ICLR 2015

ADAM: A Method for Stochastic Optimization

Diederik P. Kingma *
University of Amsterdam, OpenAI

Jimmy Lei Ba *
University of Toronto

* Equal contribution

What is optimization?

The process of finding the best solution from all possible solutions

What is optimization in machine learning?

The process of adjusting model parameters to <u>minimize the objective function</u> (loss function) and improve model performance.

Line-up

1. Optimization algorithms
2. Related work
3. Adam
4. Convergence analysis
5. Codes
6. Experiments
7. Limitations

Line-up

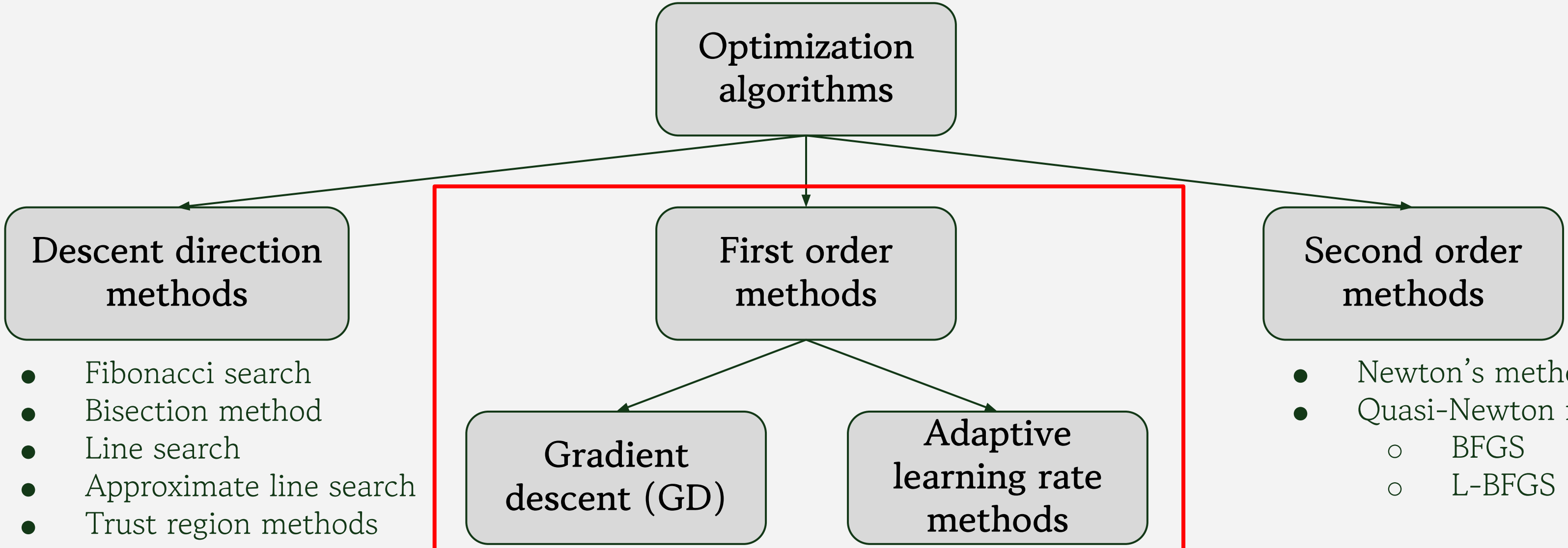
1	2	3	4	5	6	7
Optimization algorithms	Related work	Adam	Convergence analysis	Codes	Experiments	Limitations

Big picture!

Optimization algorithms

- Descent direction methods
  - Fibonacci search
  - Bisection method
  - Line search
  - Approximate line search
  - Trust region methods
- First order methods
  - Gradient descent (GD)
    - Batch GD
    - Mini-batch GD
    - Stochastic GD
    - GD with momentum
    - Conjugate GD
  - Adaptive learning rate methods
    - RMSProp
    - Adam
    - Adagrad
    - Adadelta
- Second order methods
  - Newton's method
  - Quasi-Newton methods
    - BFGS
    - L-BFGS

Big picture!



Descent direction methods

- Fibonacci search
- Bisection method
- Line search
- Approximate line search
- Trust region methods

Gradient descent (GD)

- Batch GD
- Mini-batch GD
- Stochastic GD
- GD with momentum
- Conjugate GD

Adaptive learning rate methods

- RMSProp
- Adam
- Adagrad
- Adadelta

Second order methods

- Newton's method
- Quasi-Newton methods
  - BFGS
  - L-BFGS

Line-up

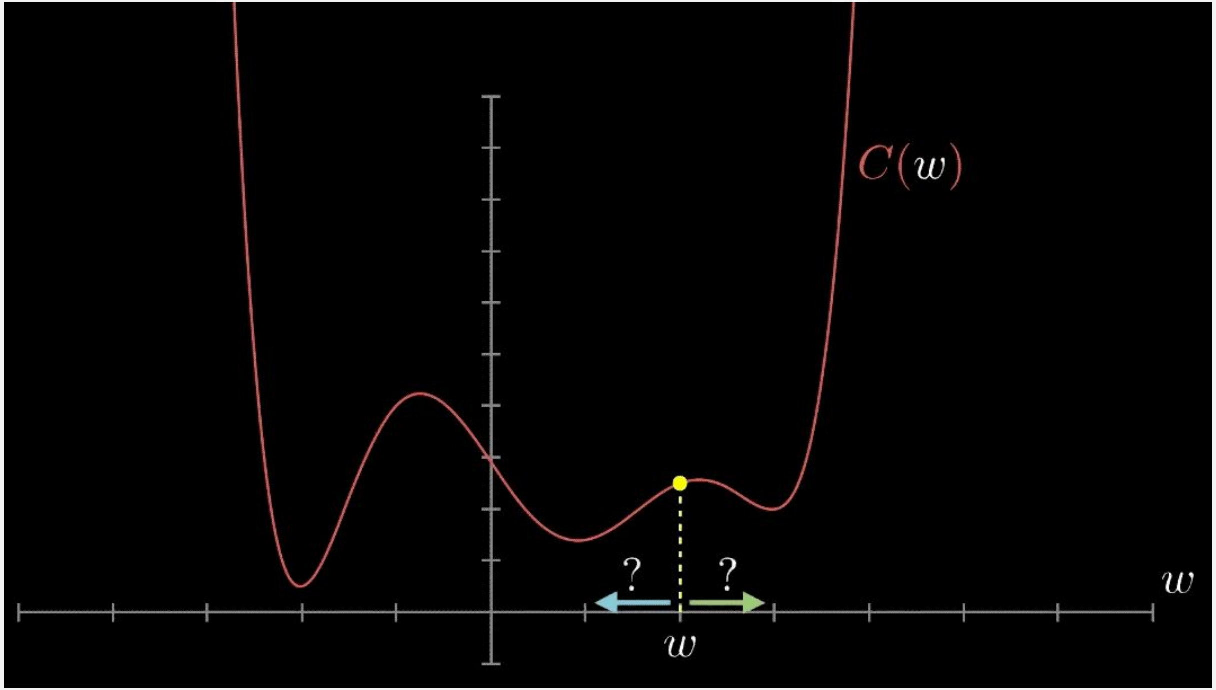
1. Optimization algorithms
2. Related work
3. Adam
4. Convergence analysis
5. Codes
6. Experiments
7. Limitations

Recap: Gradient descent

$$\mathrm{w} = \mathrm{w} - \alpha \frac{\partial}{\partial w}[C(w)]$$

w: model parameter

$C(w)$: objective function



Credits: https://mlfromscratch.com/optimizers-explained

Gradient descent variants

Batch gradient descent

$$X_{(n\times m)} = \left[x^{(1)}\ x^{(2)}\ \dots\ x^{(m)}\right]$$

$$Y_{(1\times m)} = \left[y^{(1)}\ y^{(2)}\ \dots\ y^{(m)}\right]$$

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)})$$

$$\mathrm{w} = \mathrm{w} - \alpha \frac{\partial}{\partial w}[J(w,b)]$$

$$b = b - \alpha \frac{\partial}{\partial b}[J(w,b)]$$

w: weight parameter

b : bias parameter

$J(w,b)$: loss function

- All training examples are processed at once
- High computational cost (utilizes vectorization w/o for loops)
- Slow convergence (1 parameters per epoch)

Mini-batch gradient descent

$$X_{(n\times m)} = \big[\underbrace{x^{(1)}\ x^{(2)}\ \dots\ x^{(b)}}_{X^{\{1\}}_{(n\times b)}} \,\big|\, \underbrace{x^{(b+1)}\ x^{(b+2)}\ \dots x^{(2b)}}_{X^{\{2\}}_{(n\times b)}} \,\big|\, \dots\big]$$

$$Y_{(1\times m)} = \big[\underbrace{y^{(1)}\ y^{(2)}\ \dots\ y^{(b)}}_{Y^{\{1\}}_{(1\times b)}} \,\big|\, \underbrace{y^{(b+1)}\ y^{(b+2)}\ \dots y^{(2b)}}_{Y^{\{2\}}_{(1\times b)}} \,\big|\, \dots\big]$$

$$J^{\{t\}}(w,b) = \frac{1}{b}\sum_{i=1}^{b} L(\hat{y}^{(i)}, y^{(i)})$$

$$\mathrm{w} = \mathrm{w} - \alpha \frac{\partial}{\partial w}[J^{\{t\}}(w,b)]$$

$$b = b - \alpha \frac{\partial}{\partial b}[J^{\{t\}}(w,b)]$$

$X^{\{t\}}$: t^th mini-batch of training examples
$Y^{\{t\}}$: t^th mini-batch of targets

- All training examples are processed in mini-batches
- Medium computational cost (reduced vectorization w/ for loops)
- Fastest learning (parameters get updated ceil(m/b) number of steps per epoch)

Stochastic gradient descent

$$X_{(n\times m)} = [\underbrace{x^{(1)}}_{X^{\{1\}}_{(n\times 1)}}\ x^{(2)}\ \dots\ \underbrace{x^{(m)}}_{X^{\{m\}}_{(n\times 1)}}]$$

$$Y_{(1\times m)} = [\underbrace{y^{(1)}}_{Y^{\{1\}}_{(1\times 1)}}\ y^{(2)}\ \dots\ \underbrace{y^{(m)}}_{Y^{\{m\}}_{(1\times 1)}}]$$

$b = 1$: here mini-batch size is one

$X^{\{t\}}$: t^{th} mini-batch of training examples

$Y^{\{t\}}$: t^{th} mini-batch of targets

$$J^{\{t\}}(w,b) = \frac{1}{b}\sum_{i=1}^{b} L(\hat{y}^{(i)}, y^{(i)})$$

$$\mathrm{w} = \mathrm{w} - \alpha \frac{\partial}{\partial w}[J^{\{t\}}(w,b)]$$

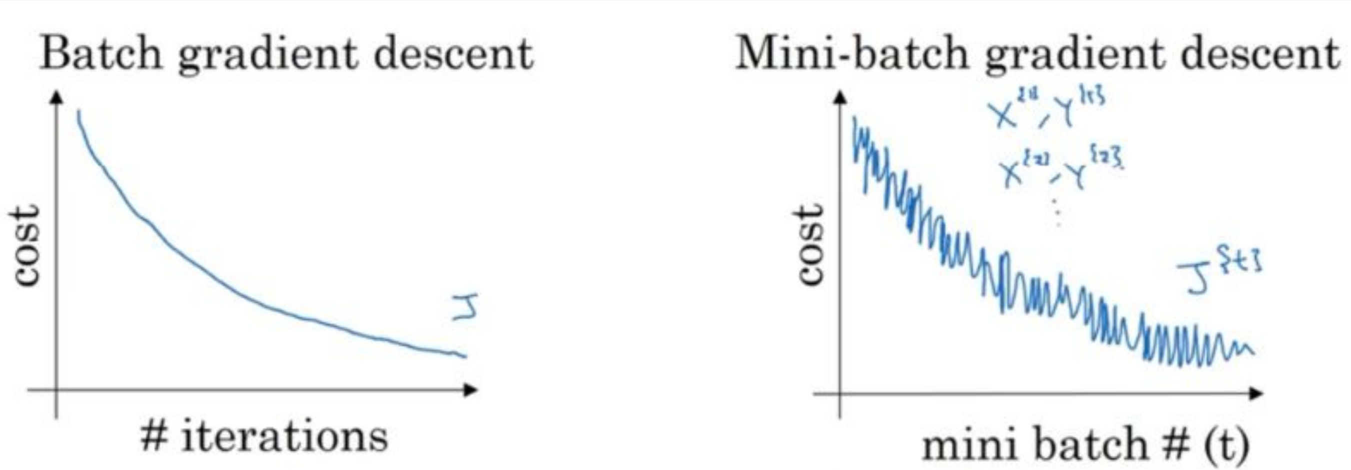
$$b = b - \alpha \frac{\partial}{\partial b}[J^{\{t\}}(w,b)]$$

- All training examples are processed one sample at a time
- Low computational cost per update
- Fast but oscillates, loses speedup from vectorization (parameters get updated per sample per epoch)

Training

Low noise in updates

More noise in updates
(less noise than SGD)



Credits: Deep learning specialization by Prof. Andrew Ng, Coursera

Stochasticity

Randomness or unpredictability in a system or process

E.g.,

- Mini-batch gradient descent
- SGD

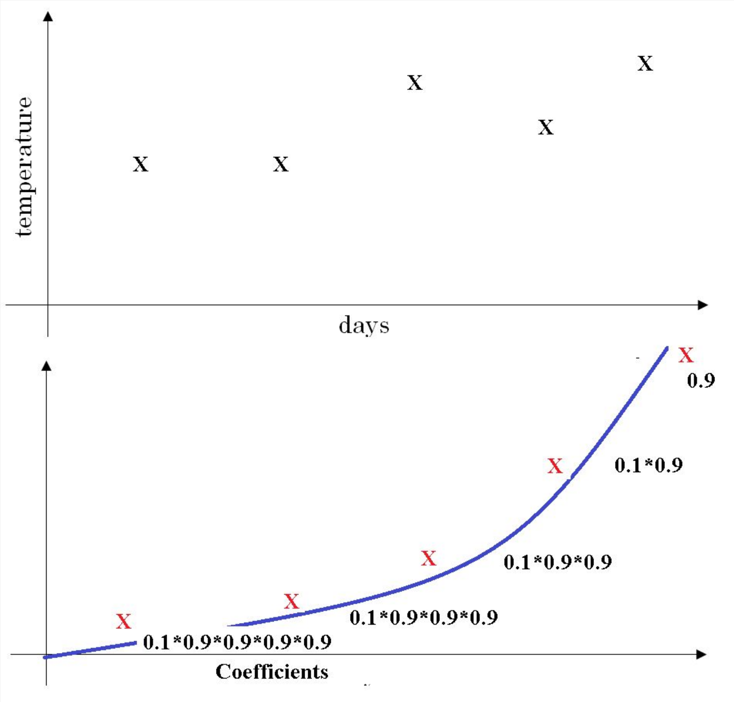
Stochastic objective functions

EWA (Exponentially Weighted Average)

$v_t = \beta v_{t-1} + (1-\beta)\theta_t$ β is usually a decimal close to 1 (e.g. 0.9, 0.99, etc.)

$$v_0 = 0$$
$$v_1 = (1-\beta)\theta_1$$
$$v_2 = \beta v_1 + (1-\beta)\theta_2 \Leftrightarrow v_2 = \beta((1-\beta)\theta_1) + (1-\beta)\theta_2$$
$$v_3 = \beta v_2 + (1-\beta)\theta_3 \Leftrightarrow v_3 = \beta(\beta((1-\beta)\theta_1) + (1-\beta)\theta_2) + (1-\beta)\theta_3$$
$$\Updownarrow$$
$$v_3 = \beta^2(1-\beta)\theta_1 + \beta(1-\beta)\theta_2 + (1-\beta)\theta_3$$

Why ‘exponential’?

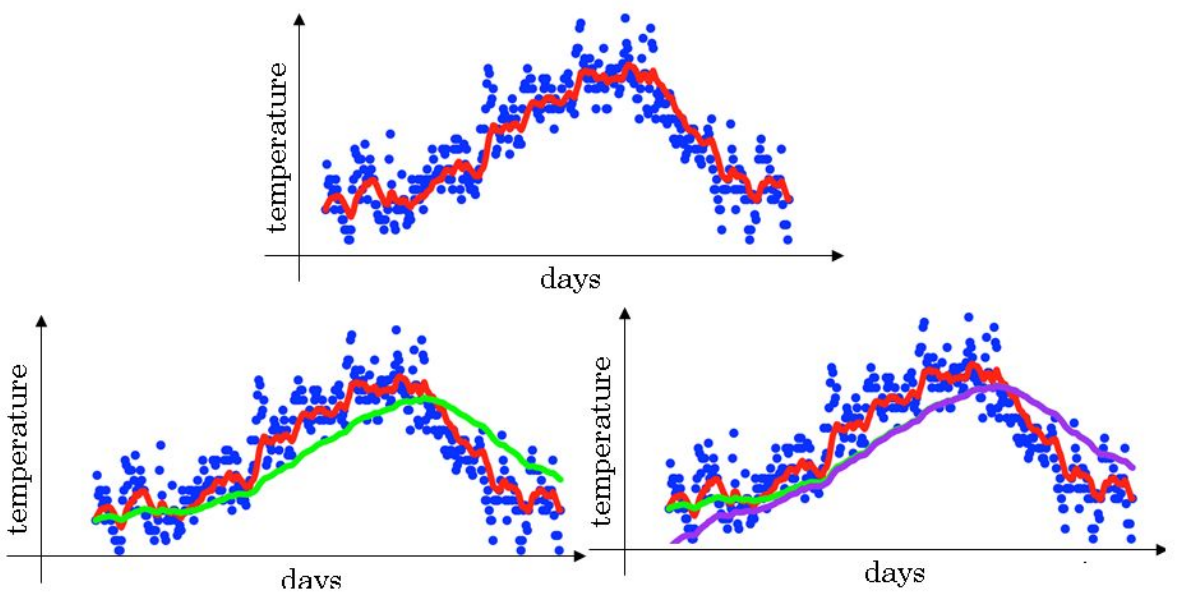


Credits: upscfever.com

$\beta = 0.9$
$(1-\beta) = 0.1$

- Places greater weights on the most recent data
- The weighting for older data points decrease exponentially
- Computationally efficient
- Saves storage (overrides)

Bias correction



Credits: Deep learning specialization by Prof. Andrew Ng, Coursera

$$v_t = \beta v_{t-1} + (1-\beta)\theta_t$$

- Initially biased towards initialization
- Bias corrected term:

$$\hat{v}_t = \frac{v_t}{1-\beta^t}$$

- Green plot: bias not corrected
- Purple plot: bias corrected

Gradient descent with momentum

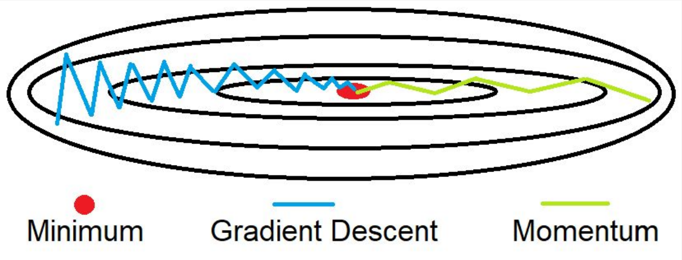
First moment estimate/estimate for mean of gradients

$$m_{t,w} = \beta_1 m_{t-1,w} + (1-\beta_1) g_{t,w}$$

$$m_{t,b} = \beta_1 m_{t-1,b} + (1-\beta_1) g_{t,b}$$

Bias correction: $\hat{m}_{t,w} = \dfrac{m_{t,w}}{1-\beta_1^t}$ $\hat{m}_{t,b} = \dfrac{m_{t,b}}{1-\beta_1^t}$

Parameter updates: $W = W - \alpha \hat{m}_{t,w}$ $b = b - \alpha \hat{m}_{t,b}$



Credits: andreaperlato.com

Gradient w.r.t. Stochastic objective at timestep t:

$$g_{t,w} = \frac{\partial}{\partial w}[J^{\{t\}}(w,b)]$$

$$g_{t,b} = \frac{\partial}{\partial b}[J^{\{t\}}(w,b)]$$

- Minimizes the vertical direction gradient component

Adagrad (Adaptive gradient)

$$g_t = \nabla_\theta J(\theta_t)$$

Accumulates the squares of past gradients:

$$G_t = G_{t-1} + g_t^2$$

Update rule for parameters:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_t + \epsilon}} \odot g_t$$

Adaptive learning rate methods

RMSProp

Second moment estimate/estimate for variance of gradients

$$v_{t,w} = \beta_2 v_{t-1,w} + (1-\beta_2) g_{t,w}^2$$

$$v_{t,b} = \beta_2 v_{t-1,b} + (1-\beta_2) g_{t,b}^2$$

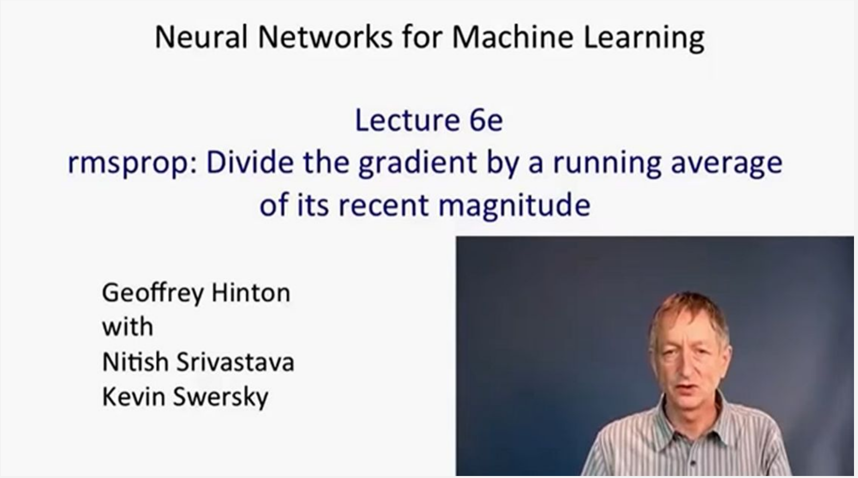
Bias correction:

$$\hat{v}_{t,w} = \frac{v_{t,w}}{1-\beta_2^t} \qquad \hat{v}_{t,b} = \frac{v_{t,b}}{1-\beta_2^t}$$

Parameter updates:

$$W = W - \alpha \frac{g_{t,w}}{\sqrt{\hat{v}_{t,w} + \varepsilon}}$$

$$b = b - \alpha \frac{g_{t,b}}{\sqrt{\hat{v}_{t,b} + \varepsilon}}$$



Root
Mean
Square
Propagation

} RMSProp

- Minimizes variance in gradients

Line-up

1. Optimization algorithms
2. Related work
3. Adam
4. Convergence analysis
5. Codes
6. Experiments
7. Limitations

Adam (Adaptive moment estimation)

1st moment estimate (momentum term):

$$m_{t,w} = \beta_1 m_{t-1,w} + (1-\beta_1) g_{t,w}$$
$$\hat{m}_{t,w} = \frac{m_{t,w}}{1-\beta_1^t}$$

$$m_{t,b} = \beta_1 m_{t-1,b} + (1-\beta_1) g_{t,b}$$
$$\hat{m}_{t,b} = \frac{m_{t,b}}{1-\beta_1^t}$$

2nd moment estimate (RMSProp term):

$$v_{t,w} = \beta_2 v_{t-1,w} + (1-\beta_2) g_{t,w}^2$$
$$\hat{v}_{t,w} = \frac{v_{t,w}}{1-\beta_2^t}$$

$$v_{t,b} = \beta_2 v_{t-1,b} + (1-\beta_2) g_{t,b}^2$$
$$\hat{v}_{t,b} = \frac{v_{t,b}}{1-\beta_2^t}$$

Parameter updates:

$$W = W - \alpha \frac{\hat{m}_{t,w}}{\sqrt{\hat{v}_{t,w}} + \varepsilon}$$

$$b = b - \alpha \frac{\hat{m}_{t,b}}{\sqrt{\hat{v}_{t,b}} + \varepsilon}$$

Algorithm

$\beta 1$: Exponential decay rate parameter for past gradient estimates

$\beta 2$: Exponential decay rate parameter for past squared gradient estimates

Algorithm 1: *Adam*, our proposed algorithm for stochastic optimization. See section 2 for details, and for a slightly more efficient (but less clear) order of computation. g_t^2 indicates the elementwise square $g_t \odot g_t$. Good default settings for the tested machine learning problems are $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. All operations on vectors are element-wise. With β_1^t and β_2^t we denote β_1 and β_2 to the power t.

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0: Initial parameter vector
  $m_0 \leftarrow 0$ (Initialize 1st moment vector)
  $v_0 \leftarrow 0$ (Initialize 2nd moment vector)
  $t \leftarrow 0$ (Initialize timestep)
  while θ_t not converged **do**
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
    $\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
    $\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
    $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
  end while
  return θ_t (Resulting parameters)

Adam's update rule

$$\Delta_t = \alpha \cdot \widehat{m}_t / \sqrt{\widehat{v}_t}$$

case $(1 - \beta_1) > \sqrt{1 - \beta_2}$ $\quad |\Delta_t| \leq \alpha \cdot (1 - \beta_1)/\sqrt{1 - \beta_2}$

If $\underbrace{\text{gradient}}_{g_t}$ has been zero at all time steps except current step,

$$m_t \approx (1-\beta_1) g_t \Longrightarrow \hat{m}_t \approx \frac{(1-\beta_1) g_t}{(1-\beta_1^t)}$$

$$V_t \approx (1-\beta_2) g_t^2 \Longrightarrow \hat{V}_t \simeq \frac{(1-\beta_2) g_t^2}{(1-\beta_2^t)}$$

when $t \uparrow$ $\beta_1^t, \beta_2^t \simeq 0$

$\hat{m}_t \simeq m_t$

$\hat{V}_t \simeq V_t$

$$\Longrightarrow \left| \frac{\hat{m}_t}{\sqrt{\hat{V}_t}} \right| = \left| \frac{(1-\beta_1) g_t}{\sqrt{(1-\beta_2) g_t^2}} \right| = \frac{1-\beta_1}{\sqrt{1-\beta_2}}$$

$(1-\beta_1) > \sqrt{1-\beta_2}$

$$|\Delta_t| \leq \alpha \cdot \frac{(1-\beta_1)}{\sqrt{1-\beta_2}} \lesssim \alpha \cdot 1$$

Tighter upper bound (for sparsity)

Adam's update rule

$$\Delta_t = \alpha \cdot \widehat{m}_t / \sqrt{\widehat{v}_t}$$

Common case: $|\Delta_t| \leq \alpha$

SNR (signal to noise ratio)

Jensen's inequality

For any stochastic gradient, g_t:

1st moment estimate $\hat{m}_t$

2nd moment est v_t

$$\frac{E[g_t]}{\sqrt{E[g_t^2]}} \leq 1$$

$\therefore$ General case $|\Delta_t| \leq \alpha \cdot 1$

- Adam is bounded
- Does not grow exponentially

Line-up

1. Optimization algorithms
2. Related work
3. Adam
4. Convergence analysis
5. Codes
6. Experiments
7. Limitations

Line-up

1. Optimization algorithms
2. Related work
3. Adam
4. Convergence analysis
5. Codes
6. Experiments
7. Limitations

PyTorch

Adam

CLASS torch.optim.Adam(*params*, *lr=0.001*, *betas=(0.9, 0.999)*, *eps=1e-08*, *weight_decay=0*, *amsgrad=False*, ***, *foreach=None*, *maximize=False*, *capturable=False*, *differentiable=False*, *fused=None*) [SOURCE]

Parameters

- **params** (*iterable*) – iterable of parameters or named_parameters to optimize or iterable of dicts defining parameter groups. When using named_parameters, all parameters in all groups should be named
- **lr** (*float*, *Tensor*, *optional*) – learning rate (default: 1e-3). A tensor LR is not yet supported for all our implementations. Please use a float LR if you are not also specifying fused=True or capturable=True.
- **betas** (*Tuple*[*float*, *float*], *optional*) – coefficients used for computing running averages of gradient and its square (default: (0.9, 0.999))
- **eps** (*float*, *optional*) – term added to the denominator to improve numerical stability (default: 1e-8)
- **weight_decay** (*float*, *optional*) – weight decay (L2 penalty) (default: 0)

Line-up

1. Optimization algorithms
2. Related work
3. Adam
4. Convergence analysis
5. Codes
6. Experiments
7. Limitations

Experiments

Logistic regression (convex)

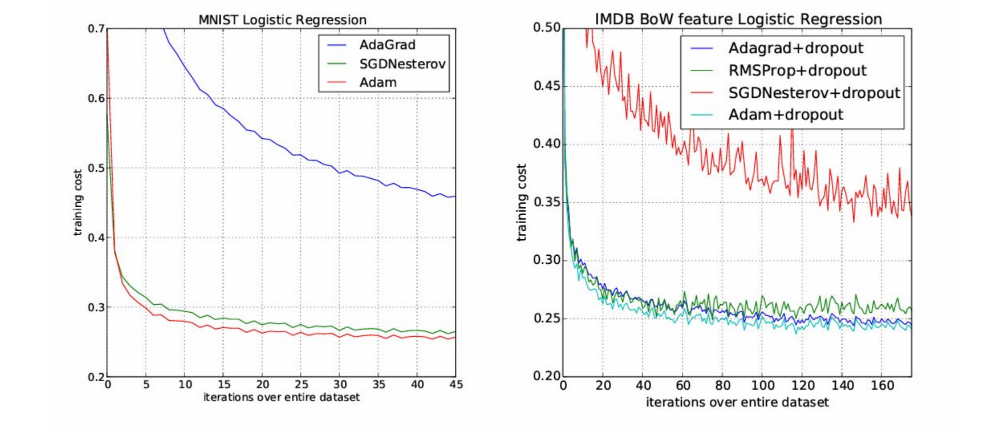


Figure 1: Logistic regression training negative log likelihood on MNIST images and IMDB movie reviews with 10,000 bag-of-words (BoW) feature vectors.

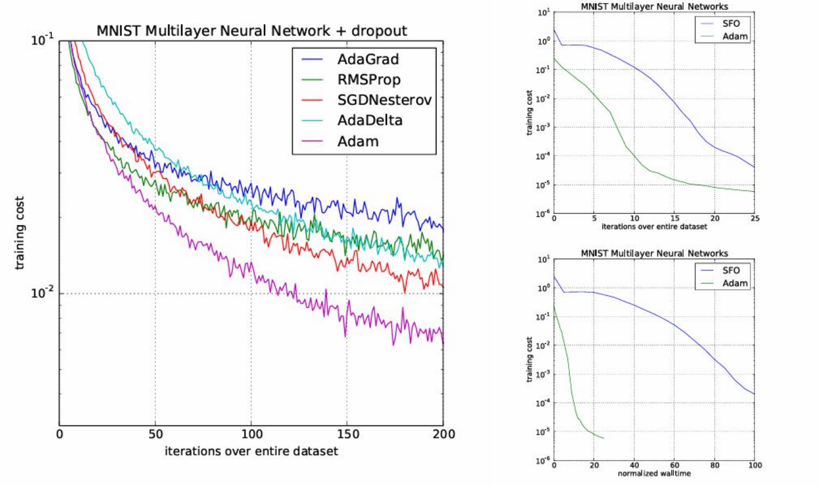
Adam's learning rate decay:

$$\alpha_t = \frac{\alpha}{\sqrt{t}}$$

- The 10,000 dimension BoW feature vector for each review is highly sparse
- Adam can take advantage of sparse features and obtain faster convergence

Experiments

Multi-layer neural network (non-convex)



(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)

- A neural network model with:
  - 2 fully connected
  - Hidden layers with 1000 hidden units each
  - ReLU activation
- Mini-batch size of 128
- L2 weight decay on the parameters to prevent over-fitting
- Normalized walltime: actual elapsed time (walltime) adjusted or normalized relative to a reference standard

Experiments

CNN (non-convex)

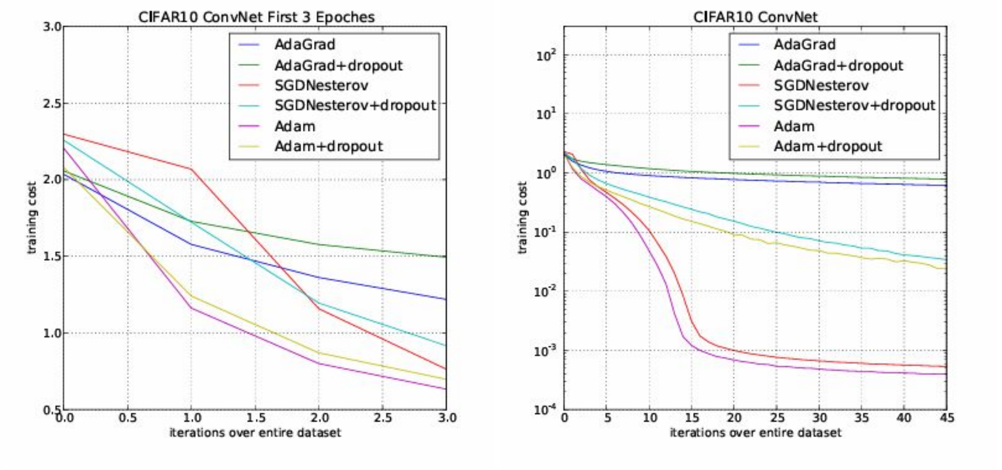


Figure 3: Convolutional neural networks training cost. (left) Training cost for the first three epochs. (right) Training cost over 45 epochs. CIFAR-10 with c64-c64-c128-1000 architecture.

- CNN architecture
  - three alternating stages of 5x5 convolution filters
  - 3x3 max pooling with stride of 2
  - fully connected layer of 1000 rectified linear hidden units (ReLU’s)

Experiments (Ablation)

Bias correction term

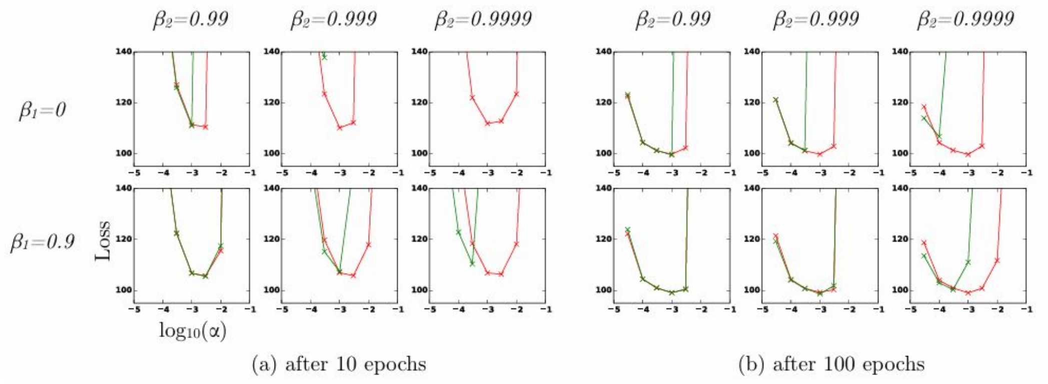


Figure 4: Effect of bias-correction terms (red line) versus no bias correction terms (green line) after 10 epochs (left) and 100 epochs (right) on the loss (y-axes) when learning a Variational Auto-Encoder (VAE) (Kingma & Welling, 2013), for different settings of stepsize α (x-axes) and hyper-parameters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

- Values of **β**2 close to 1, required for robustness to sparse gradients, results in larger initialization bias
- values **β**2 close to 1 indeed lead to instabilities in training when no bias correction term was present, especially at first few epochs of the training

Extensions

AdaMax

Algorithm 2: *AdaMax*, a variant of Adam based on the infinity norm. See section 7.1 for details. Good default settings for the tested machine learning problems are $\alpha = 0.002$, $\beta_1 = 0.9$ and $\beta_2 = 0.999$. With β_1^t we denote β_1 to the power t. Here, $(\alpha/(1-\beta_1^t))$ is the learning rate with the bias-correction term for the first moment. All operations on vectors are element-wise.

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0: Initial parameter vector
  $m_0 \leftarrow 0$ (Initialize 1st moment vector)
  $u_0 \leftarrow 0$ (Initialize the exponentially weighted infinity norm)
  $t \leftarrow 0$ (Initialize timestep)
  while θ_t not converged **do**
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
    $u_t \leftarrow \max(\beta_2 \cdot u_{t-1}, |g_t|)$ (Update the exponentially weighted infinity norm)
    $\theta_t \leftarrow \theta_{t-1} - (\alpha/(1 - \beta_1^t)) \cdot m_t/u_t$ (Update parameters)
  end while
  return θ_t (Resulting parameters)

- We don't need to correct for initialization bias in this case
- Also note that the magnitude of parameter updates has a simpler bound with AdaMax than Adam

Line-up

1. Optimization algorithms
2. Related work
3. Adam
4. Convergence analysis
5. Codes
6. Experiments
7. Limitations

Limitations

- **Non-Convergence Issues:** fail to converge to global minima in certain scenarios, particularly when the learning rates do not diminish over time

  [Dereich et al. 2024. Non-convergence of Adam and other adaptive stochastic gradient descent optimization methods for non-vanishing learning rates]

- **Limit Cycles and Oscillations:** Adam can exhibit limit cycles, leading to oscillations around suboptimal points rather than converging smoothly to an optimal solution

  [Bock et al. 2019. Non-convergence and Limit Cycles in the Adam Optimizer]

Thank you!

Credits: meme-arsenal

Transparent pricing. Flexible Solutions. Pick from a range of packages

01

The Basic Package

Lorem ipsum odor amet, consectetuer adipiscing elit. Senectus natoque curae vehicula fermentum lobortis. Viverra mi eros, id cubilia maximus egestas nisl. Inceptos nisi viverra, faucibus dictum orci bibendum commodo tristique amet. Dolor eu eget suspendisse purus diam massa at leo. Pretium lobortis sit faucibus dapibus taciti.

02

The Classic Package

Lorem ipsum odor amet, consectetuer adipiscing elit. Senectus natoque curae vehicula fermentum lobortis. Viverra mi eros, id cubilia maximus egestas nisl. Inceptos nisi viverra, faucibus dictum orci bibendum commodo tristique amet. Dolor eu eget suspendisse purus diam massa at leo. Pretium lobortis sit faucibus dapibus taciti.

03

The VIP Package

Lorem ipsum odor amet, consectetuer adipiscing elit. Senectus natoque curae vehicula fermentum lobortis. Viverra mi eros, id cubilia maximus egestas nisl. Inceptos nisi viverra, faucibus dictum orci bibendum commodo tristique amet. Dolor eu eget suspendisse purus diam massa at leo. Pretium lobortis sit faucibus dapibus taciti.

Gold

Full day of documentary and portrait photography
All images, hand-edited, available in hi-res and web-size jpegs
Private online gallery for viewing and downloading photos
Access to a high-quality print shop, plus a print release
Unlimited phone consultations and assistance with planning.

Silver

Half day of documentary and portrait photography
All images, hand-edited, available in hi-res and web-size jpegs
Private online gallery for viewing and downloading photos
1 phone consultation and assistance with planning.

Bronze

Up 2 hours of documentary and portrait photography
All images, hand-edited, available in hi-res and web-size jpegs

“

Hassan was world-class. He took his time making sure I was comfortable in front of the camera.

Lorem Ipsum, Portrait Session