+\tilde{H}_t^{-1/2}u_i)-f(x)}{\mu}\tilde{H}_t^{-1/2}u_i$;
  Update $x_{t+1} = x_t - \eta\tilde{g}_\mu(x_t)$.
end for

At the current time this leads to the most effective attack method

Theory (local convergence)

Assumption: Hessian approximation satisfies: (quality measured by ρ)

$$\rho\tilde{H} \leq \nabla^2 f(x) \leq (2-\rho)\tilde{H}, \qquad \xi I \leq \tilde{H}$$

Convergence Result: If $\xi > 0$ then we have local linear convergence:

$$\mathbf{E}[f(x_{t+1}) - f(x_*)] \leq \left(1 - \frac{b\rho}{16(d+2)}\right)(f(x_t) - f(x_*)) + O(\mu^2)$$

d : problem dimension

Convergence: depends on $\frac{\rho}{d}$ independent of the condition number of $\nabla^2 f(x)$

Derivative Free Hessian Approximation Methods

- Method 1: ZOHA-Gauss

$$\tilde{H} = b^{-1} \sum_{i=1}^{b} \frac{f(x + \mu u_i) + f(x - \mu u_i) - 2f(x)}{2\mu^2} u_i u_i^\top + \lambda I_d, \text{ with } u_i \sim N(0, I_d)$$

- Method 2: ZOHA-Diag

$$\begin{aligned}
\tilde{g}_\mu(x_{t-1}) =& \frac{1}{b} \sum_{i=1}^{b} \frac{f(x_{t-1} + \mu \tilde{u}_i) - f(x_{t-1})}{\mu} \tilde{u}_i, \text{ with } \tilde{u}_i \sim N(0, \tilde{H}_{t-1}^{-1}) \\
D_t =& \nu D_{t-1} + (1 - \nu) \tilde{g}_\mu^2(x_{t-1}) \\
\tilde{H}_t =& \text{diag}\left(\frac{D_t}{1 - \nu^t} \right)
\end{aligned}$$

Attack CNN model on MNIST

	Algorithm	success rate %	median queries	average queries
targeted	`ZOO` (Chen et al., 2017)	42.13	15,200	17,091
	`PGD-NES` (Ilyas et al., 2018)	44.19	7,300	10,496
	`ZOHA-Gauss`	50.03	3,712	6,649
	`ZOHA-Gauss-DC`	**56.14**	**2,941**	**6,246**
	`ZOHA-Diag`	52.13	6,400	9,128
	`ZOHA-Diag-DC`	55.56	3,936	7,239
un-targeted	`ZOO` (Chen et al., 2017)	77.18	13,300	16,390
	`PGD-NES` (Ilyas et al., 2018)	81.55	5,800	8,567
	`ZOHA-Gauss`	85.06	3,612	5,000
	`ZOHA-Gauss-DC`	88.80	**2,152**	**3,629**
	`ZOHA-Diag`	90.37	4,500	6,439
	`ZOHA-Diag-DC`	**91.90**	2,460	4,352

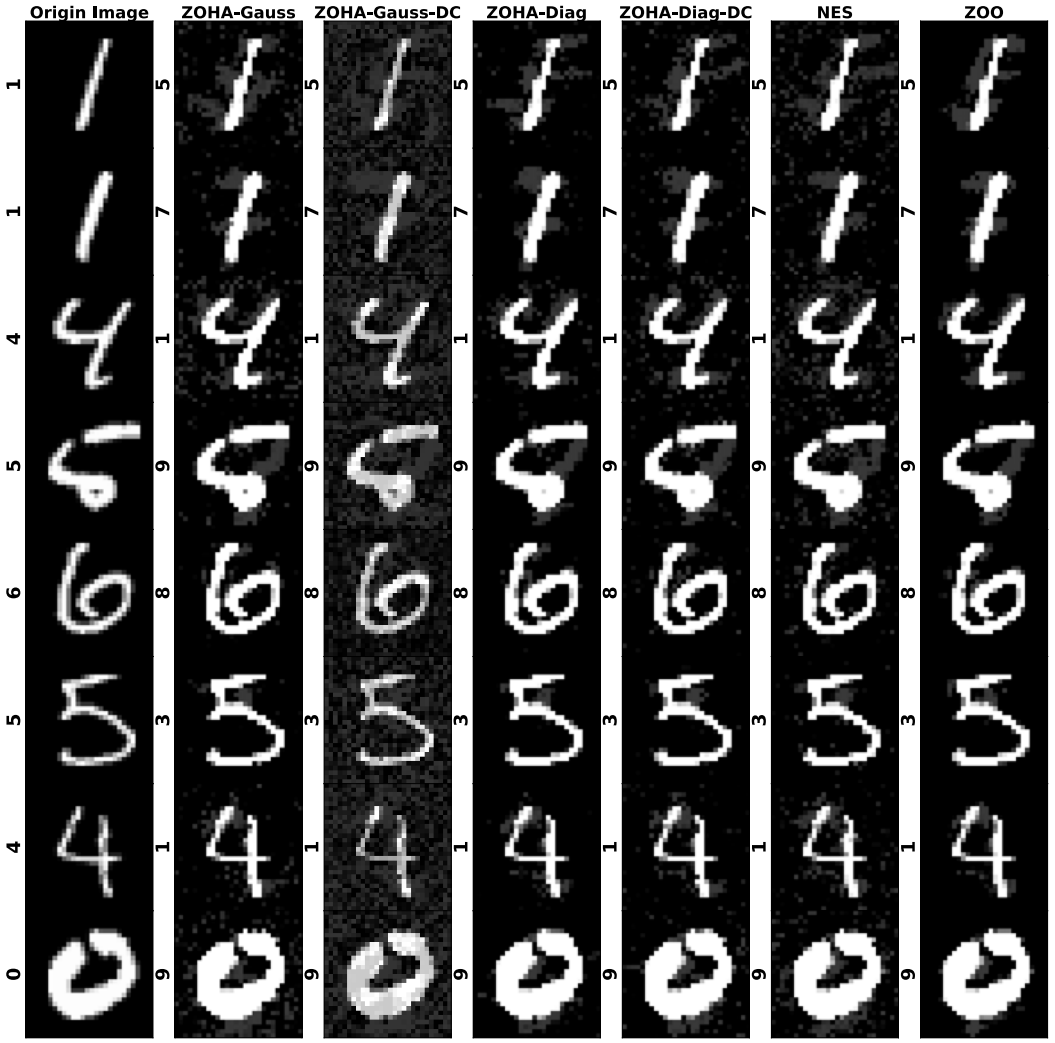
$\|\delta\|_\infty \leq 0.2$

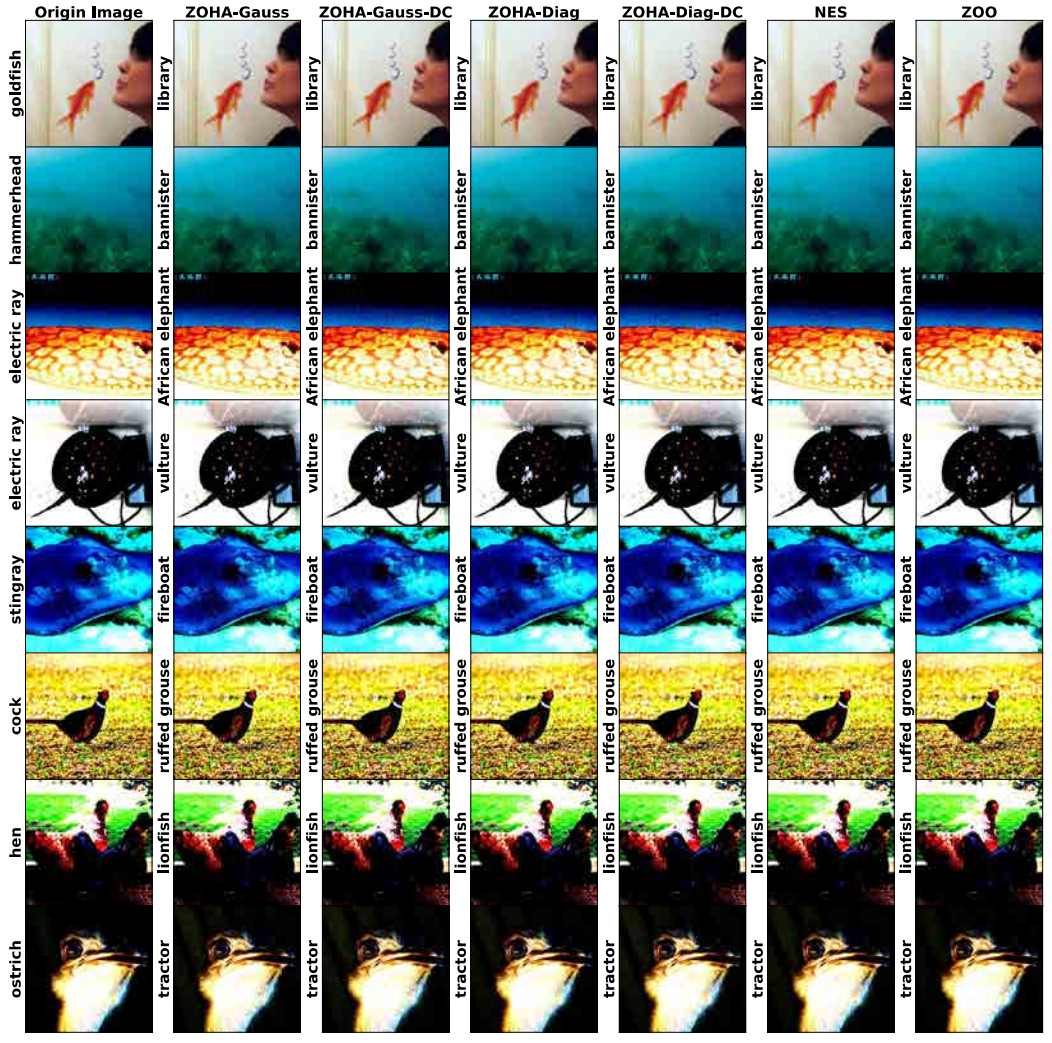
Attack Resnet 50 on Imagenet

	Algorithm	success rate %	median queries	average queries
targeted	ZOO (Chen et al., 2017)	**100**	39,100	45,822
	PGD-NES (Ilyas et al., 2018)	99.37	11,270	17,435
	ZOHA-Gauss	99.62	8,748	12,257
	ZOHA-Gauss-DC	100	8,588	11,770
	ZOHA-Diag	**100**	7,400	9,123
	ZOHA-Diag-DC	**100**	**6,273**	**8,574**
un-targeted	ZOO (Chen et al., 2017)	100	12,700	14,199
	PGD-NES (Ilyas et al., 2018)	100	1,500	2,283
	ZOHA-Gauss	100	1,212	2,259
	ZOHA-Gauss-DC	100	1,124	1,959
	ZOHA-Diag	100	800	1,149
	ZOHA-Diag-DC	100	**561**	**945**

$\|\delta\|_\infty \leq 0.05$

Some Attacked Image Examples





Summary

Statistics and **Optimization** are central to modern machine learning

- Many new problems require new techniques
- Exciting area with rapid research progress

What's next?

- Big Data and System ML
  - Improved algorithm for neural network training
  - Cloud-device collaboration: federated learning
  - Automated machine learning
- Robust and Adaptive Algorithms
  - Attack → Defense → Understand/Prevent
  - Causal learning
  - Meta learning and small sample learning
- Generation Models
- Reinforcement Learning

Thank you