Adam: A Method for Stochastic Optimization

Diederick P. Kingma, Jimmy Lei Bai

Jaya Narasimhan

February 10, 2016

Outline

Introduction

- A method for stochastic, gradient-based optimization
- Name is from "adaptive moment estimation"
- Uses the first and second moment estimates of the gradients to compute the learning rates of each parameter
- Suitable for huge data sets and/or high-dimensional parameter spaces.

Adam's Advantages

- Magnitude of parameter updates are invariant to re-scaling of the gradient.
- Does not require a stationary objective
- Works with sparse gradients
- Naturally performs a form of step size annealing

Algorithm

Objective: Want to minimize $\mathbb{E}[f(\theta)]$.

- $f(\theta)$: A noisy objective function, differentiable w.r.t parameters θ
- α: stepsize
- $\beta_1, \beta_2 \in [0, 1]$: Exponential decay rates for the moment estimates
- θ_0: Initial parameter vector
- $f_t(\theta)$ as the realization of the stochastic function at time t
- $g_t = \nabla_\theta f_t(\theta)$, the gradient of $f_t(\theta)$ w.r.t θ evaluated at time t

Algorithm

- $m_0 \leftarrow 0$: First moment estimate
- $v_0 \leftarrow 0$: Second moment estimate
- $t \leftarrow 0$: Time step

while θ_t not converged do:

- $t \leftarrow t + 1$
- $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$: Compute the gradient
- $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$: Update the first moment
- $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$: Update the second moment
- $\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$: Correct first moment for bias
- $\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$: Correct second moment for bias
- $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$: Update parameters

return θ_t

Algorithm

while θ_t not converged do:

- $t \leftarrow t + 1$
- $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$: Compute the gradient
- $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$: Update the first moment
- $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$: Update the second moment
- $\alpha_t \leftarrow \alpha \cdot \sqrt{1 - \beta_2^t}/(1 - \beta_1^t)$
- $\theta_t \leftarrow \theta_{t-1} - \alpha_t \cdot m_t/(\sqrt{v_t} + \epsilon)$

return θ_t

Stepsize

Assuming $\epsilon = 0$, effective stepsize at time t is $\Delta t \leq \alpha \cdot \hat{m}_t / \sqrt{\hat{v}_t}$

Two Upper Bounds

- $|\Delta t| \leq \alpha(1 - \beta_1)/\sqrt{1 - \beta_2}$ when $(1 - \beta_1) > \sqrt{1 - \beta_2}$
- $|\Delta t| \leq \alpha$ otherwise

Other Notes

- When close to an optimum, $\hat{m}_t / \sqrt{\hat{v}_t}$ is small
- Stepsize is the same if the gradients are rescaled: $(c \cdot \hat{m}_t)/\sqrt{c^2 \cdot \hat{v}_t} = \hat{m}_t / \sqrt{\hat{v}_t}$

Related Work

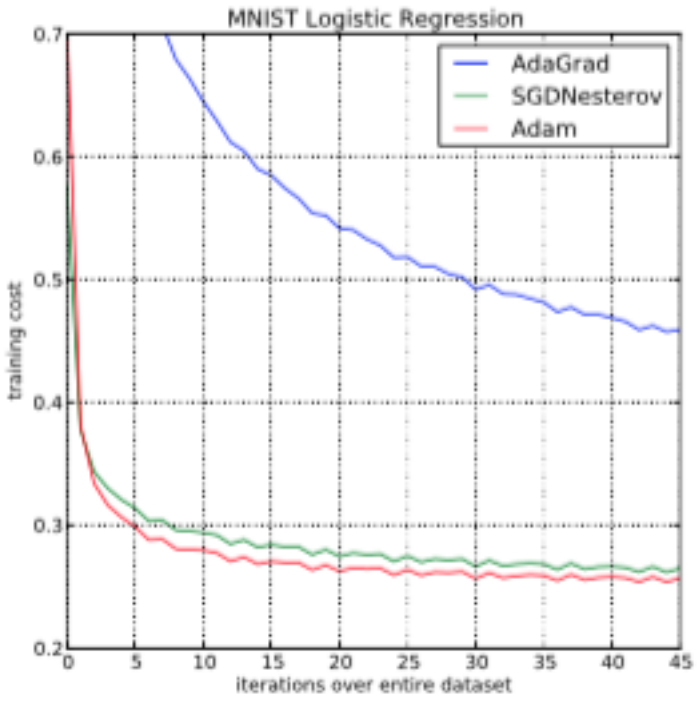
RMSProp

- Also used on stochastic objectives
- Generates parameter updates using momentum
- No bias correction

AdaGrad

- Corresponds to a version of Adam with $\beta_1 = 0$ and infinitesimal $(1 - \beta_2)$
- Replacement of stepsize α with $\alpha_t = \alpha \cdot t^{-\frac{1}{2}}$

Logistic Regression



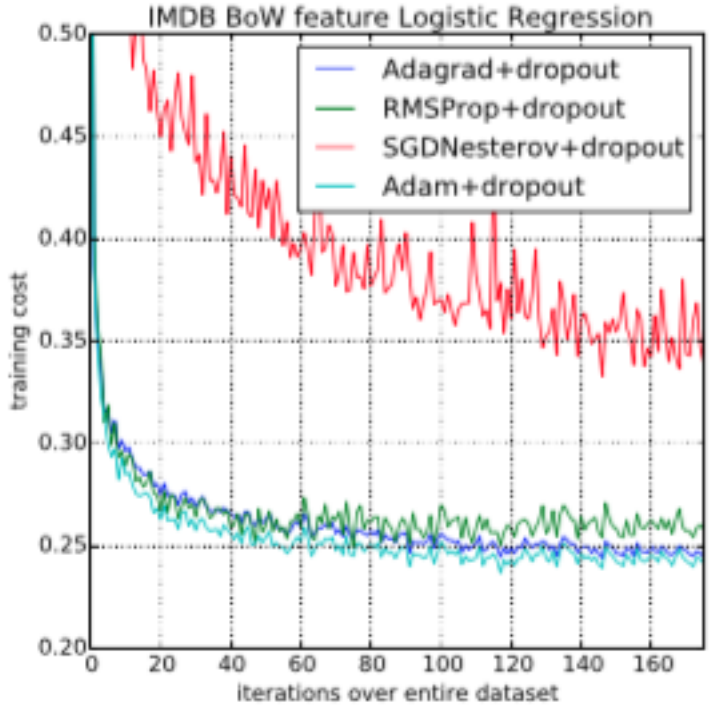
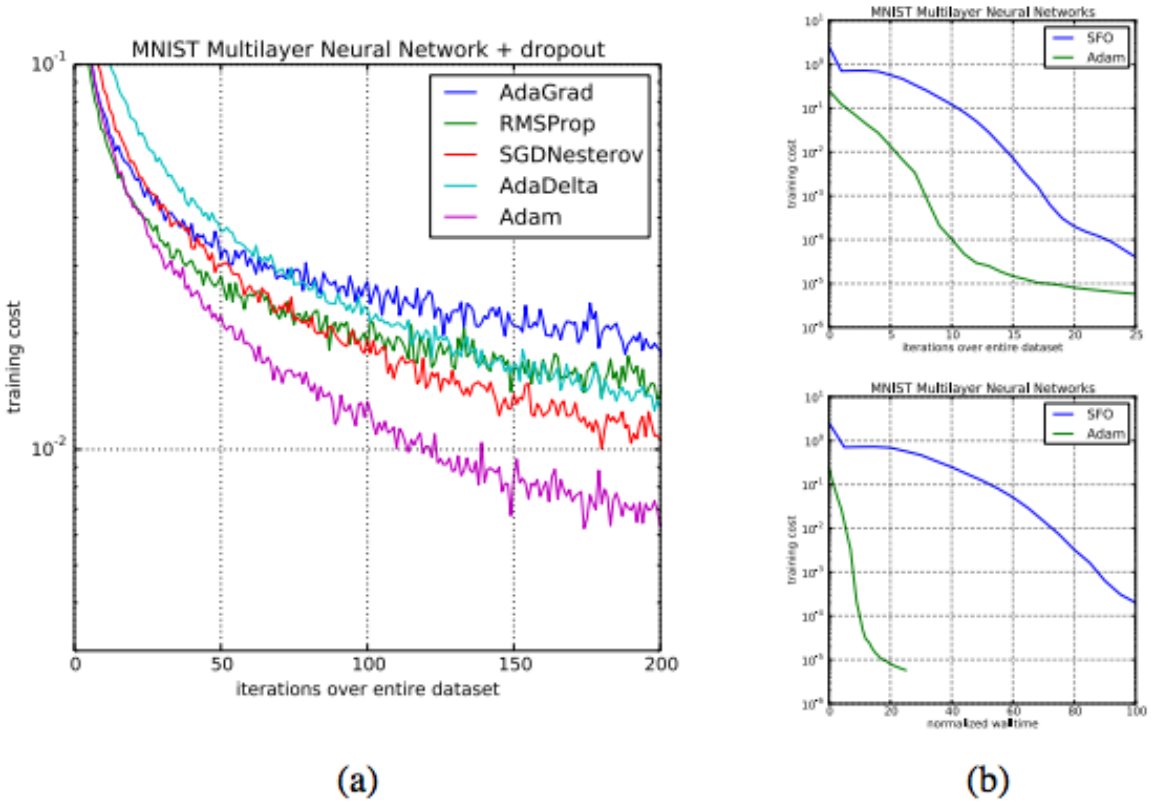


Figure 1: Logistic regression training negative log likelihood on MNIST images and IMDB movie reviews with 10,000 bag-of-words (BoW) feature vectors.

Muti-layer Neural Nets



(a) (b)

Figure 2: Training of multilayer neural networks on MNIST images. (a) Neural networks using dropout stochastic regularization. (b) Neural networks with deterministic cost function. We compare with the sum-of-functions (SFO) optimizer (Sohl-Dickstein et al., 2014)

Convolutional Neural Nets

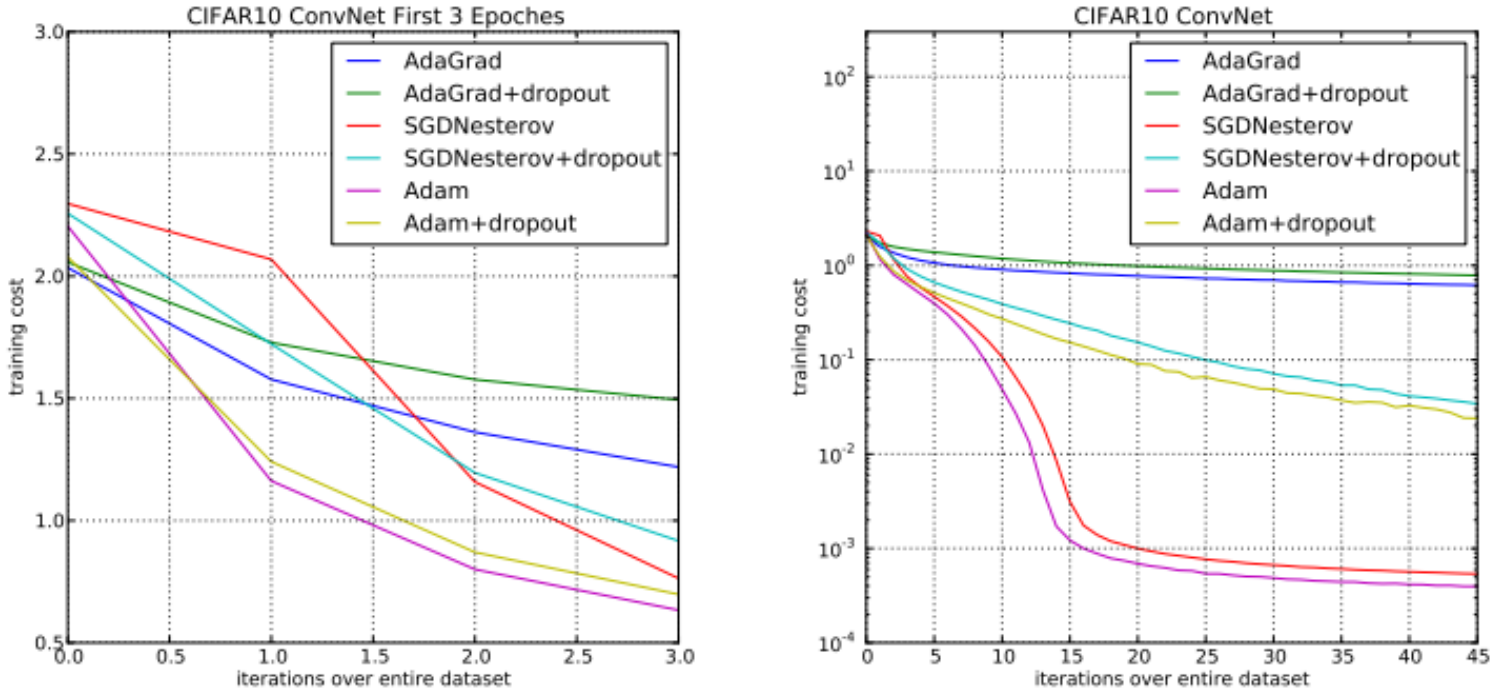
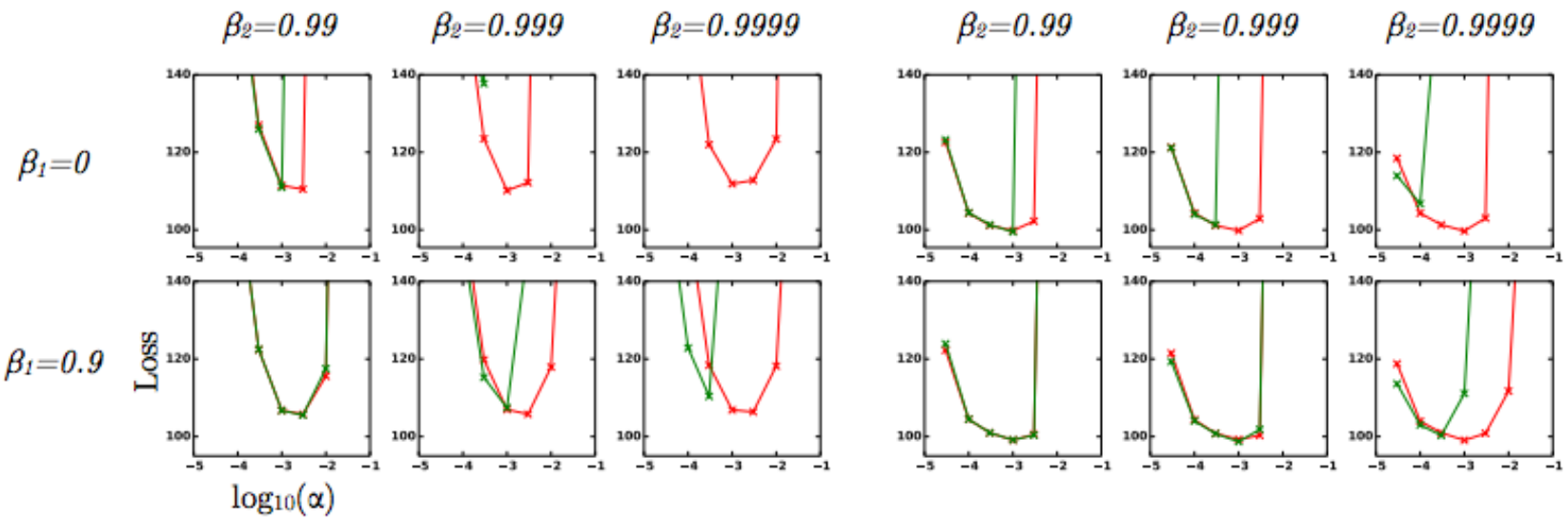


Figure 3: Convolutional neural networks training cost. (left) Training cost for the first three epochs. (right) Training cost over 45 epochs. CIFAR-10 with c64-c64-c128-1000 architecture.

Evaluating the Bias Constant



(a) after 10 epochs

(b) after 100 epochs

Figure 4: Effect of bias-correction terms (red line) versus no bias correction terms (green line) after 10 epochs (left) and 100 epochs (right) on the loss (y-axes) when learning a Variational Auto-Encoder (VAE) (Kingma & Welling, 2013), for different settings of stepsize α (x-axes) and hyper-parameters β_1 and β_2.

AdaMax

Instead of scaling the gradient inversely proportional to the L2 norm, AdaMax uses the infinity norm.

- $m_0 \leftarrow 0$: First moment estimate
- $u_0 \leftarrow 0$: Exponentially weighted infinity norm
- $t \leftarrow 0$: Time step

while θ_t not converged do:

- $t \leftarrow t + 1$
- $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$: Compute the gradient
- $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$: Update the first moment
- $u_t \leftarrow max(\beta_2 \cdot u_{t-1}, |g_t|)$: Update infinity norm
- $\theta_t \leftarrow \theta_{t-1} - (\alpha/(1 - \beta_1^t)) \cdot m_t/u_t$: Update parameters

return θ_t

Summary

- Adam is aimed towards machine learning problems with large datasets and/or high dimensional parameter spaces.
- It can work with sparse gradients and stochastic objectives.
- The method is straightforward and requires little memory.
- Adam is well-suited to a wide range of non-convex optimization problems.