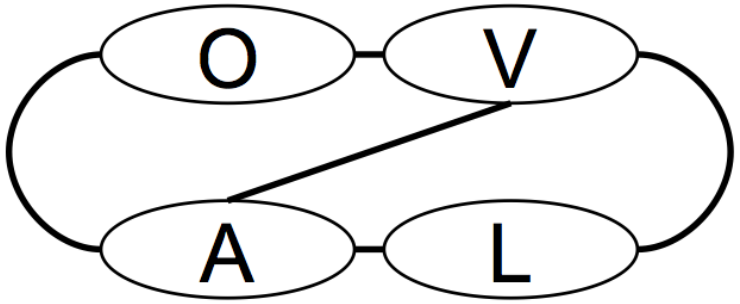


Discriminative Machine Learning

Topic 5: *Neural Network Optimization*

M. Pawan Kumar

Slides available online http://mpawankumar.info

Optimization for Deep Learning

Many principled convex optimization algorithms

Deep learning objectives are highly non-convex

Intuition from convex optimization is borrowed

But there is a lack of theoretical guarantees

Outline

- Gradient Descent with Nesterov Momentum
- AdaGrad
- AdaDelta
- Adam

Subgradient Descent

Consider a convex function f

Step size η_t s.t. $\lim_{t\to\infty} \eta_t = 0$ and $\sum_t \eta_t = \infty$

Intuition?

Let $\mathbf{x}_t$ be the solution at iteration t

Provably, $\sum_s f(\mathbf{x}_s)/t - f(\mathbf{x}^*) \leq O(1/\sqrt{t})$

Subgradient Descent

Step size diminishes because subgradients may not

Step sizes form divergent series to allow us to go far

No guarantee of monotonic improvement

Hence convergence rate involves average value

Can we go faster? Only with additional assumptions

Gradient Descent

Consider a smooth convex function f

β-Lipschitz gradients: $||\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})|| \leq \beta||\mathbf{x}-\mathbf{y}||$

Let $\mathbf{x}_t$ be the solution at iteration t of gradient descent

Provably, $f(\mathbf{x}_t) - f(\mathbf{x}^*) \leq O(1/t)$ for step-size $\eta = 1/\beta$

Speed-up of $O(1/\sqrt{t})$ Intuition?

Gradient Descent

Smoothness allows the use of fixed step-size

Monotonic improvement in objective function value

In practice, β is unknown

Step-size is cross-validated

Can we go faster?

Momentum

Gradients change smoothly so use previous gradients

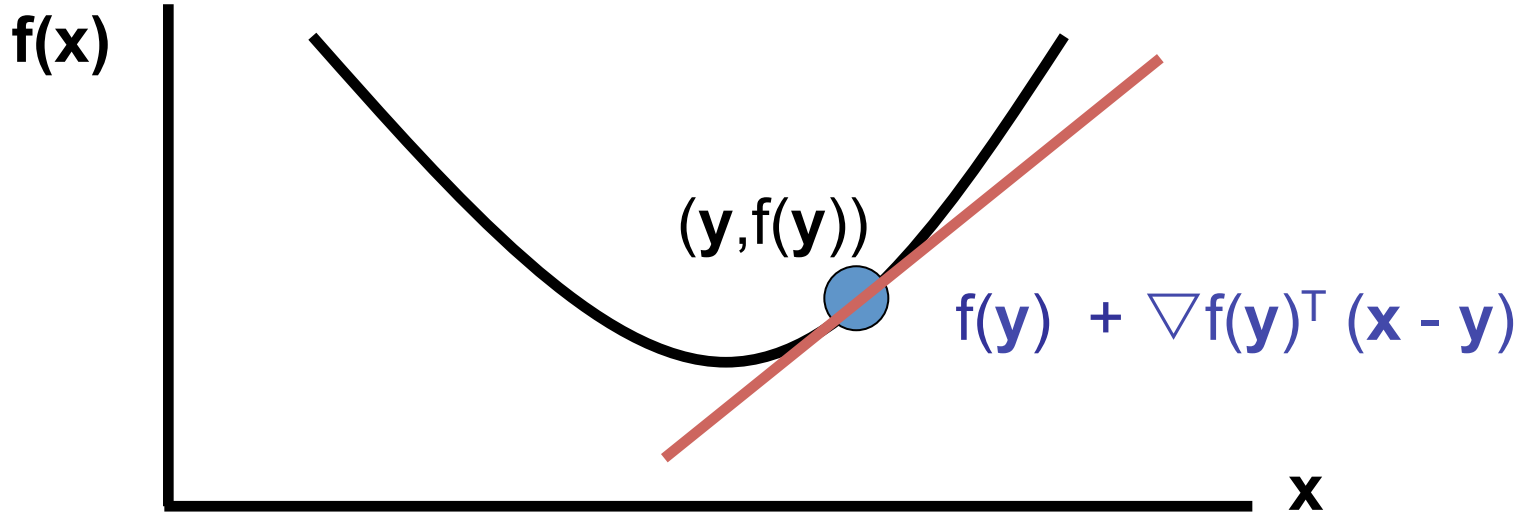
Define $\mathbf{v}_0 = 0$. Compute $\mathbf{v}_{t+1} = \alpha\mathbf{v}_t - \eta\nabla f(\mathbf{x}_t)$

Update $\mathbf{x}_{t+1} = \mathbf{x}_t + \mathbf{v}_{t+1}$

If f is μ strongly convex, speed-up of $O(\sqrt{\beta}/\sqrt{\mu})$

Polyak, 1964

Strong Convexity



Tangent lies below curve (convexity)

Tangent only touches curve at **y** (strong convexity)

$f(\mathbf{x}) \geq f(\mathbf{y}) + \nabla f(\mathbf{y})^T(\mathbf{x}-\mathbf{y}) + \mu||\mathbf{x}-\mathbf{y}||^2$

Momentum

Gradients change smoothly so use previous gradients

Define $\mathbf{v}_0 = 0$. Compute $\mathbf{v}_{t+1} = \alpha\mathbf{v}_t - \eta\nabla f(\mathbf{x}_t)$

Could be wrong

Update $\mathbf{x}_{t+1} = \mathbf{x}_t + \mathbf{v}_{t+1}$

If f is μ strongly convex, speed-up of $O(\sqrt{\beta}/\sqrt{\mu})$

Polyak, 1964

Nesterov Momentum

Gradients change smoothly so use previous gradients

Define $\mathbf{v}_0 = 0$. Compute $\mathbf{v}_{t+1} = \alpha\mathbf{v}_t - \eta\nabla f(\mathbf{x}_t+\alpha\mathbf{v}_t)$

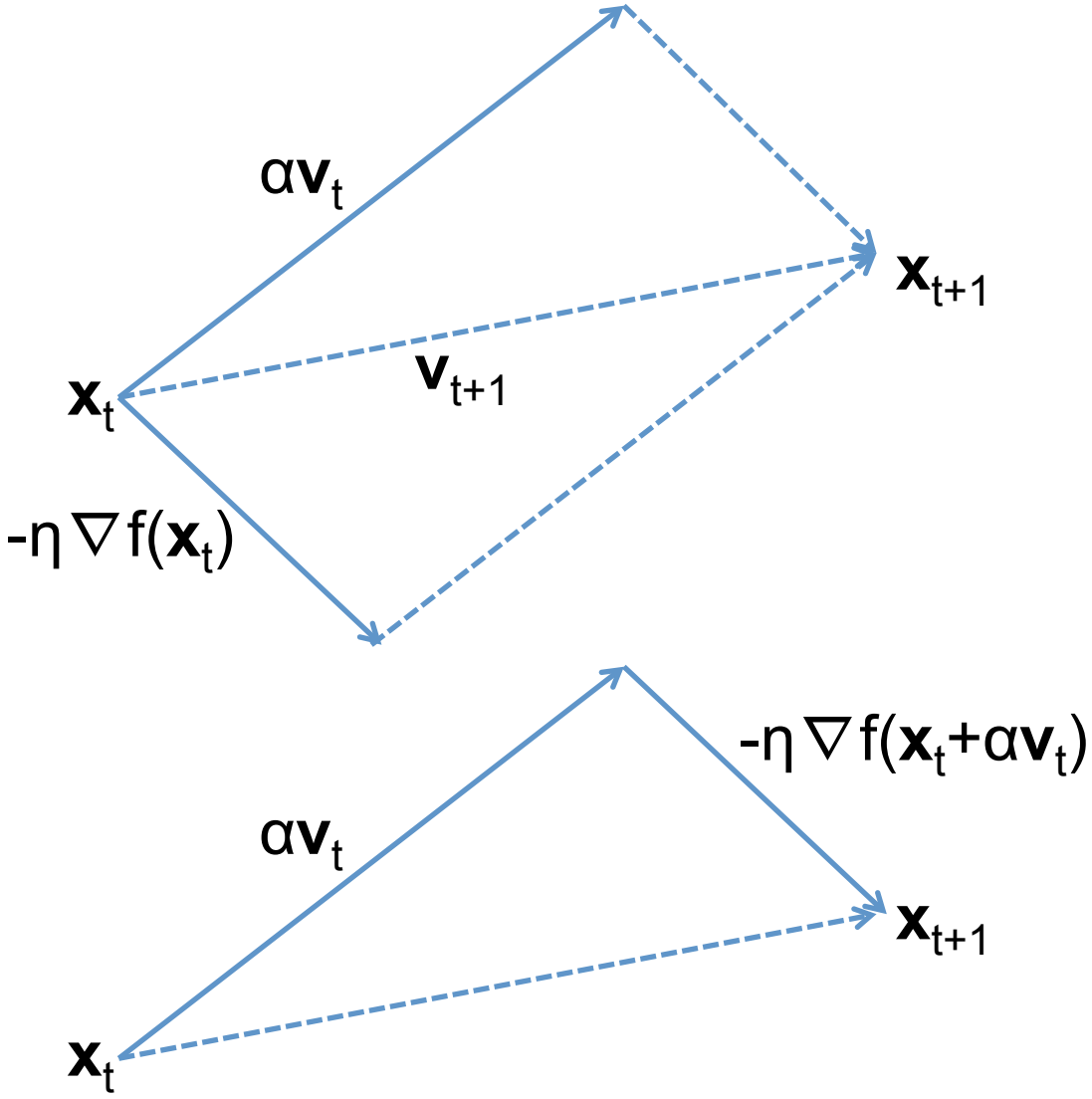
Make an immediate correction

Update $\mathbf{x}_{t+1} = \mathbf{x}_t + \mathbf{v}_{t+1}$

Provably, $f(\mathbf{x}_t) - f(\mathbf{x}^*) \leq O(1/t^2)$

Nesterov, 1983; Sutskever et al., 2013

Classic vs. Nesterov Momentum



Nesterov Momentum

Also called accelerated gradient

Requires a convex function

Ignore

Requires a smooth function

Cross entropy is smooth wrt **w**

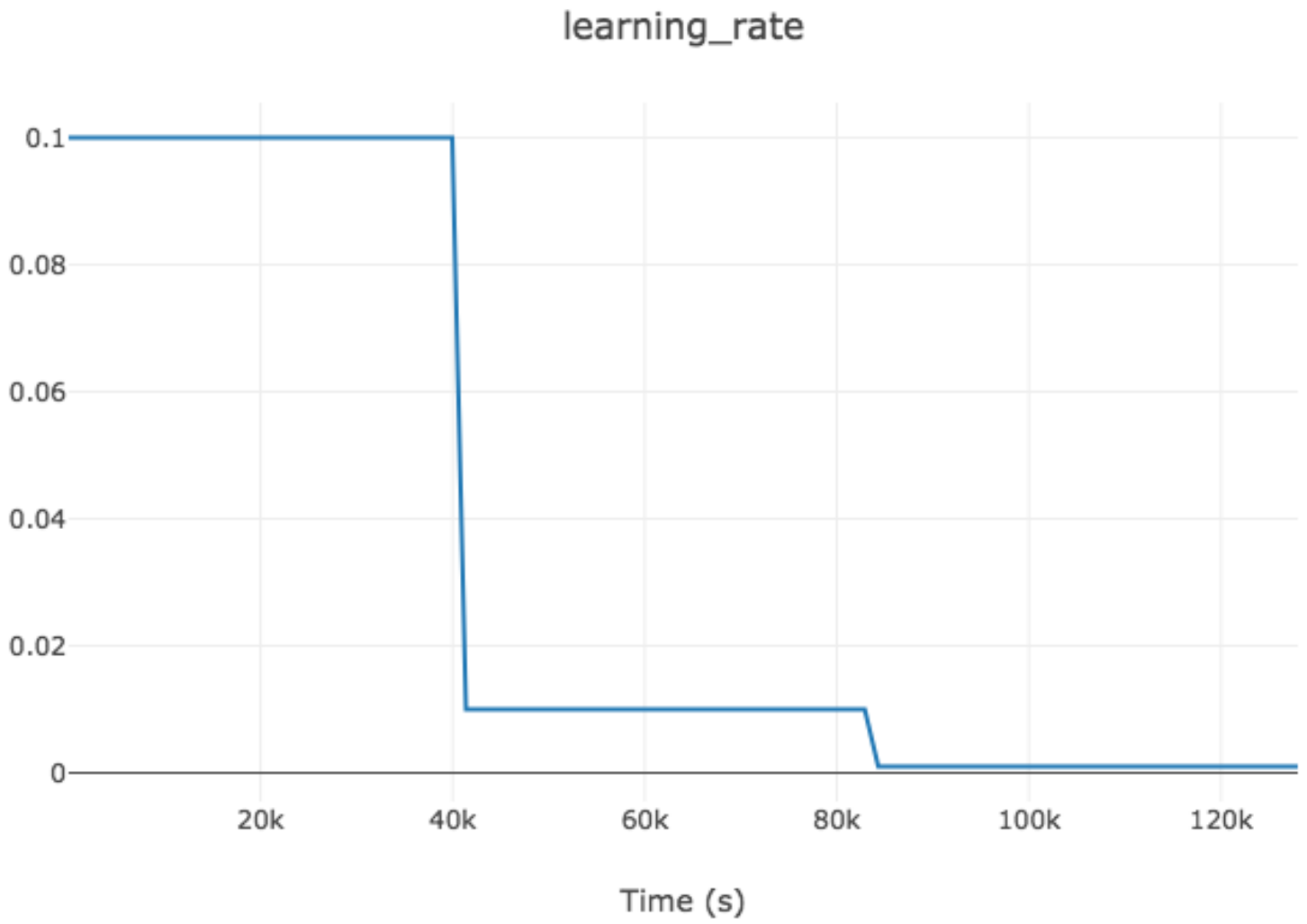
Hinge loss is not

Not a stochastic algorithm

Ignore

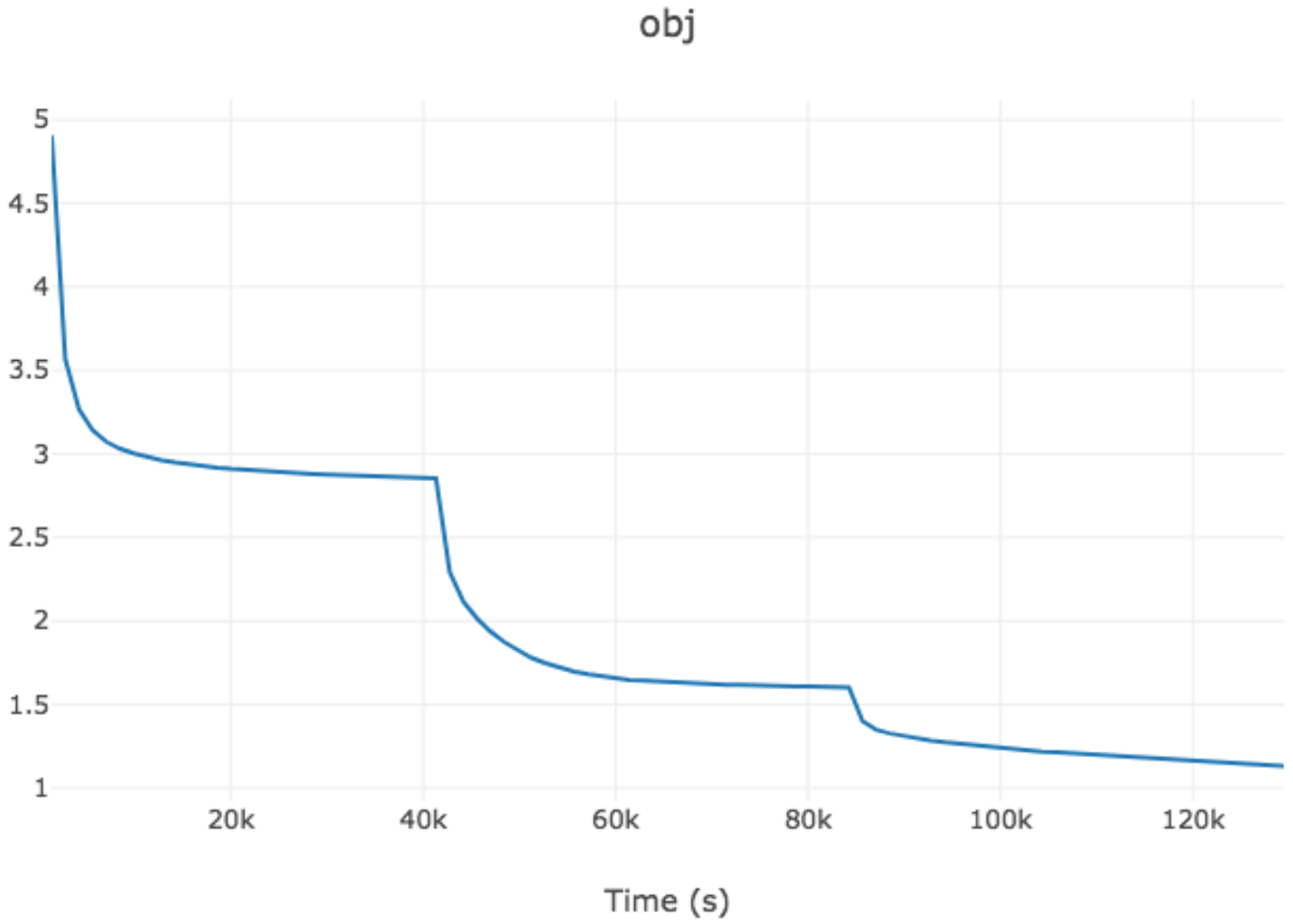
ImageNet with ResNet-18

Set α = 0.9 Decay η after 30 epochs



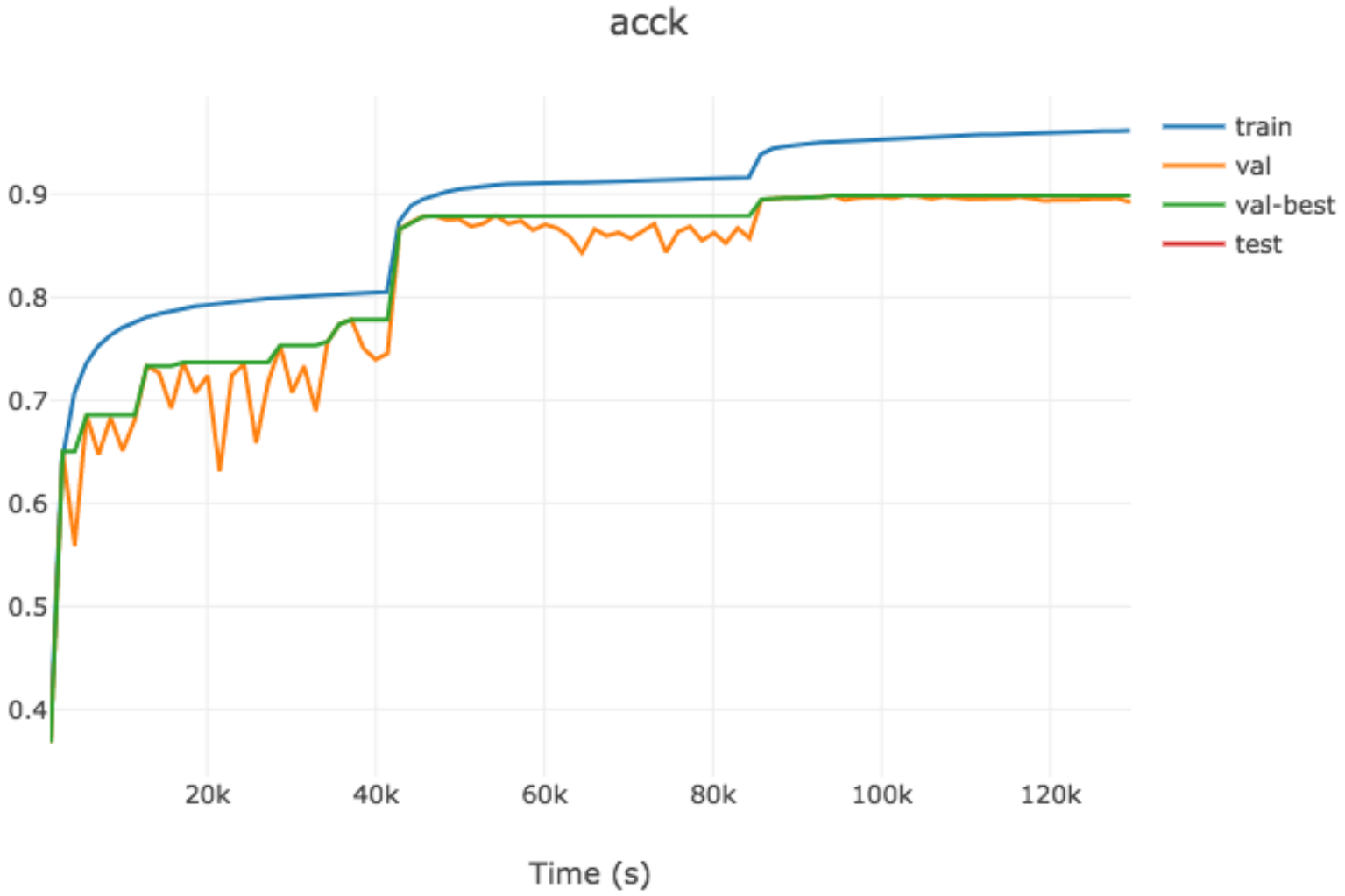
ImageNet with ResNet-18

Set $\alpha = 0.9$ Decay η after 30 epochs



ImageNet with ResNet-18

Set α = 0.9 Decay η after 30 epochs



Nesterov Momentum

Also called accelerated gradient

Requires a convex function

Ignore

Requires a smooth function

Cross entropy is smooth wrt **w**

Hinge loss is not

Not a stochastic algorithm

Ignore

Smoothing

$f(\mathbf{x}) = \max_i \{\mathbf{a}_i^T\mathbf{x}+b_i\}$ log-sum-exp smoothing

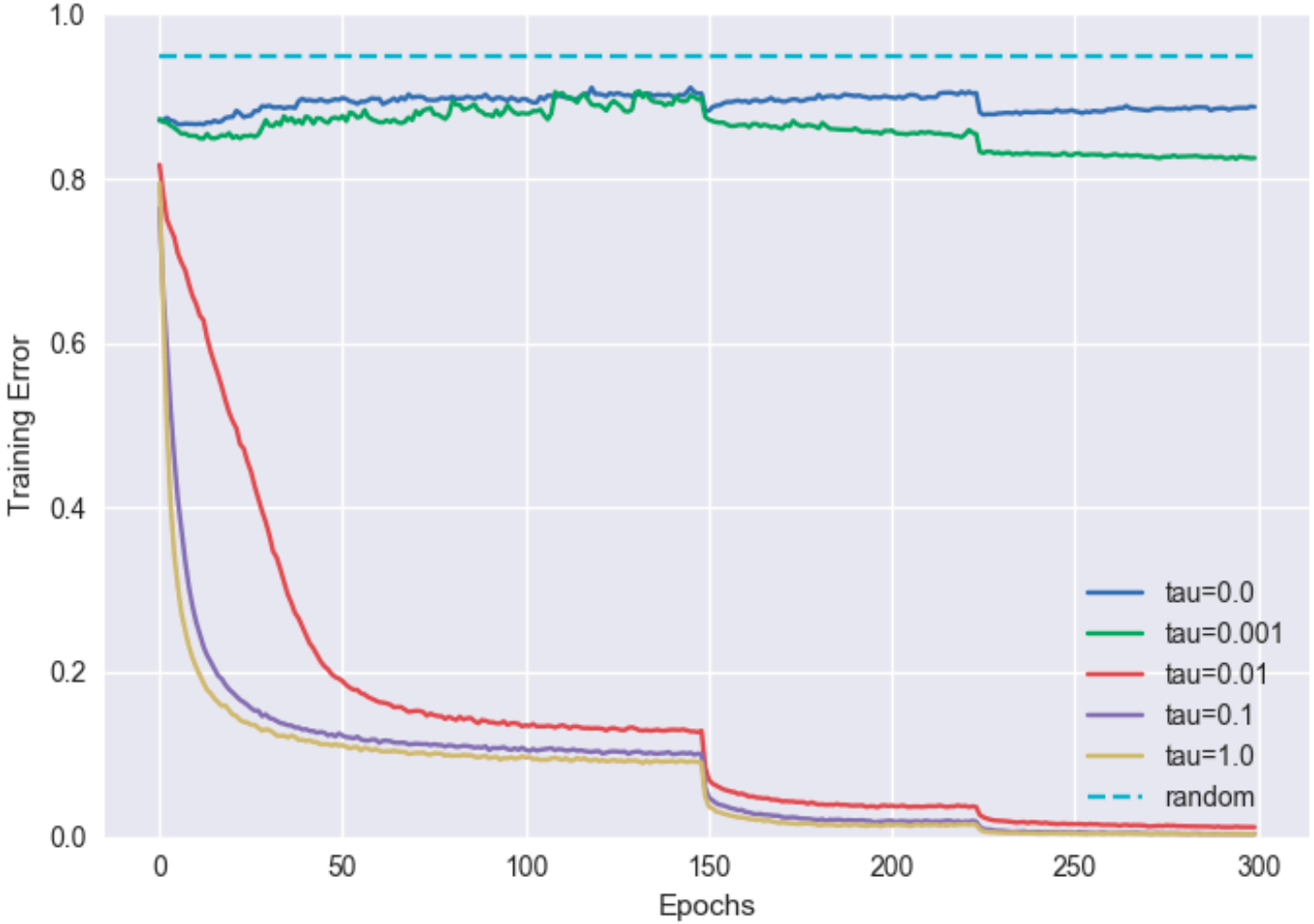
$g(\mathbf{x}) = \tau \log(\sum_i \exp((\mathbf{a}_i^T\mathbf{x}+b_i)/\tau))$

As $\tau \rightarrow 0+$, $g(\mathbf{x}) \rightarrow f(\mathbf{x})$

Obtains ε-optimal solutions for original problem

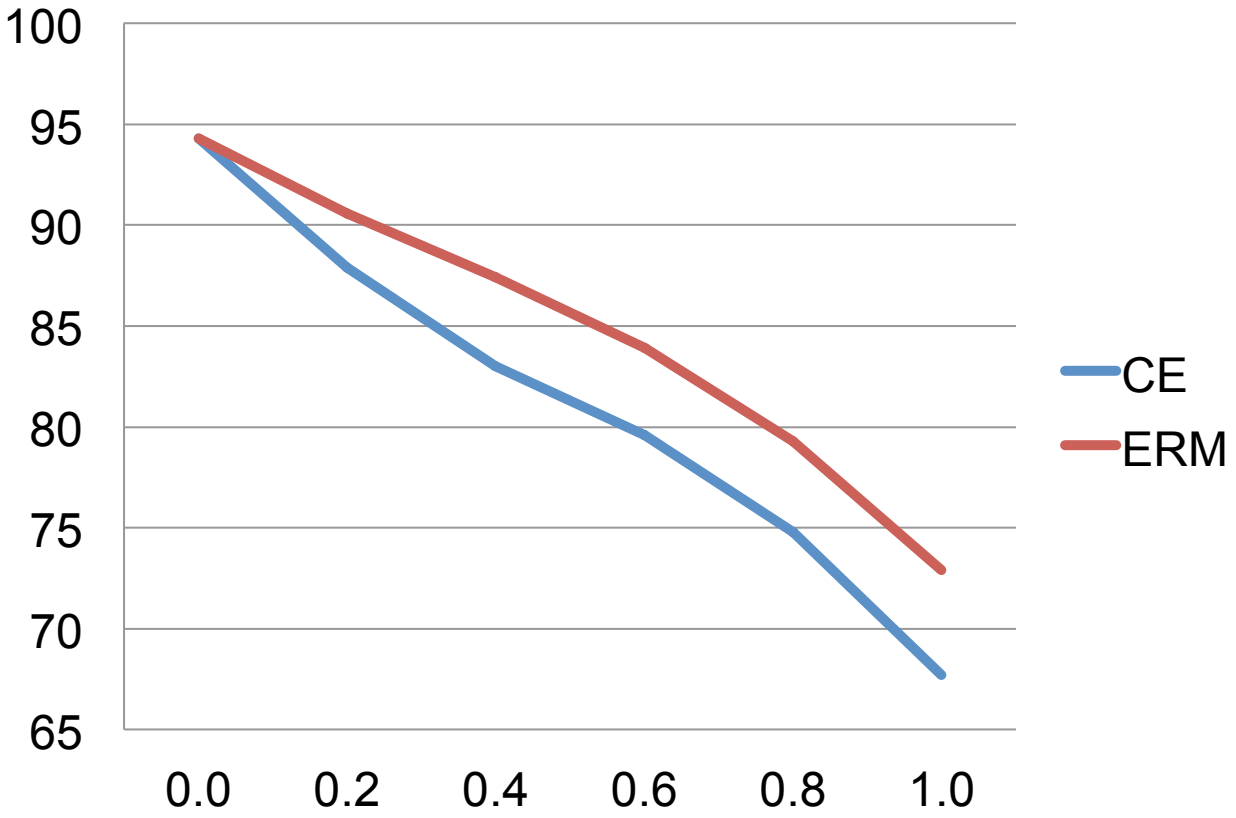
Nesterov, 2005; Beck and Teboulle, 2012

CIFAR-100 with DenseNet-40-12

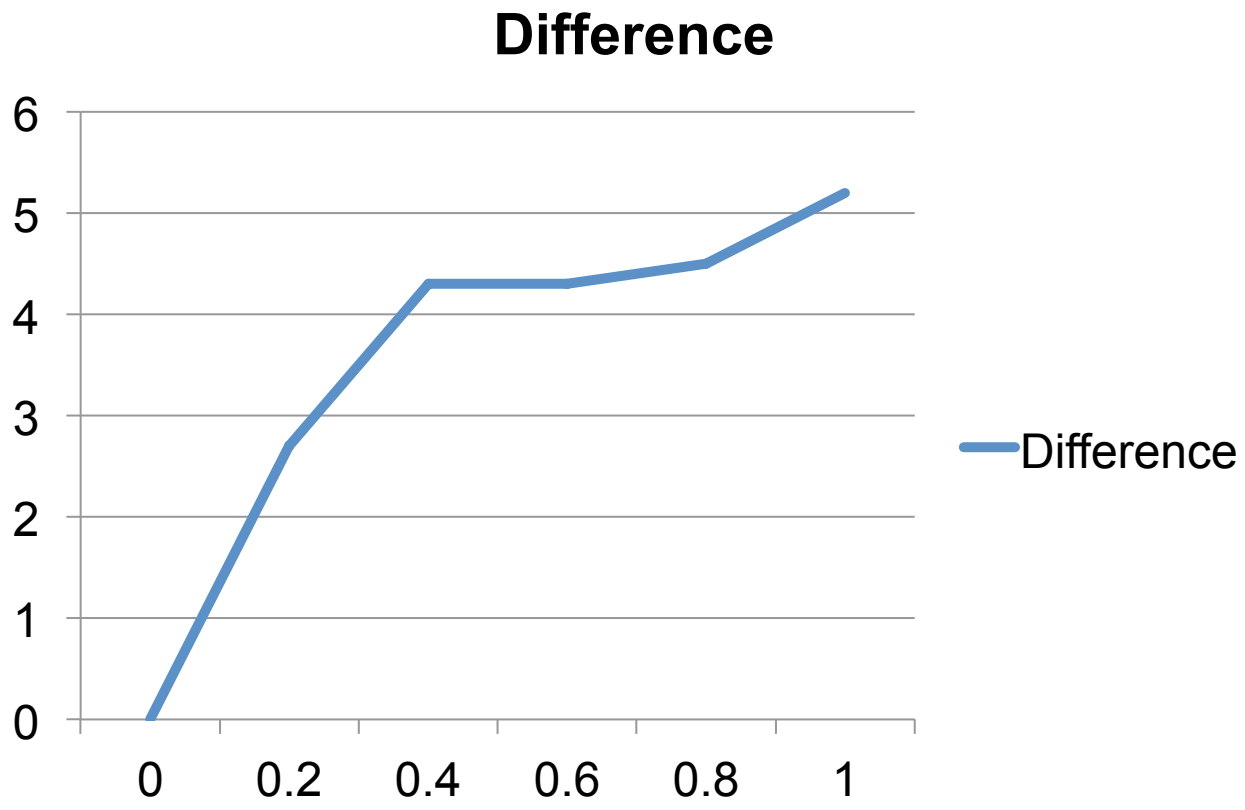


Berrada, Zisserman and Kumar, 2018

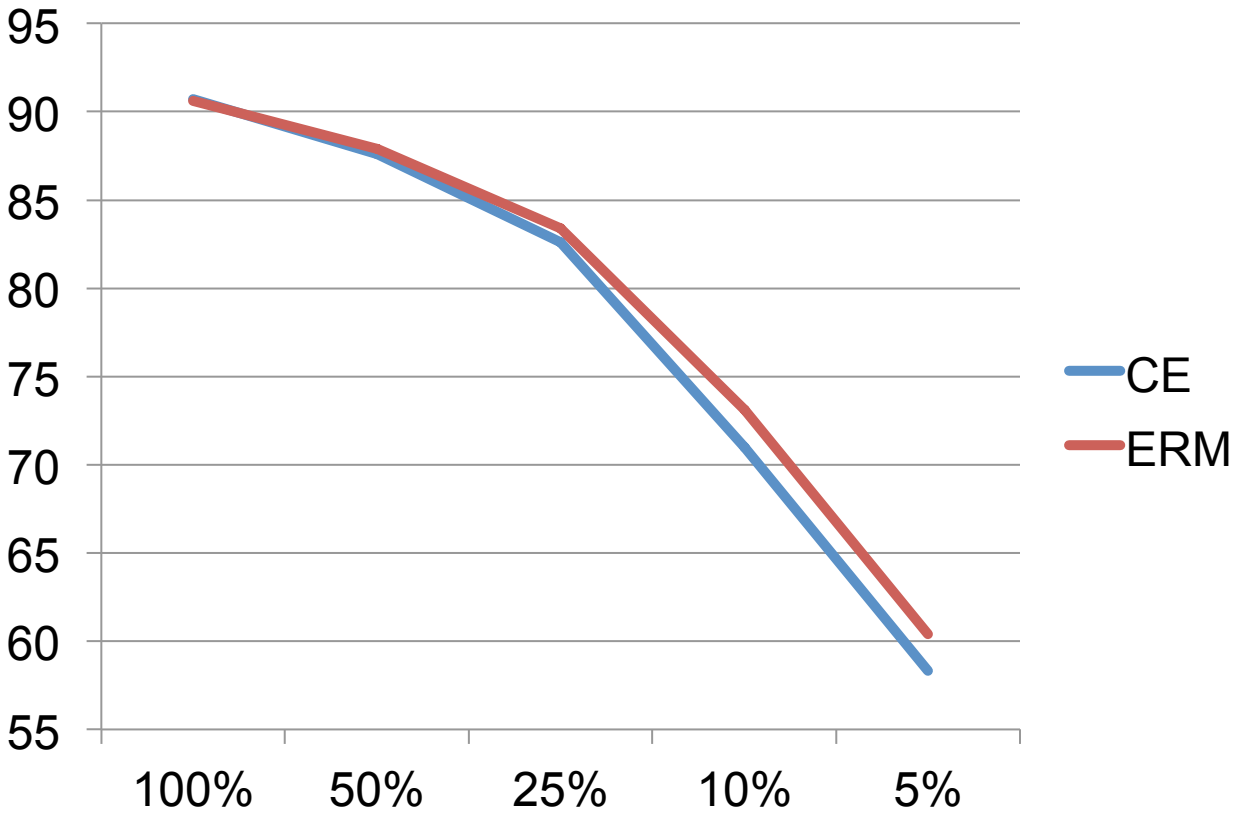
Noisy CIFAR-100



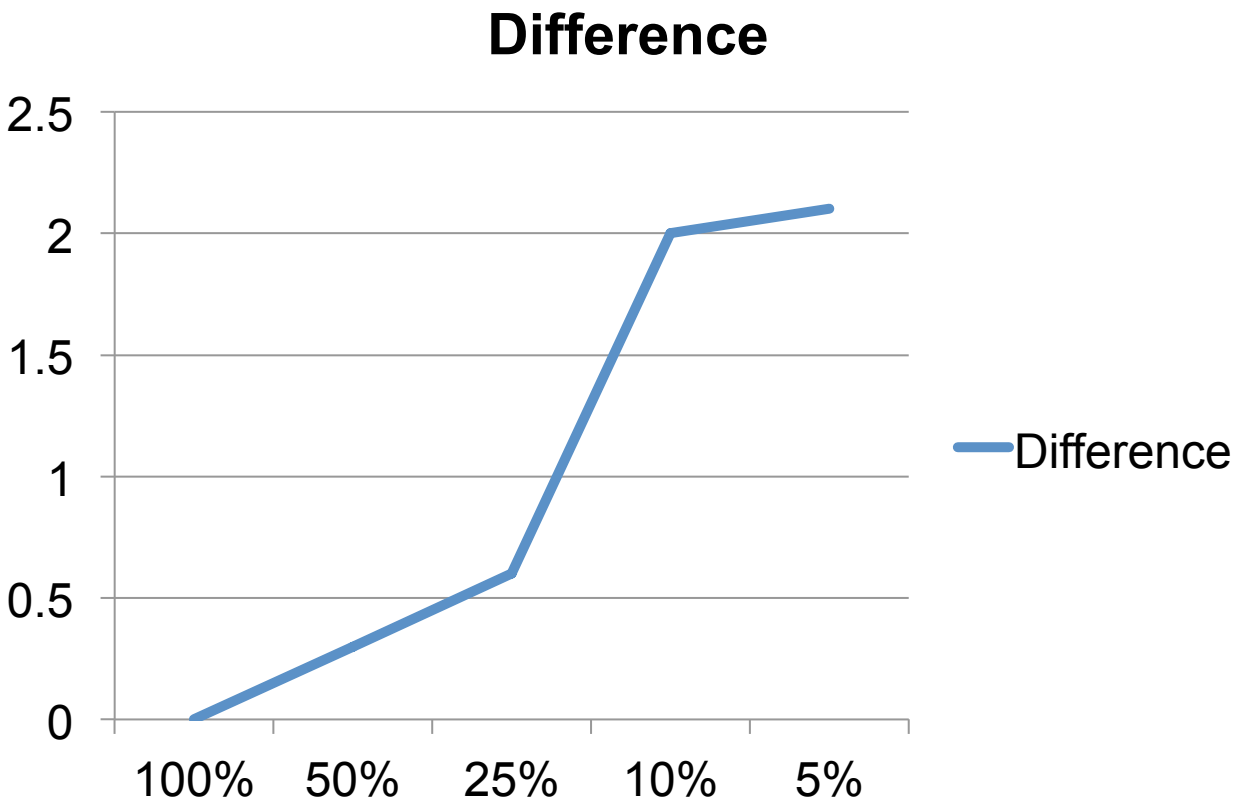
Noisy CIFAR-100



ImageNet Subset



ImageNet Subset



Nesterov Momentum

Also called accelerated gradient

Requires a convex function

Ignore

Requires a smooth function

Cross entropy is smooth wrt **w**

Hinge loss is not

Not a stochastic algorithm

Ignore

Outline

- Gradient Descent with Nesterov Momentum
- **AdaGrad**
- AdaDelta
- Adam

Online Learning

Unknown sequence of functions $f_1, f_2, \ldots, f_T$

Update the estimate $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$

Regret $R(T) = \sum_t f_t(\mathbf{x}_t) - f_t(\mathbf{x}^*)$

Closely related to stochastic optimization

Can we use stochastic subgradient descent?

Zinkevich, 2003

Online Learning

Compute subgradient $\mathbf{g}_t$ of f_t at $\mathbf{x}_t$

Update $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta/\sqrt{t}\ \mathbf{g}_t$

Provably, $R(T) = O(\sqrt{T})$

As $T \to \infty$, $R(T)/T$ tends to 0

Can we do better? Only with additional assumptions

Example

$f_t(\mathbf{x}) = \max_i \{\mathbf{a}_{ti}^T \mathbf{x} + b_{ti}\}$

Say the d-th element of $\mathbf{x}^*$ is $x^*(d)$

Scale down $a_{ti}(d)$ by κ

Algorithm slows down

Need to scale up $x^*(d)$ by κ

d-th element of the subgradient is scaled down by κ

Second-Order Method

Use Newton's method

Invariant to affine transformations

Needs the computation of Hessian

Could be prohibitively expensive

Use adaptive gradients

AdaGrad

Compute subgradient $\mathbf{g}_t \in R^D$ of f_t at $\mathbf{x}_t \in R^D$

Compute $\mathbf{s}_t = \sum_\tau (\mathbf{g}_\tau)^2$ (element-wise multiplication)

Update $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta\, \mathbf{g}_t/(\sqrt{\mathbf{s}_t} + \varepsilon)$ (element-wise division)

Invariant to scaling O(log D/√D) speed-up

Duchi, Hazan and Singer, 2011

Outline

- Gradient Descent with Nesterov Momentum
- AdaGrad
- **AdaDelta**
- Adam

Zeiler, 2012 (unpublished arXiv report)

AdaDelta

Two main modifications to AdaGrad

Modification I

AdaGrad compute $\mathbf{s}_t = \sum_\tau (\mathbf{s}_\tau)^2 \quad = \mathbf{s}_{t-1} + \mathbf{g}_t^2$

AdaDelta weighs recent iterations more

$\mathbf{s}_t = \rho \mathbf{s}_{t-1} + (1-\rho)\, \mathbf{g}_t^2 \qquad 0 < \rho < 1$

AdaDelta

Two main modifications to AdaGrad

Modification II

Define update at time t as $\Delta \mathbf{x}_t$

$\mathbf{u}_t = \mathbf{u}_{t-1} + (\Delta \mathbf{x}_t)^2$ $\qquad$ $\Delta \mathbf{x}_t = -(\sqrt{\mathbf{u}_{t-1}} + \varepsilon)/(\sqrt{\mathbf{s}_t} + \varepsilon)\ \mathbf{g}_t$

Adaptive Delta

Offers no guarantees even in the convex case

Outline

- Gradient Descent with Nesterov Momentum
- AdaGrad
- AdaDelta
- **Adam (Adaptive Moments)**

Kingma and Ba, 2015

Adam

Compute subgradient $\mathbf{g}_t \in R^D$ of f_t at $\mathbf{x}_t \in R^D$

Compute 1st moment $\mathbf{m}_t = \beta_1 \mathbf{m}_{t-1} + (1-\beta_1)\, \mathbf{g}_t$

Similar intuition to momentum

Compute 2nd moment $\mathbf{s}_t = \beta_2 \mathbf{s}_{t-1} + (1-\beta_2)\, (\mathbf{g}_t)^2$

Similar intuition to AdaDelta

Adam

Compute unbiased estimates of moments

$\mathbf{m'}_t = \mathbf{m}_t / (1-(\beta_1)^t)$

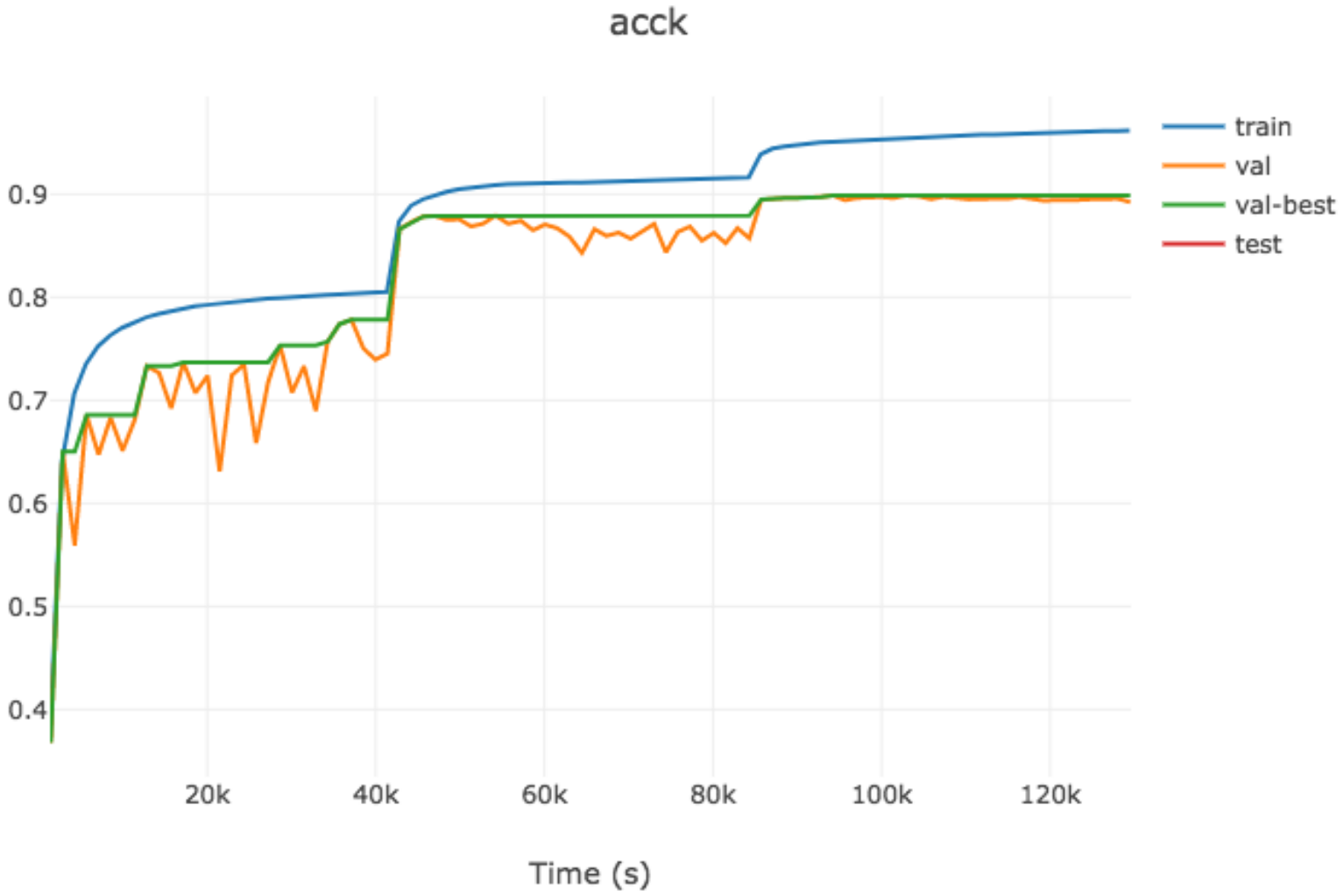
$\mathbf{s'}_t = \mathbf{s}_t / (1-(\beta_2)^t)$

Update $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta\, \mathbf{m'}_t/(\sqrt{\mathbf{s'}_t} + \varepsilon)$

Same guarantees as AdaGrad were claimed

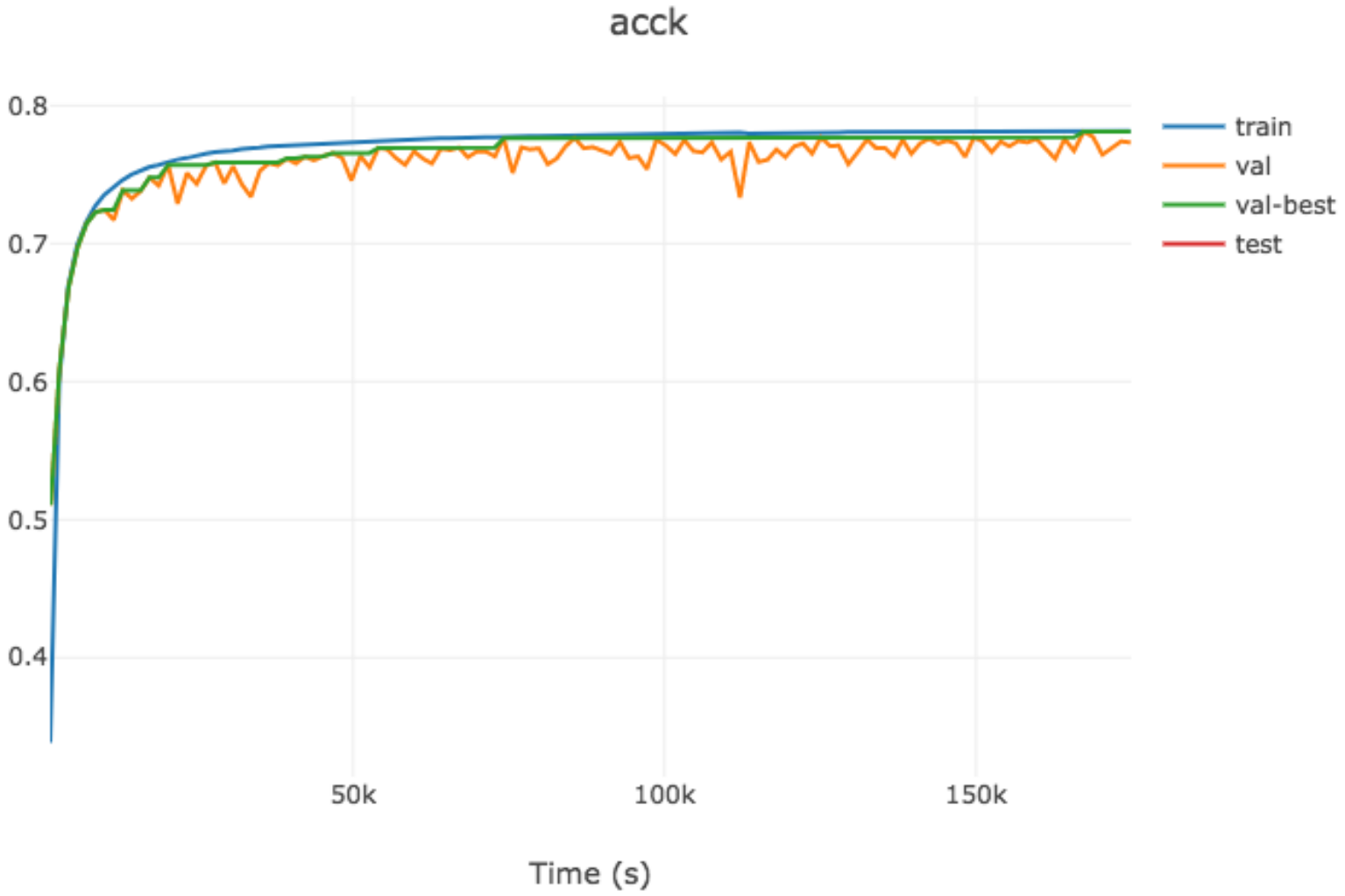
ImageNet with ResNet-18

SGD with Nesterov Momentum



ImageNet with ResNet-18

Adam



Adaptive Gradients

Hand-designed example of linearly separable data

Adaptive gradients provide zero test accuracy

Empirical comparison on state-of-the-art models

Adaptive gradients may do better in training

But they generalize (significantly) worse

Wilson, Roelofs, Stern, Srebro and Recht, 2017

Questions?