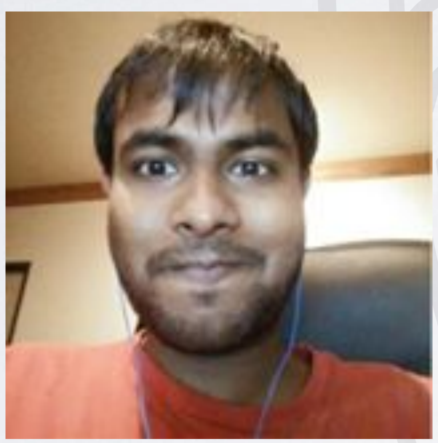
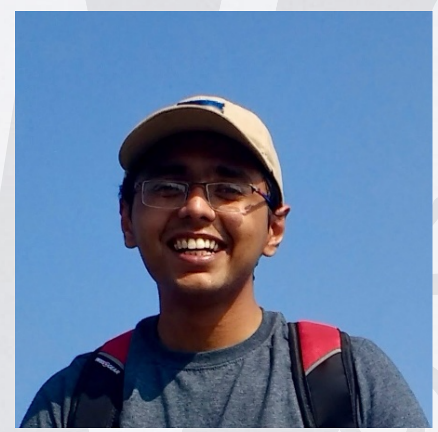
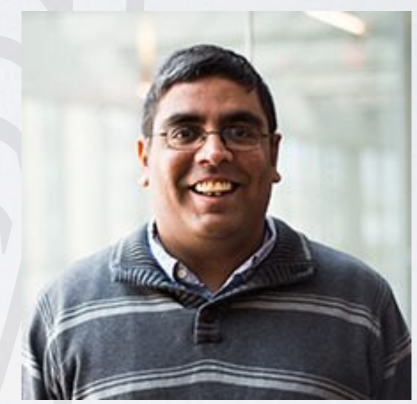
AAAI' 19

EXPLICITLY IMPOSING CONSTRAINTS IN DEEP NETWORKS VIA CONDITIONAL GRADIENTS GIVES IMPROVED GENERALIZATION AND FASTER CONVERGENCE



Sathya Ravi, Tuan Dinh, Vishnu Lokhande, Vikas Singh

Department of Computer Sciences

University of Wisconsin–Madison

11/14/2018

WISCONSIN

UNIVERSITY OF WISCONSIN–MADISON

DEEP LEARNING

DEEP LEARNING

Solve $$\min_{W \in \mathbb{R}^n} L(W)$$

DEEP LEARNING

Solve $$\min_{W \in \mathbb{R}^n} L(W)$$

$$L(W) = \mathbb{E}_\xi f(W, \xi)$$

$$\xi = (x, y) \sim \mathscr{D}$$

Compute an estimate of gradient

Compute an estimate of gradient

$$W_{t+1} = W_t - \eta_t \nabla \tilde{L}_t(W_t)$$

Compute an estimate of gradient

$$W_{t+1} = W_t - \eta_t \nabla \tilde{L}_t(W_t)$$

$$\mathbb{E}\left[\nabla \tilde{L}_t(W_t)\right] = \nabla L(W_t)$$

$$\mathbb{E}\left[\left\|\nabla \tilde{L}_t(W) - \nabla L(W)\right\|^2\right] \leq \sigma^2$$

Compute an estimate of gradient

$$W_{t+1} = W_t - \eta_t \nabla \tilde{L}_t(W_t)$$

What about learning?

$$\mathbb{E}\left[\nabla \tilde{L}_t(W_t)\right] = \nabla L(W_t)$$

$$\mathbb{E}\left[\left\|\nabla \tilde{L}_t(W) - \nabla L(W)\right\|^2\right] \leq \sigma^2$$

QUALITY/PREDICTIVE PERFORMANCE

QUALITY/PREDICTIVE PERFORMANCE

$$\mathcal{R}(W) = \mathbb{E}_{(x,y)\sim\mathcal{D}} L(W; (x, y))$$

QUALITY/PREDICTIVE PERFORMANCE

$$\mathcal{R}(W) = \mathbb{E}_{(x,y)\sim\mathcal{D}} L(W;(x,y))$$

$$\mathcal{R}_S(W) = \frac{1}{n}\sum_{i=1}^{n} L(W;(x_i,y_i))$$

QUALITY/PREDICTIVE PERFORMANCE

$$\mathcal{R}(W) = \mathbb{E}_{(x,y)\sim\mathcal{D}} L(W;(x,y))$$

$$\mathcal{R}_S(W) = \frac{1}{n}\sum_{i=1}^{n} L(W;(x_i,y_i))$$

The one true theorem

$$\mathcal{R}(W) = \mathcal{R}_S(W) + \mathcal{R}(W) - \mathcal{R}_S(W)$$

QUALITY/PREDICTIVE PERFORMANCE

$$\mathcal{R}(W) = \mathbb{E}_{(x,y)\sim\mathcal{D}} L(W; (x, y))$$

$$\mathcal{R}_S(W) = \frac{1}{n}\sum_{i=1}^{n} L(W; (x_i, y_i))$$

The one true theorem

$$\mathcal{R}(W) = \underbrace{\mathcal{R}_S(W)}_{\text{Train error}} + \underbrace{\mathcal{R}(W) - \mathcal{R}_S(W)}_{\Delta_S(W) := \text{Test error}}$$

Train error **Δ_S(W):=Test error**

QUALITY/PREDICTIVE PERFORMANCE

QUALITY/PREDICTIVE PERFORMANCE

Classical

$\Delta_S(W) \propto \#\text{parameters}$

QUALITY/PREDICTIVE PERFORMANCE

Classical

$\Delta_S(W) \propto \#\text{parameters}$

Modern (refined)

$\Delta_S(W) \propto \|W\|$

QUALITY/PREDICTIVE PERFORMANCE

Classical

$\Delta_S(W) \propto \#\text{parameters}$

Modern (refined)

$\Delta_S(W) \propto \|W\|$

Simple idea: Enforce “high” quality

HOW TO HAVE YOUR CAKE AND EAT IT (TOO)?

HOW TO HAVE YOUR CAKE AND EAT IT (TOO)?

Ingredients

HOW TO HAVE YOUR CAKE AND EAT IT (TOO)?

Ingredients

1. High quality: constraints with nice theoretical properties

HOW TO HAVE YOUR CAKE AND EAT IT (TOO)?

Ingredients

1. High quality: constraints with nice theoretical properties
2. A “fast” algorithm: Projection free approaches are more parsimonious

HOW TO HAVE YOUR CAKE AND EAT IT (TOO)?

Ingredients

1. High quality: constraints with nice theoretical properties
2. A “fast” algorithm: Projection free approaches are more parsimonious
3. Resources: GPUs, fellow graduate students, adviser etc.

CONTRIBUTIONS

CONTRIBUTIONS

- Enforcing various generic constraints: $R(W) \leq \lambda$

CONTRIBUTIONS

- Enforcing various generic constraints: $R(W) \leq \lambda$

- Enforcing path norm constraint

CONTRIBUTIONS

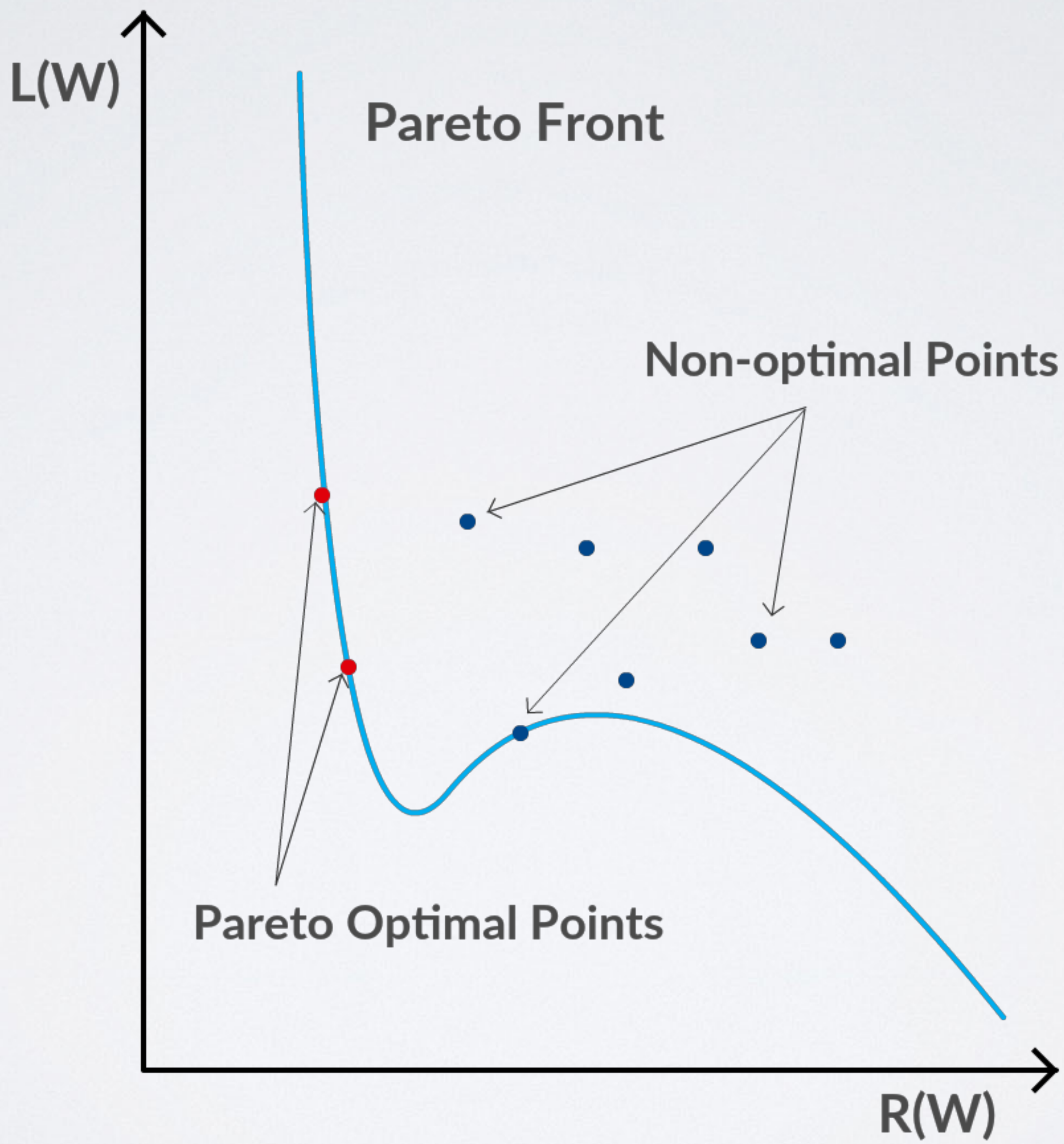
- Enforcing various generic constraints: $R(W) \leq \lambda$

- Enforcing path norm constraint

- Experiments with three different tasks and datasets

WHY?

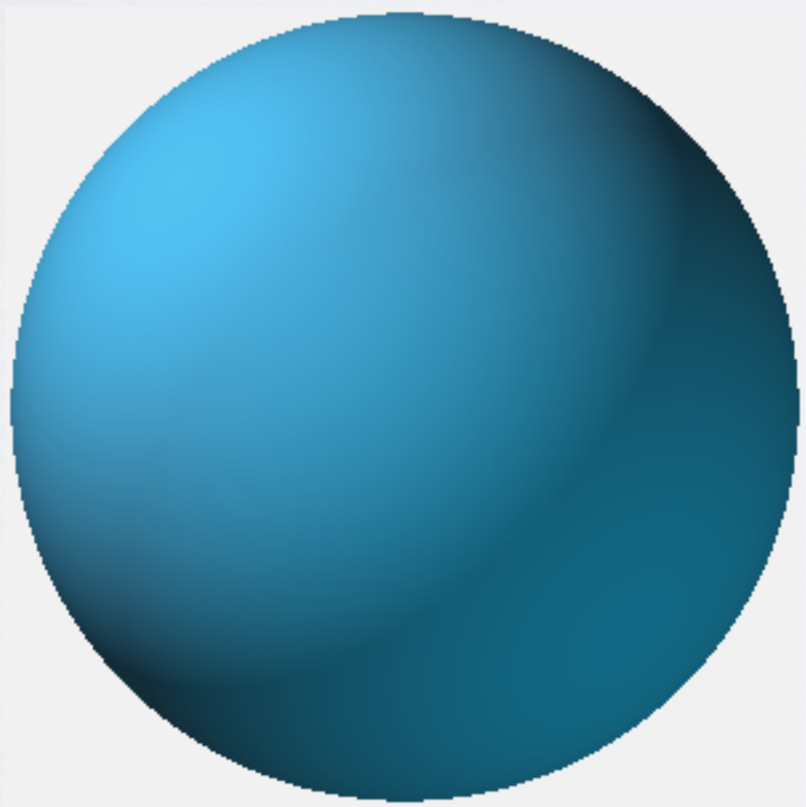
WHY?


L(W)
Pareto Front
Non-optimal Points
Pareto Optimal Points
R(W)

EXCELLENT GENERIC CONSTRAINTS

EXCELLENT GENERIC CONSTRAINTS

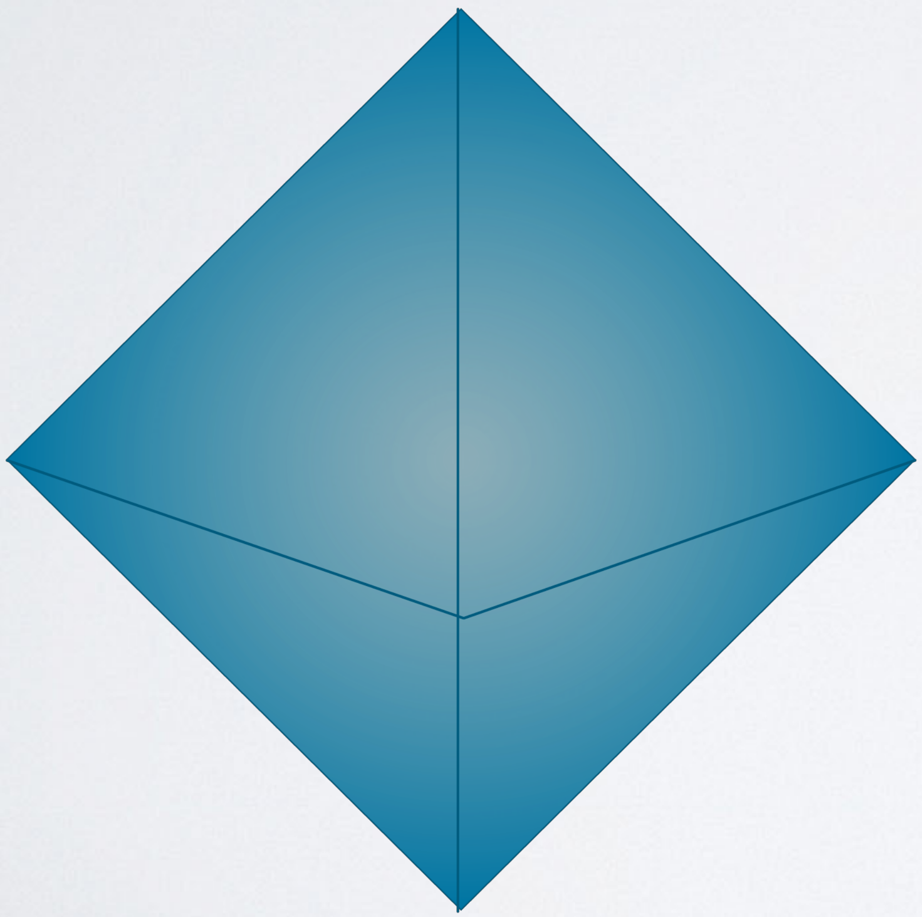
$$\|W\|_F, \quad \|W\|_*$$



GOOD GENERIC CONSTRAINTS

GOOD GENERIC CONSTRAINTS

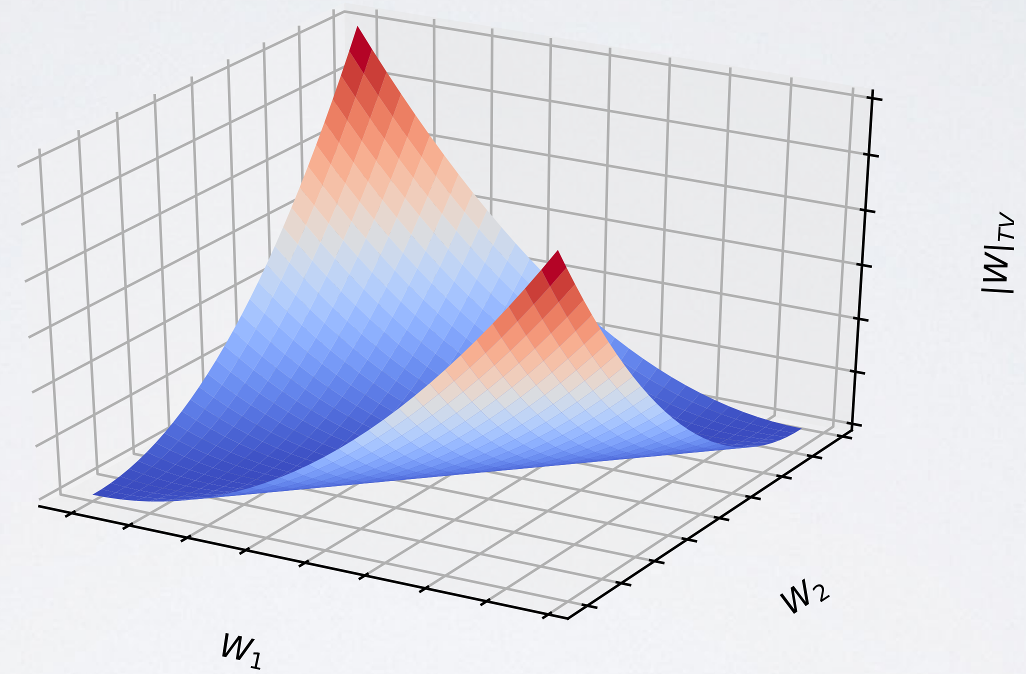
$$\|W\|_1, \quad \|W\|_\infty$$



PESKY GENERIC CONSTRAINTS

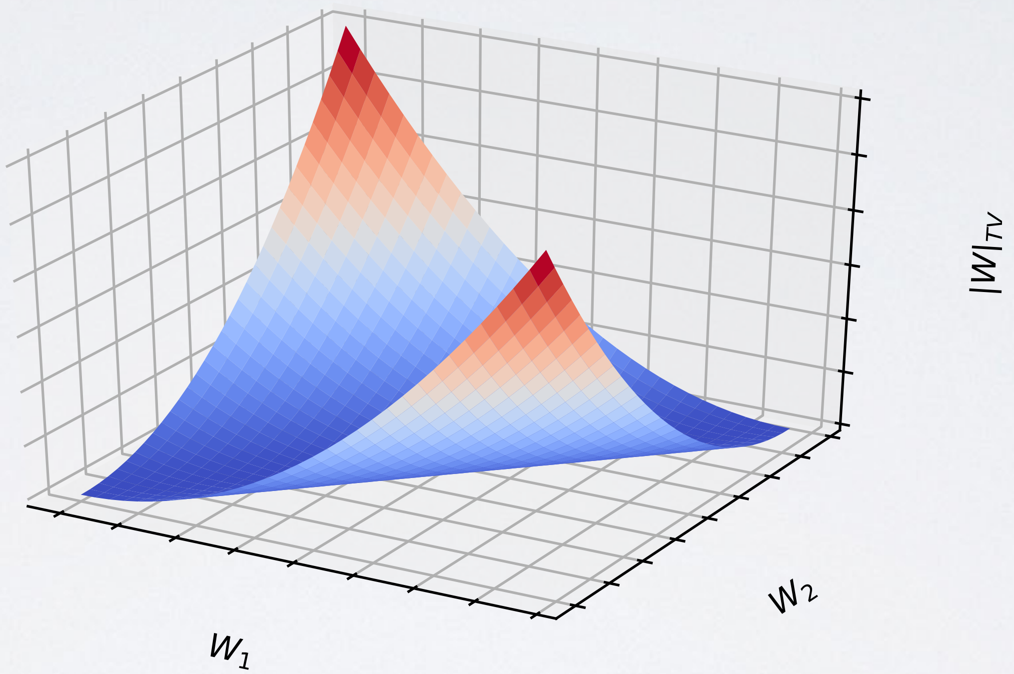
PESKY GENERIC CONSTRAINTS

$$\|W\|_{\mathbf{TV}}$$



PESKY GENERIC CONSTRAINTS

$$\|W\|_{\mathbf{TV}}$$



Lemma: An $\epsilon -$ approximate CG step can be computed in $O(1/\epsilon)$ time (independent of dimensions).

PATH CG

PATH CG

$$\|W\|_\pi^2 = \sum_{v_{in}[i] \xrightarrow{e_1} v_1 \xrightarrow{e_2} \cdots v_{out}[j]} \left| \prod_{k=1}^{l} W_{e_k} \right|^2$$

PATH CG

$$\|W\|_\pi^2 = \sum_{v_{in}[i] \xrightarrow{e_1} v_1 \xrightarrow{e_2} \cdots v_{out}[j]} \left| \prod_{k=1}^{l} W_{e_k} \right|^2$$

- Why path norm? Rescale invariance

PATH CG

$$\|W\|_\pi^2 = \sum_{v_{in}[i] \xrightarrow{e_1} v_1 \xrightarrow{e_2} \cdots v_{out}[j]} \left| \prod_{k=1}^{l} W_{e_k} \right|^2$$

- Why path norm? Rescale invariance

PATH CG

$$\|W\|_\pi^2 = \sum_{v_{in}[i]\xrightarrow{e_1}v_1\xrightarrow{e_2}\cdots v_{out}[j]} \left| \prod_{k=1}^{l} W_{e_k} \right|^2$$

- Why path norm? Rescale invariance

- Projection at least NP Hard

PATH CG

$$\|W\|_\pi^2 = \sum_{v_{in}[i] \xrightarrow{e_1} v_1 \xrightarrow{e_2} \cdots v_{out}[j]} \left| \prod_{k=1}^{l} W_{e_k} \right|^2$$

- Why path norm? Rescale invariance

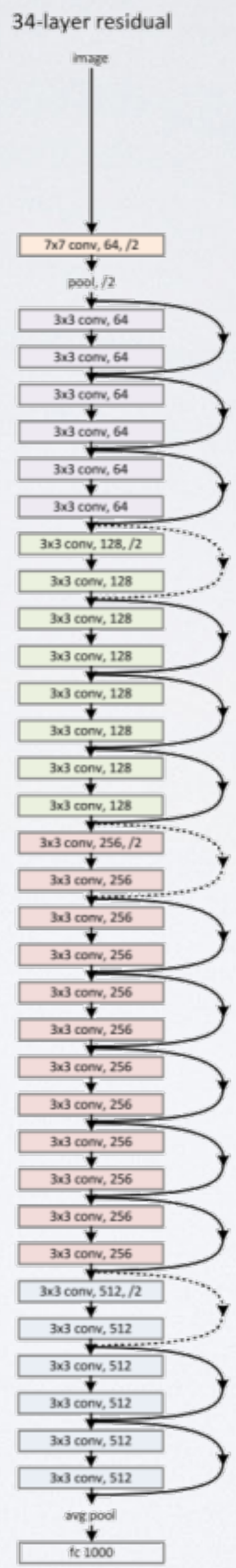
- Projection at least NP Hard

PATH CG

$$\|W\|_\pi^2 = \sum_{v_{in}[i] \xrightarrow{e_1} v_1 \xrightarrow{e_2} \cdots v_{out}[j]} \left| \prod_{k=1}^{l} W_{e_k} \right|^2$$

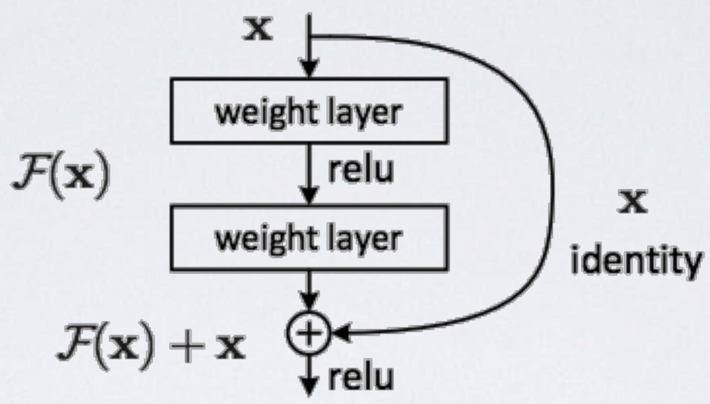
- Why path norm? Rescale invariance
- Projection at least NP Hard
- Subproblems in CG can be solved in O(Bl)

RESNETS

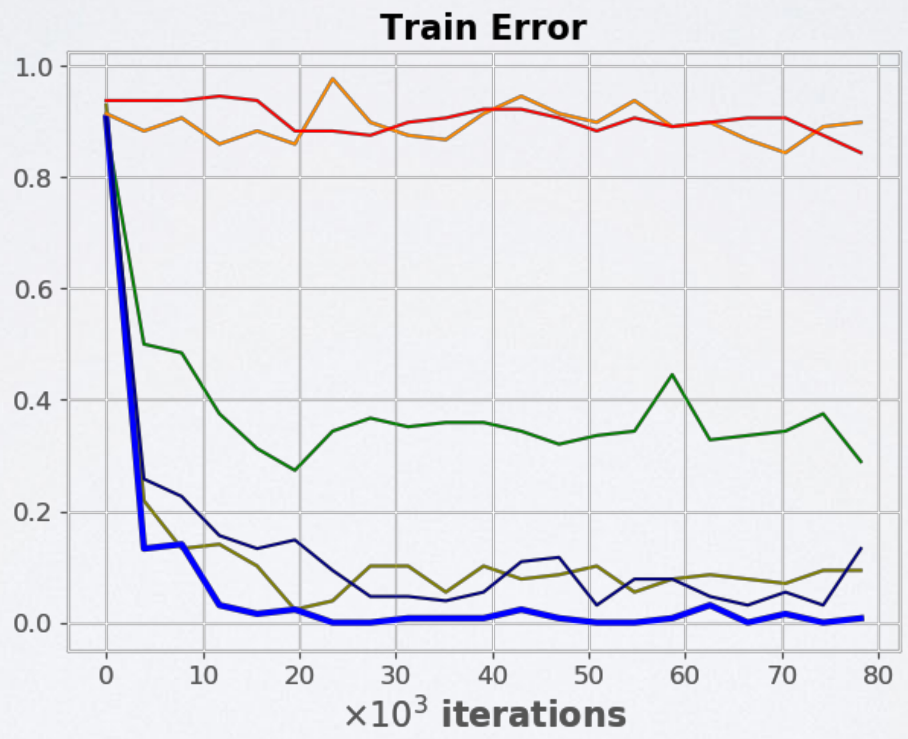


34-layer residual
image
7x7 conv, 64, /2
pool, /2
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 128, /2
3x3 conv, 128
3x3 conv, 128
3x3 conv, 128
3x3 conv, 128
3x3 conv, 128
3x3 conv, 128
3x3 conv, 128
3x3 conv, 256, /2
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
3x3 conv, 512, /2
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
avg pool
fc 1000

RESNETS

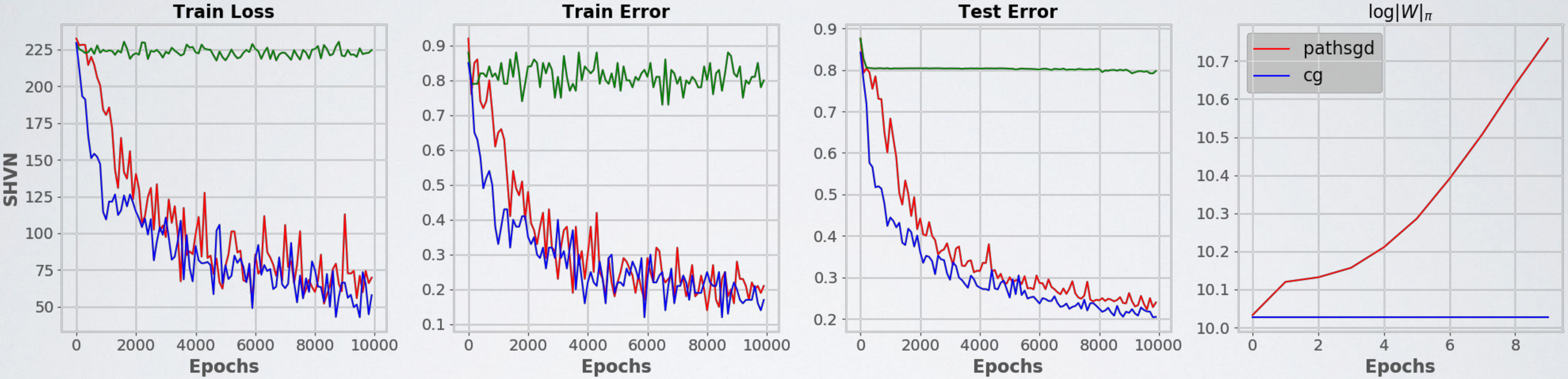

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu


Train Loss
$\log_{10}\lambda = 1$
$\log_{10}\lambda = 2$
$\log_{10}\lambda = 3$
$\log_{10}\lambda = 4$
$\log_{10}\lambda = 5$
$\log_{10}\lambda = 6$
$\times 10^3$ iterations

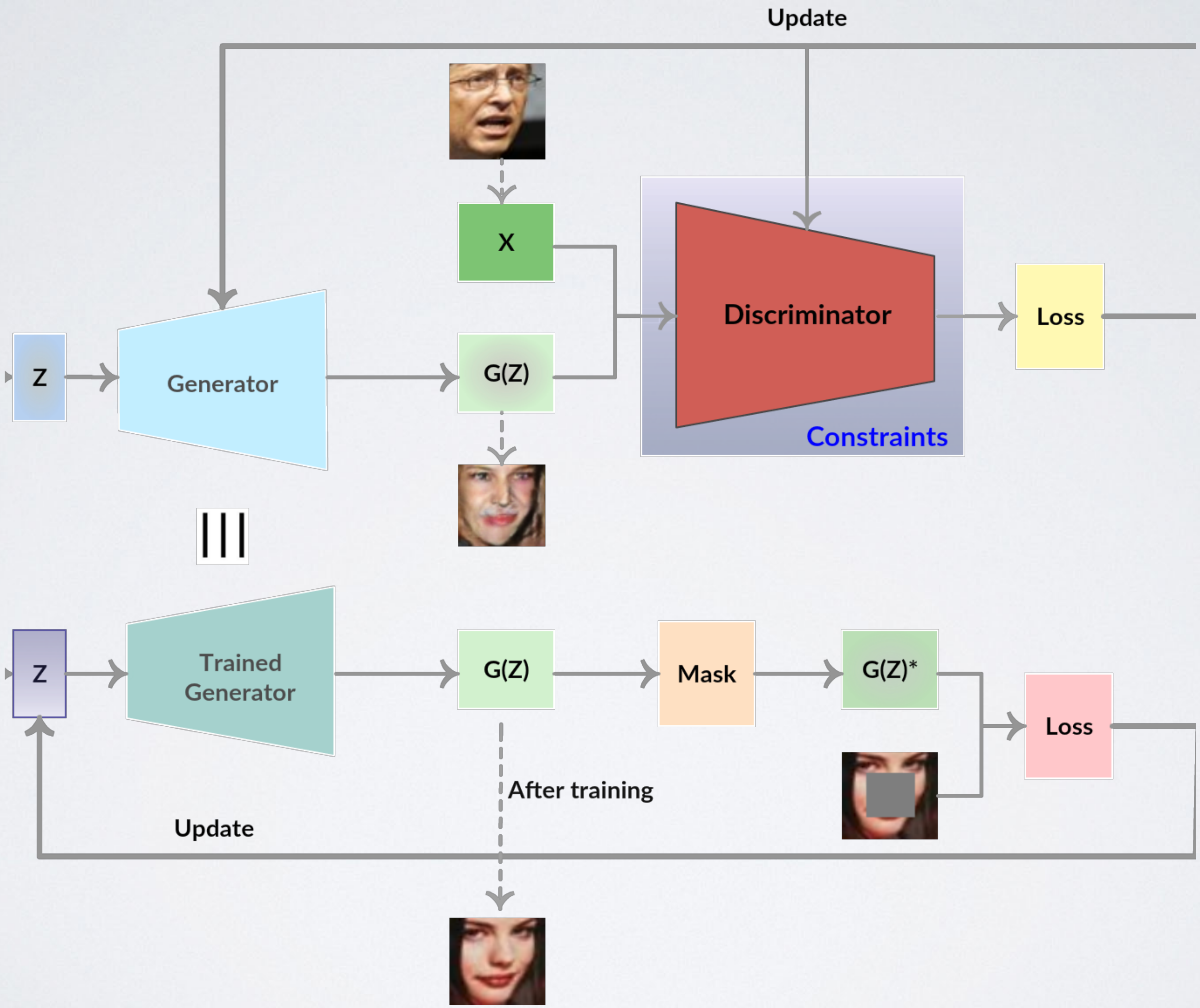

Train Error
$\times 10^3$ iterations


Validation Error
$\times 10^3$ iterations

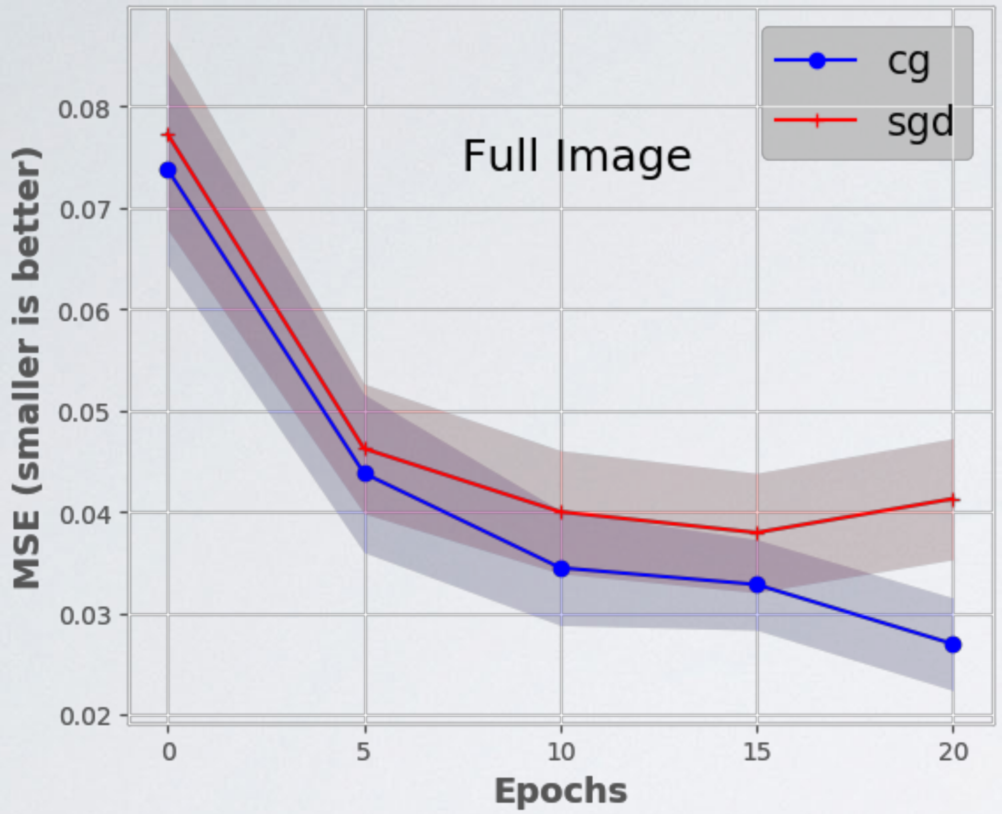
PATH-CG


Train Loss
Train Error
Test Error
$\log|W|_\pi$
SHVN
Epochs
Epochs
Epochs
Epochs
pathsgd
cg

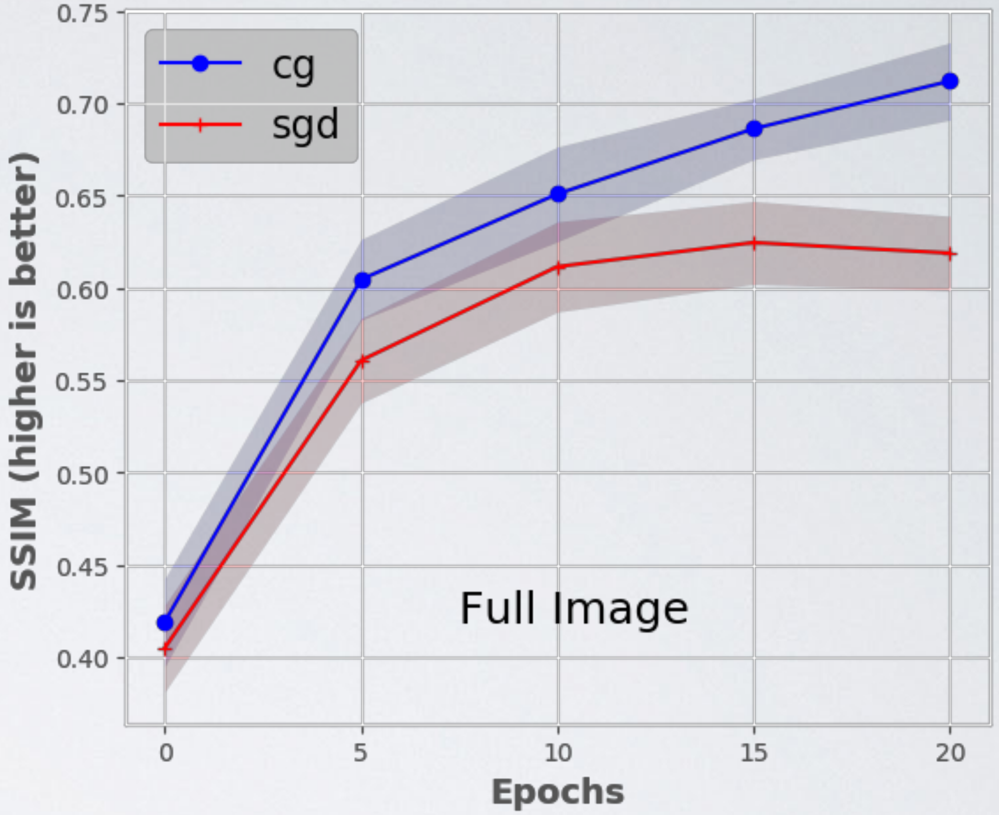
DC-GAN


Update
X
Discriminator
Loss
Z
Generator
G(Z)
Constraints
Z
Trained Generator
G(Z)
Mask
G(Z)*
Loss
After training
Update

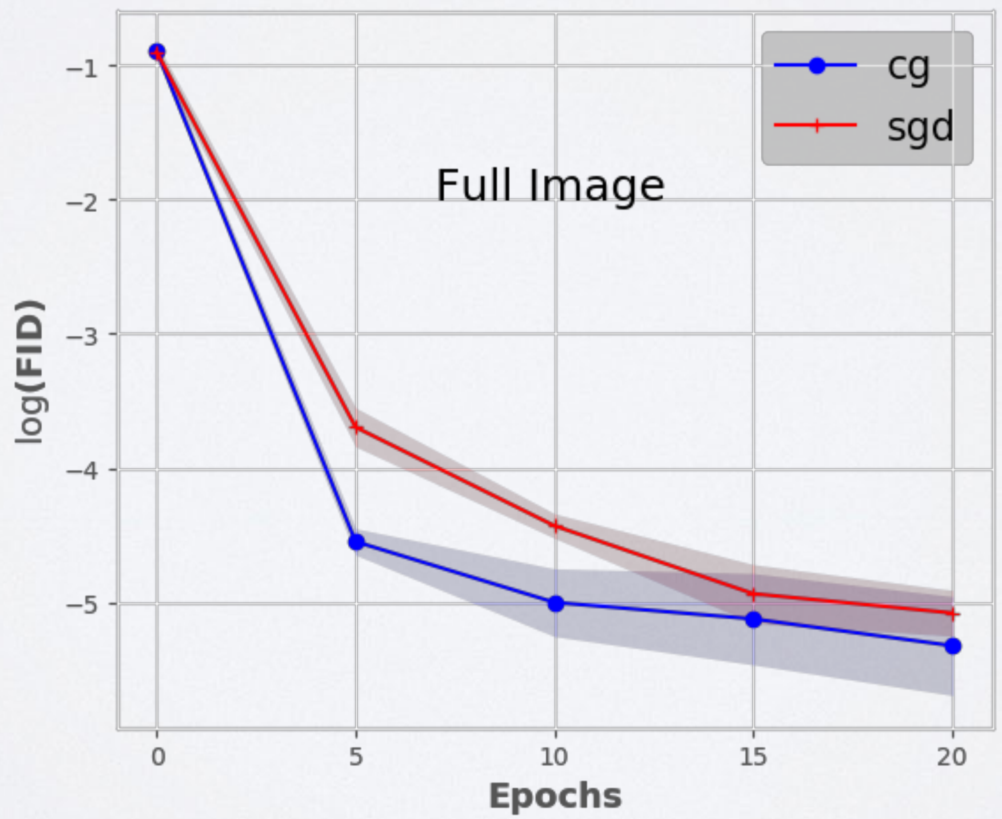
DC-GAN



cg
sgd
Full Image
MSE (smaller is better)
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0
5
10
15
20
Epochs

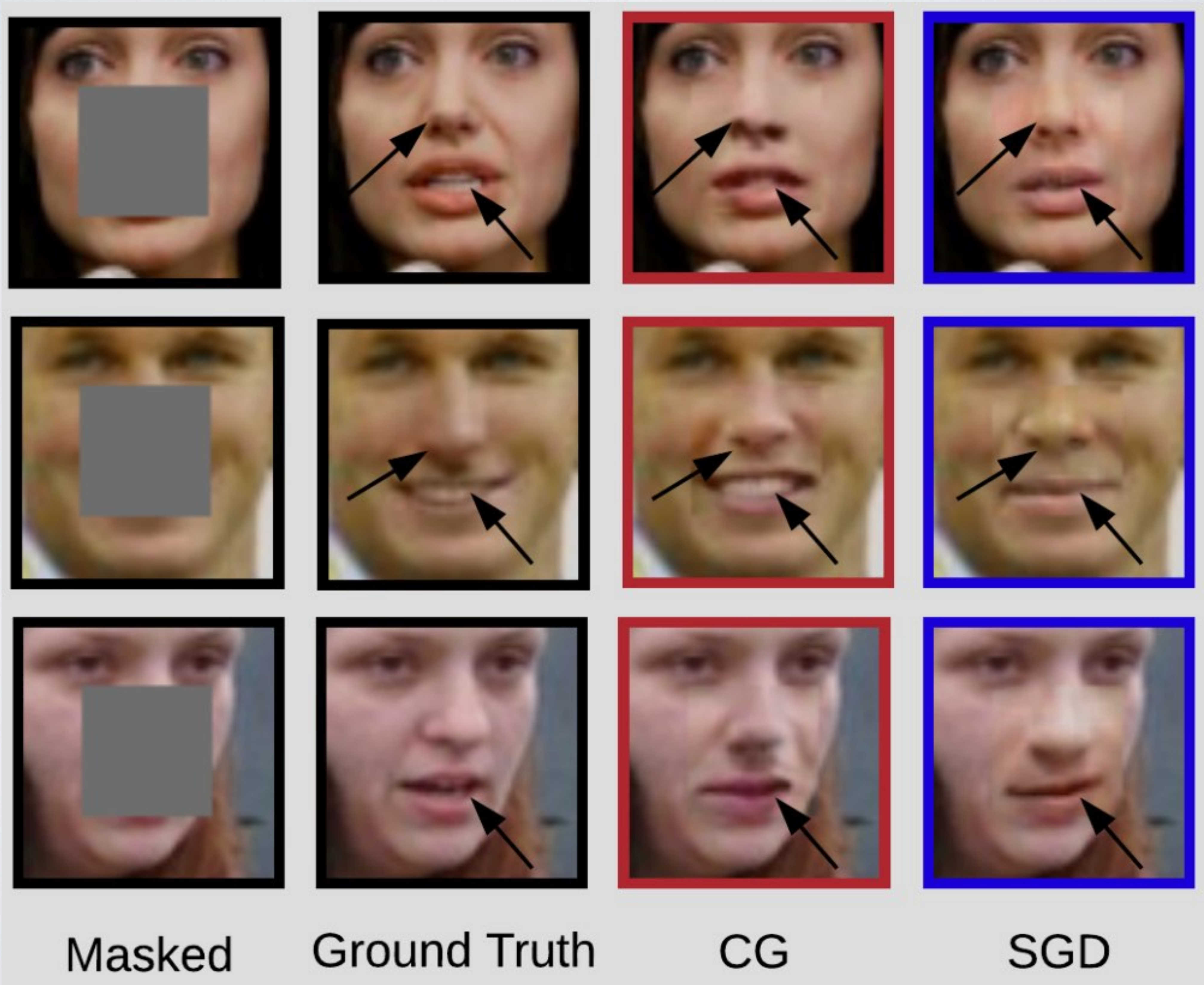


cg
sgd
Full Image
SSIM (higher is better)
0.75
0.70
0.65
0.60
0.55
0.50
0.45
0.40
0
5
10
15
20
Epochs



cg
sgd
Full Image
log(FID)
−1
−2
−3
−4
−5
0
5
10
15
20
Epochs

DC-GAN



FOR MORE DETAILS ON THEORY AND EXPERIMENTS, CATCH US AT

POSTER #1!