New Optimization

Chung-Ming Chien

2020/4/9

Background Knowledge

- μ-strong convexity
- Lipschitz continuity
- Bregman proximal inequality

~~New Optimization~~

Chung-Ming Chien

2020/4/12

New Optimizers for Deep Learning

Chung-Ming Chien

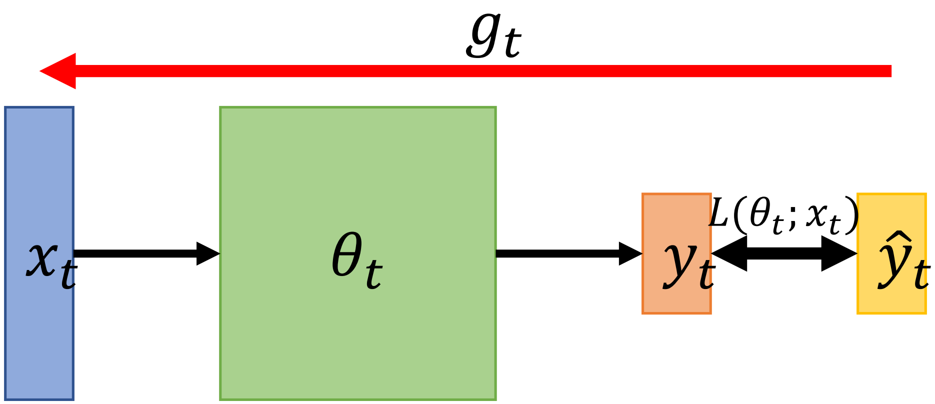
2020/4/12

What you have known before?

- SGD
- SGD with momentum
- Adagrad
- RMSProp
- Adam

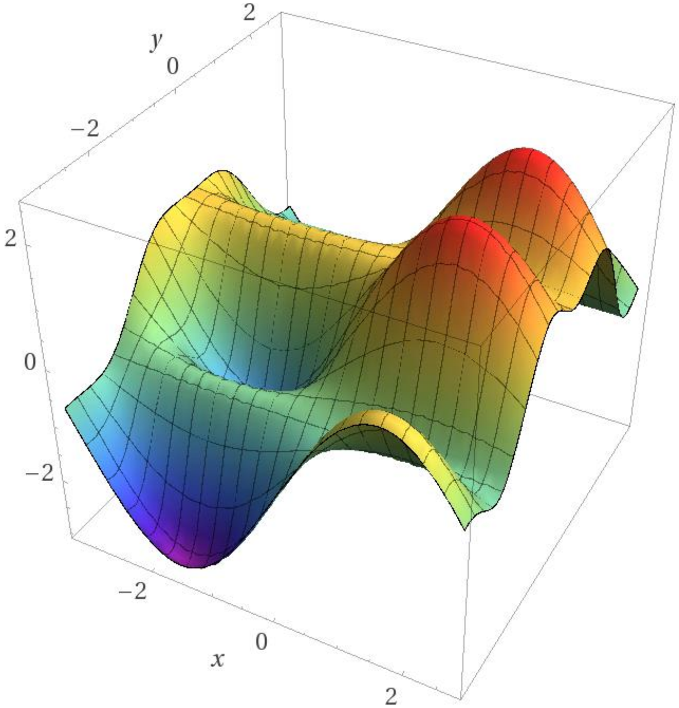
Some Notations

- θ_t : model parameters at time step t
- $\nabla L(\theta_t)$ or g_t: gradient at θ_t, used to compute θ_{t+1}
- m_{t+1}: momentum accumulated from time step 0 to time step t, which is used to compute θ_{t+1}


g_t
x_t
θ_t
y_t
$L(\theta_t; x_t)$
$\hat{y}_t$

What is Optimization about?

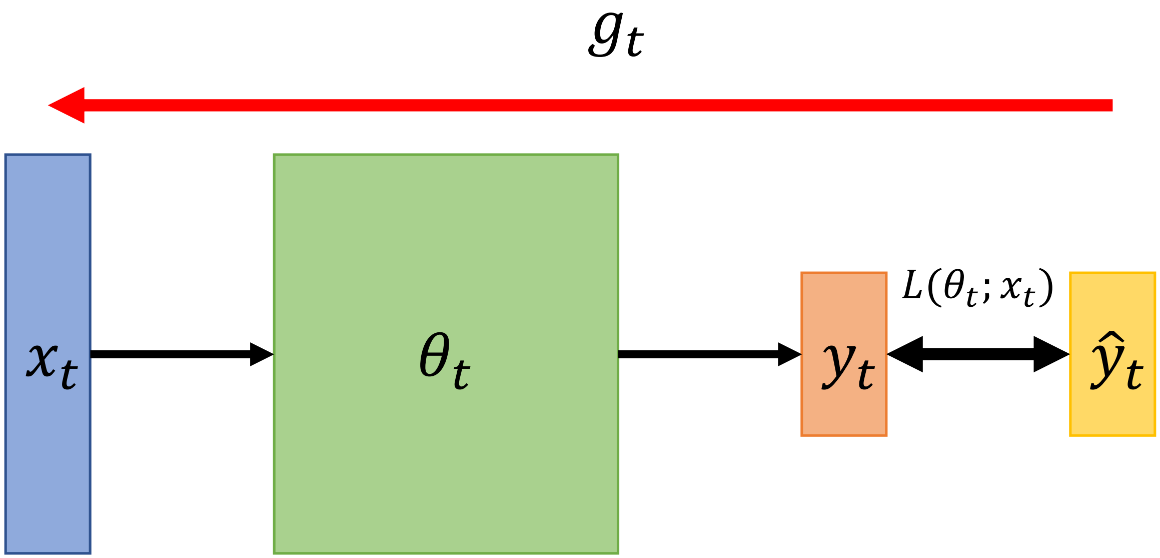
- Find a θ to get the lowest $\sum_x L(\theta; x)$!!
- Or, Find a θ to get the lowest $L(\theta)$!!



Computed by Wolfram|Alpha

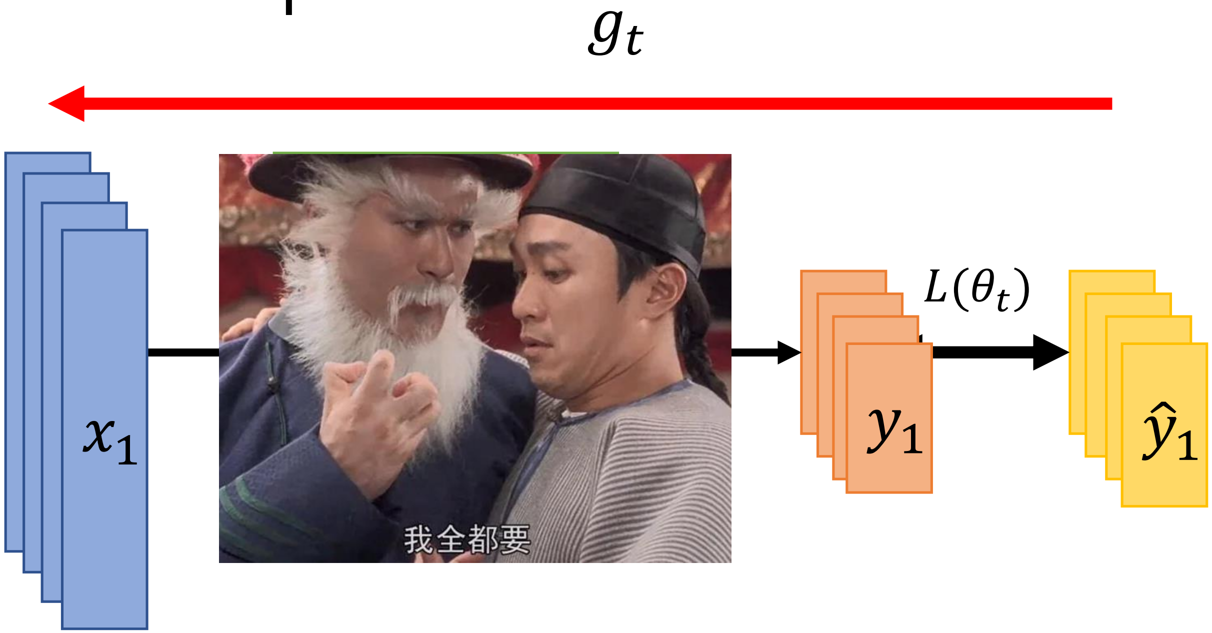
On-line vs Off-line

- On-line：one pair of $(x_t, \hat{y}_t)$ at a time step


g_t
x_t
θ_t
y_t
$L(\theta_t; x_t)$
$\hat{y}_t$

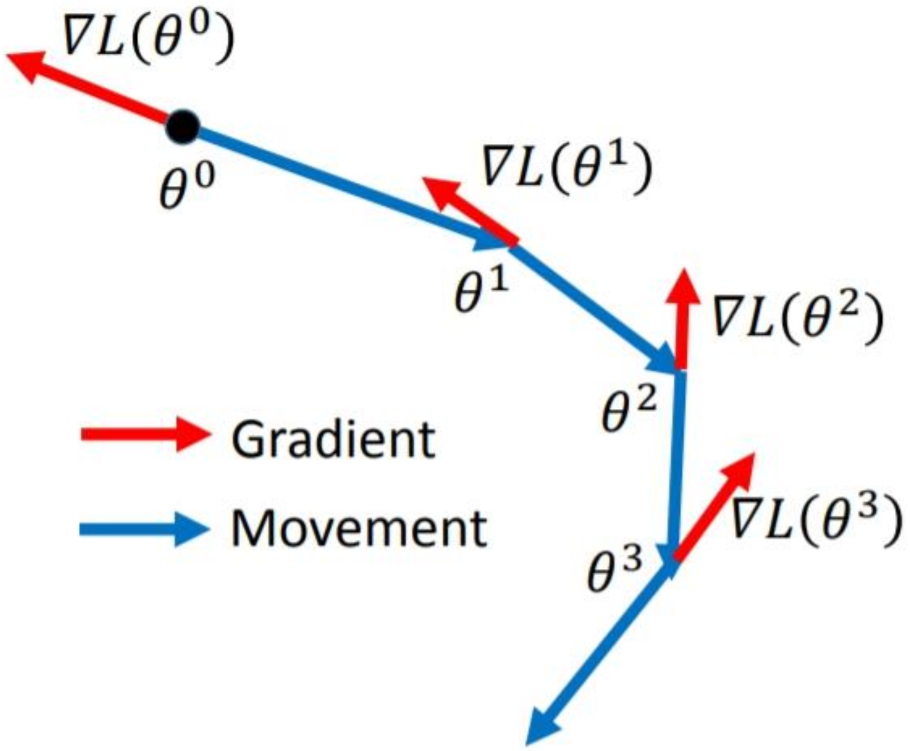
On-line vs Off-line

- Off-line：pour all $(x_t, \hat{y}_t)$ into the model at every time step


g_t
x_1
我全都要
$L(\theta_t)$
y_1
$\hat{y}_1$

- The rest of this lecture will focus on the off-line cases

SGD


$\nabla L(\theta^0)$
θ^0
$\nabla L(\theta^1)$
θ^1
$\nabla L(\theta^2)$
θ^2
Gradient
Movement
$\nabla L(\theta^3)$
θ^3

Start at position θ^0

Compute gradient at θ^0

Move to $\theta^1 = \theta^0 - \eta\nabla L(\theta^0)$

Compute gradient at θ^1

Move to $\theta^2 = \theta^1 - \eta\nabla L(\theta^1)$

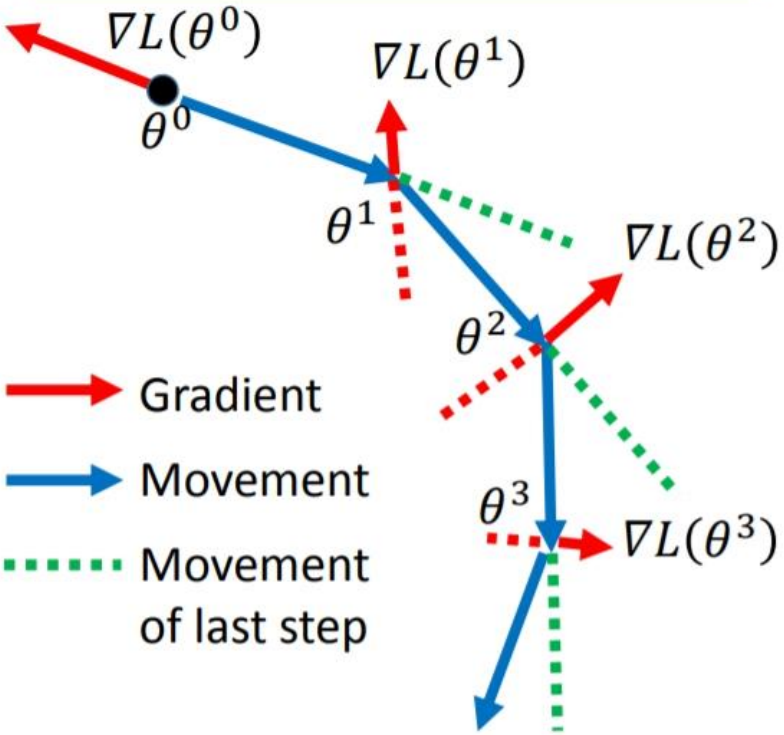
⋮

Stop until $\nabla L(\theta^t) \approx 0$

Credit to 李宏毅老師上課投影片

SGD with Momentum(SGDM)

Movement: movement of last step minus gradient at present


$\nabla L(\theta^0)$
$\nabla L(\theta^1)$
θ^0
θ^1
$\nabla L(\theta^2)$
θ^2
θ^3
$\nabla L(\theta^3)$
Gradient
Movement
Movement of last step

Start at point θ^0

Movement $v^0=0$

Compute gradient at θ^0

Movement $v^1 = \lambda v^0 - \eta \nabla L(\theta^0)$

Move to $\theta^1 = \theta^0 + v^1$

Compute gradient at θ^1

Movement $v^2 = \lambda v^1 - \eta \nabla L(\theta^1)$

Move to $\theta^2 = \theta^1 + v^2$

Movement not just based on gradient, but previous movement.

Credit to 李宏毅老師上課投影片

SGD with Momentum(SGDM)

v^i is actually the weighted sum of all the previous gradient: $\nabla L(\theta^0), \nabla L(\theta^1), \ldots \nabla L(\theta^{i-1})$

$v^0 = 0$

$v^1 = -\eta \nabla L(\theta^0)$

$v^2 = -\lambda \eta \nabla L(\theta^0) - \eta \nabla L(\theta^1)$

Start at point θ^0

Movement v^0=0

Compute gradient at θ^0

Movement $v^1 = \lambda v^0 - \eta \nabla L(\theta^0)$

Move to $\theta^1 = \theta^0 + v^1$

Compute gradient at θ^1

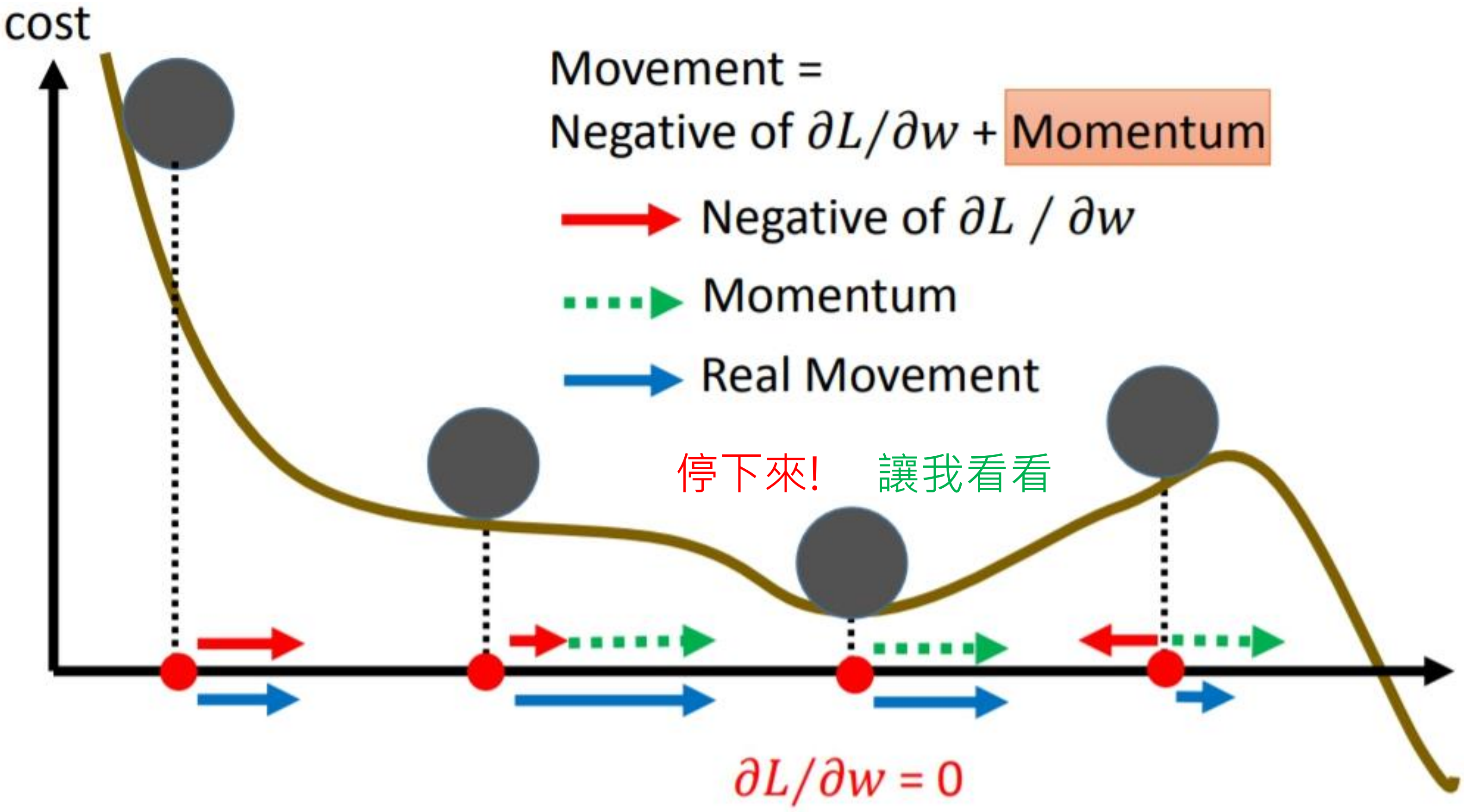
Movement $v^2 = \lambda v^1 - \eta \nabla L(\theta^1)$

Move to $\theta^2 = \theta^1 + v^2$

Movement not just based on gradient, but previous movement.

Credit to 李宏毅老師上課投影片

Why momentum?



Movement =
Negative of $\partial L/\partial w$ + **Momentum**

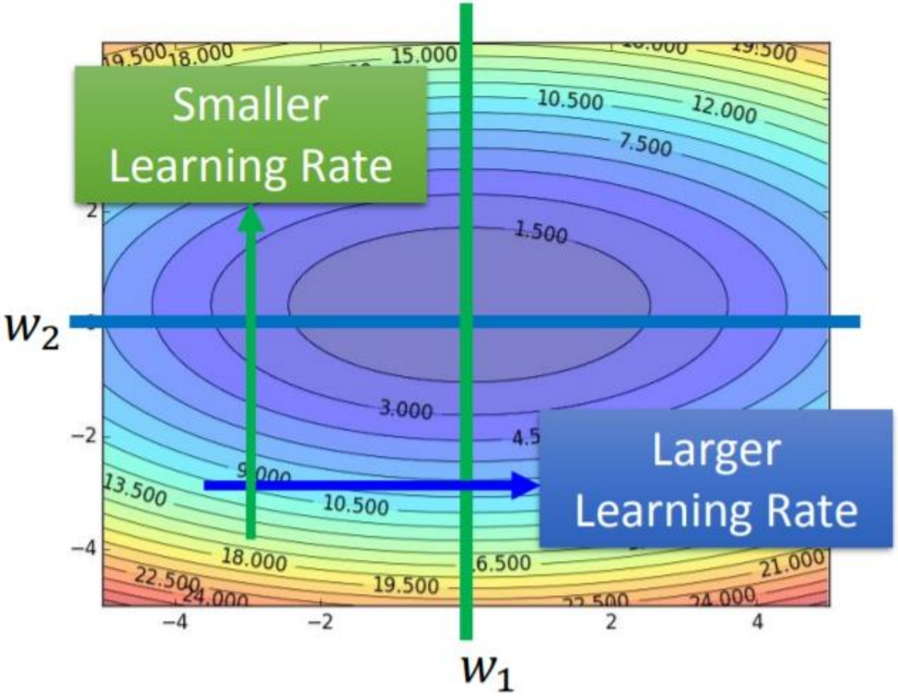
- ➡ Negative of $\partial L / \partial w$
- ⇢ Momentum
- ➡ Real Movement

cost
停下來!
讓我看看
$\partial L/\partial w = 0$

Credit to 李宏毅老師上課投影片

Adagrad

$$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{\sum_{i=0}^{t-1}(g_i)^2}} g_{t-1}$$



What if the gradients at the first few time steps are extremely large…

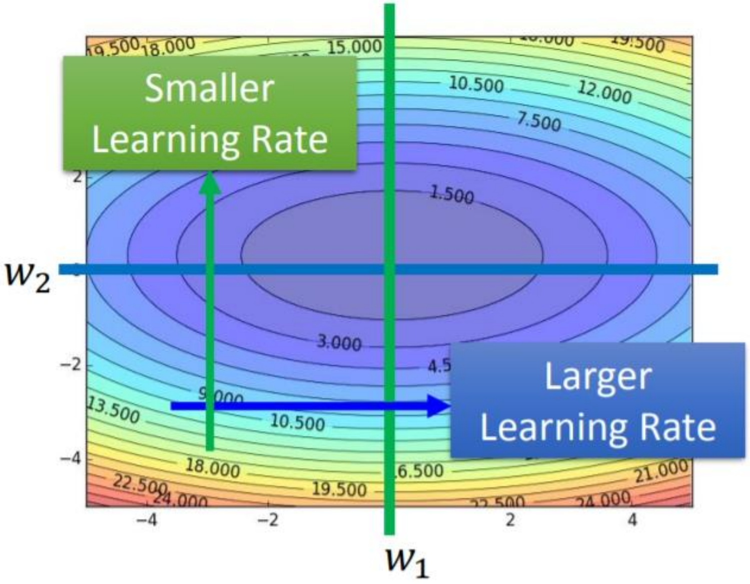
Credit to 李宏毅老師上課投影片

RMSProp

$$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{v_t}} g_{t-1}$$

$$v_1 = {g_0}^2$$

$$v_t = \alpha v_{t-1} + (1-\alpha)(g_{t-1})^2$$



Smaller Learning Rate

Larger Learning Rate

Exponential moving average (EMA) of squared gradients is not monotonically increasing

Credit to 李宏毅老師上課投影片

Adam

- SGDM

$$\theta_t = \theta_{t-1} - \eta m_t$$
$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_{t-1}$$

+

- RMSProp

$$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{v_t}} g_{t-1}$$
$$v_1 = {g_0}^2$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2)(g_{t-1})^2$$

$$\boxed{\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t} + \varepsilon} \hat{m}_t}$$

$$\hat{m}_t = \frac{m_t}{1 - {\beta_1}^t}$$
$$\hat{v}_t = \frac{v_t}{1 - {\beta_2}^t}$$
$$\beta_1 = 0.9$$
$$\beta_2 = 0.999$$
$$\varepsilon = 10^{-8}$$

de-biasing

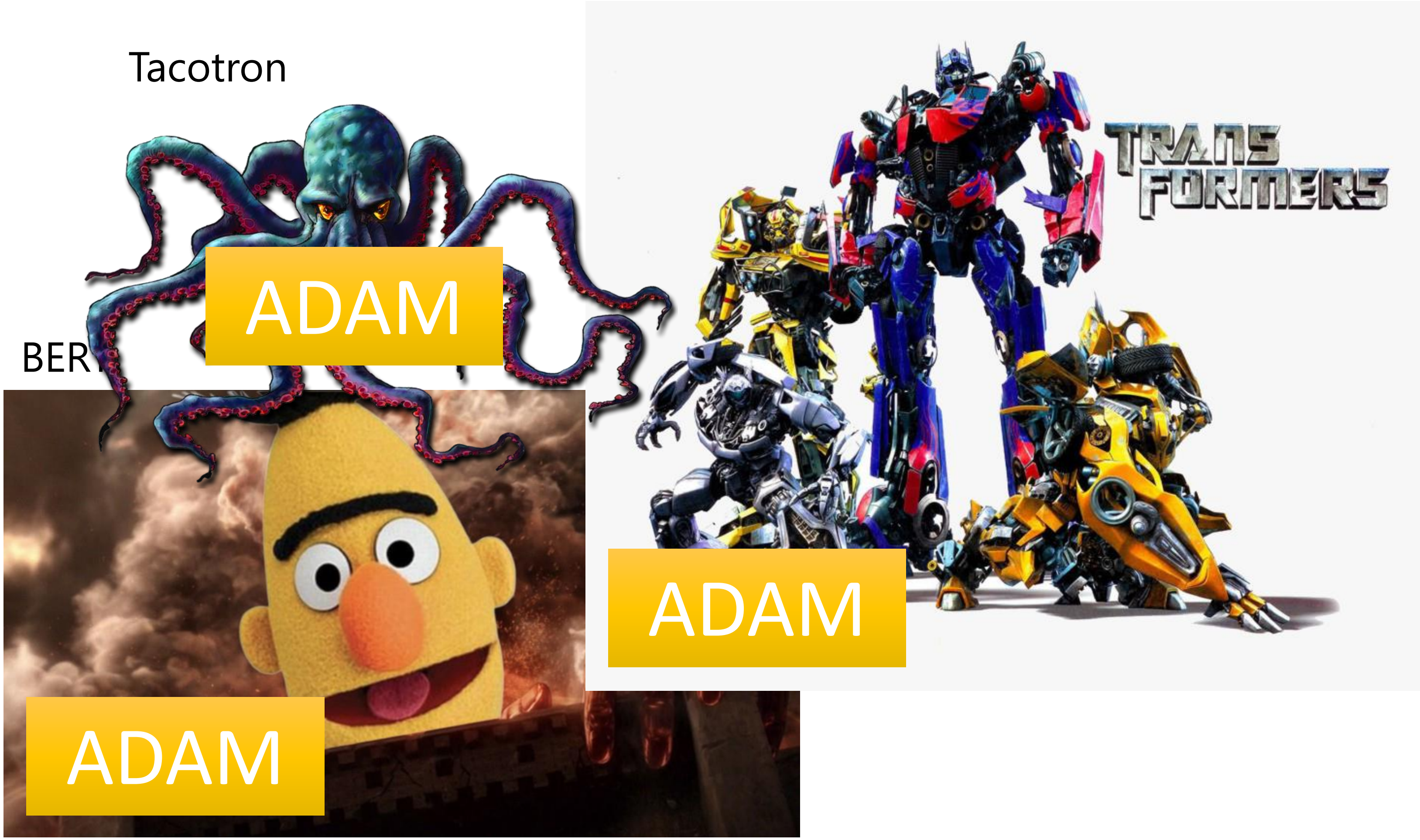
What you have known before?

- SGD [Cauchy, 1847]
- SGD with momentum [Rumelhart, et al., Nature'86]

Adaptive learning rate

- Adagrad [Duchi, et al., JMLR'11]
- RMSProp [Hinton, et al., Lecture slides, 2013]
- Adam [Kingma, et al., ICLR'15]

Optimizers: Real Application


TRANSFORMERS

Transformer

ADAM

Tacotron

ADAM

BERT

ADAM

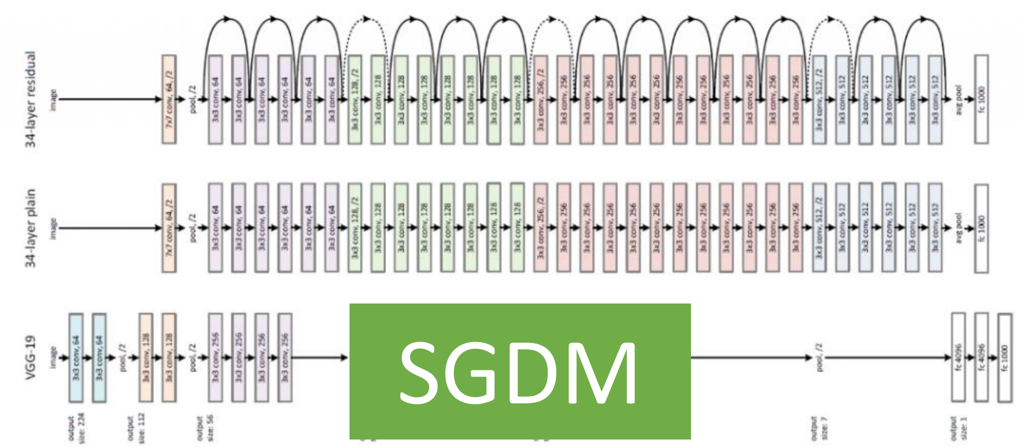
Optimizers: Real Application

Mask R-CNN

operator 1.000
operator 0.999
operator 0.999
ORANGE TEAM
WINNERS
Tom Clancy's Rainbow Six Siege

SGDM

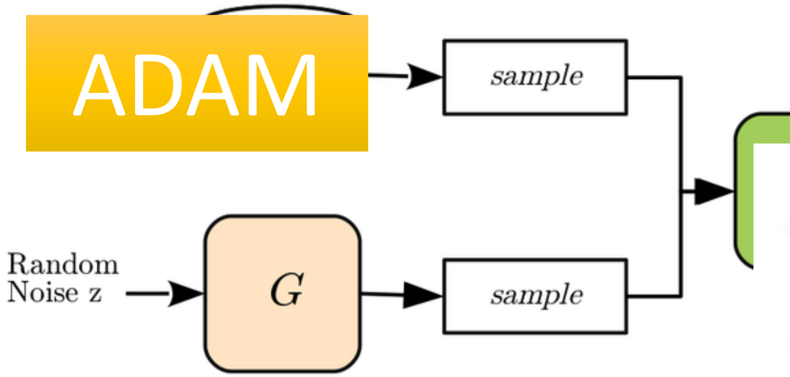
SGDM


34-layer residual
34-layer plain
VGG-19
image
7x7 conv, 64, /2
pool, /2
3x3 conv, 64
3x3 conv, 128, /2
3x3 conv, 128
3x3 conv, 256, /2
3x3 conv, 256
3x3 conv, 512, /2
3x3 conv, 512
avg pool
fc 1000
fc 4096
output size: 224
output size: 112
output size: 56
output size: 7
output size: 1

SGDM

ResNet

Optimizers: Real Application


sample
Random
Noise z
G
sample

ADAM

Big-GAN

ADAM

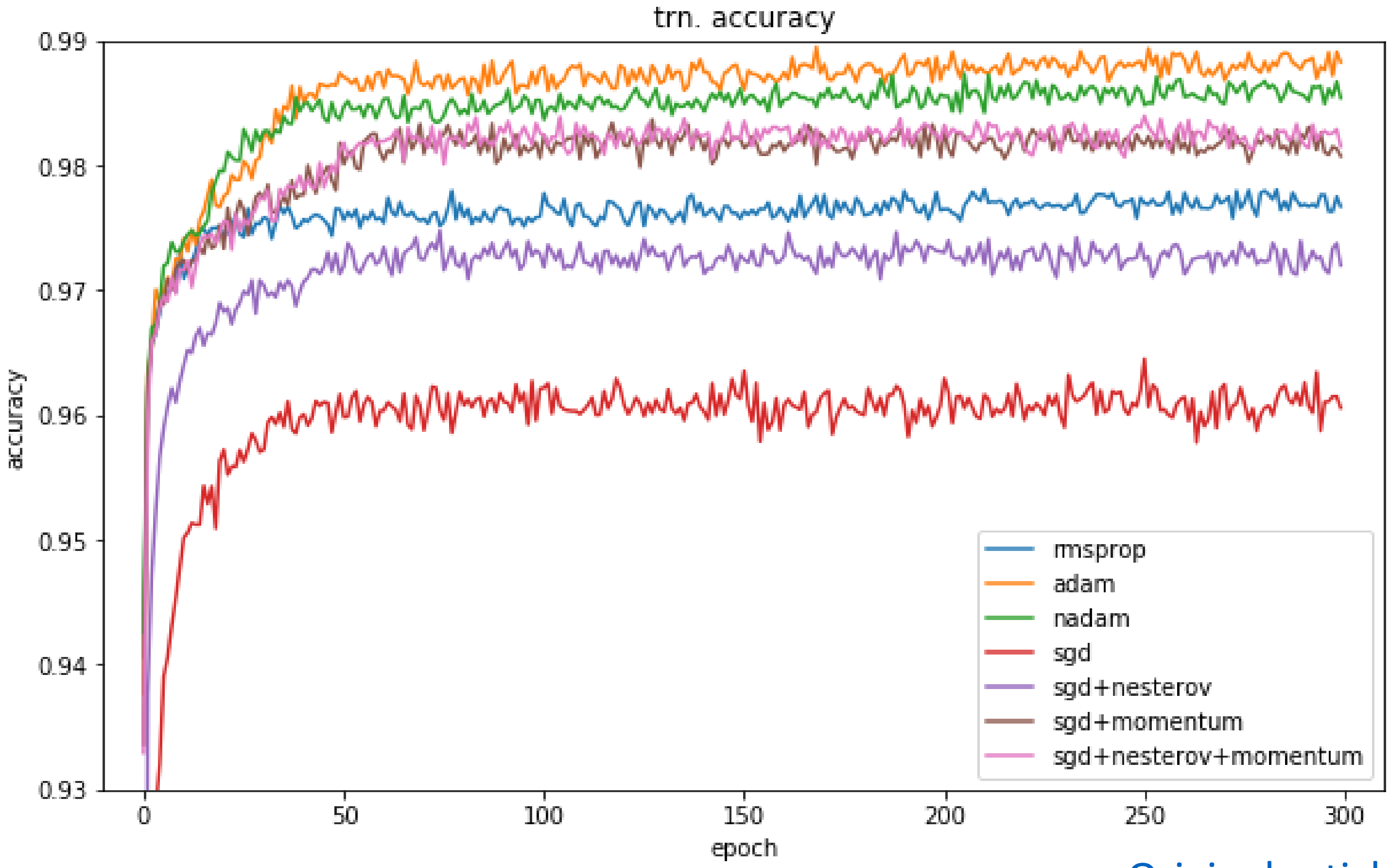
MAML

Back to 2014...

Back to 2014...

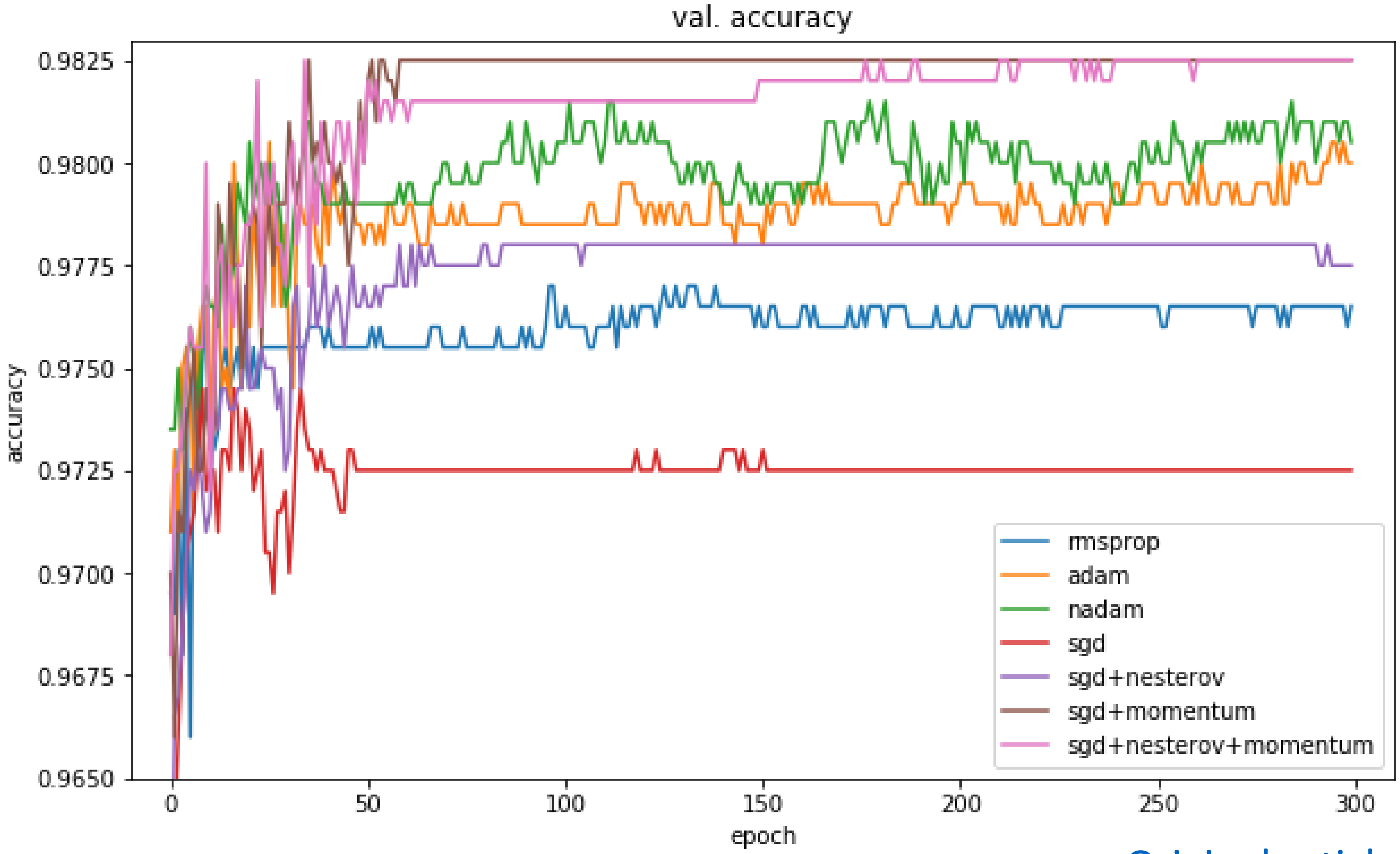
ADAM
喂 没有一个能打的

Adam vs SGDM



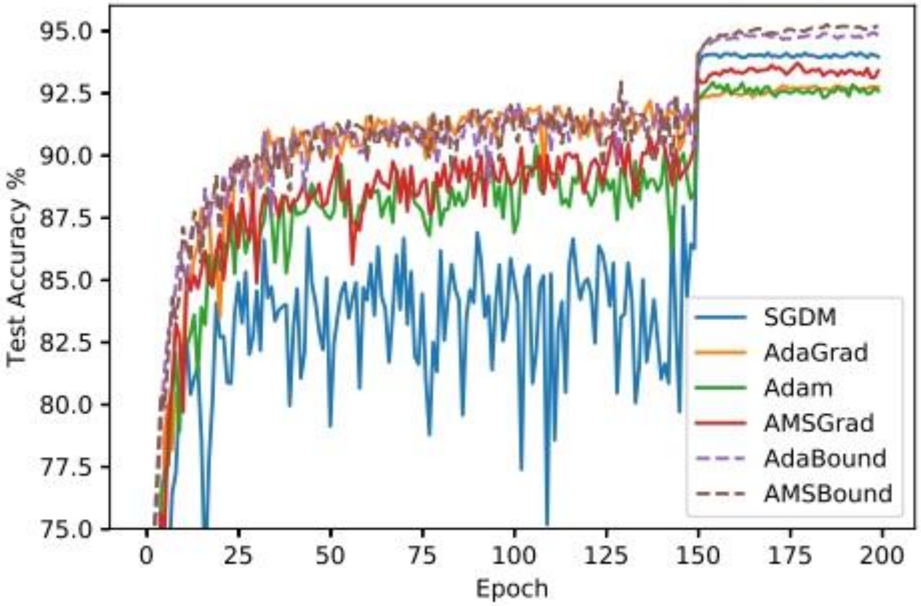
Original article

Adam vs SGDM



Original article

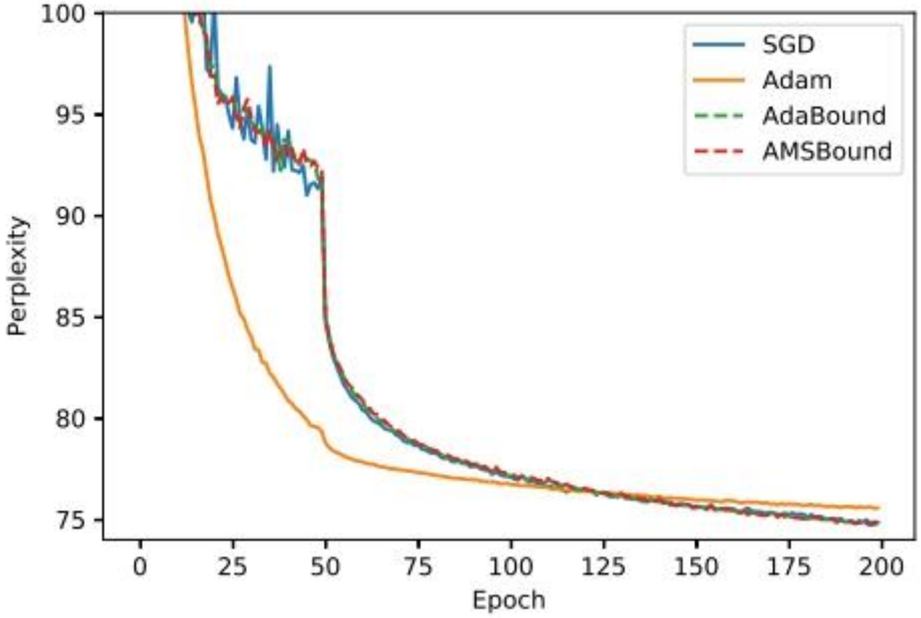
Adam vs SGDM



(d) Test Accuracy for ResNet-34

[Luo, et al., ICLR'19]

Adam vs SGDM



(a) L1: 1-Layer LSTM

[Luo, et al., ICLR'19]

Adam vs SGDM

- Adam：fast training, large generalization gap, unstable
- SGDM：stable, little generalization gap, better convergence(?)

An intuitive illustration for generalization gap

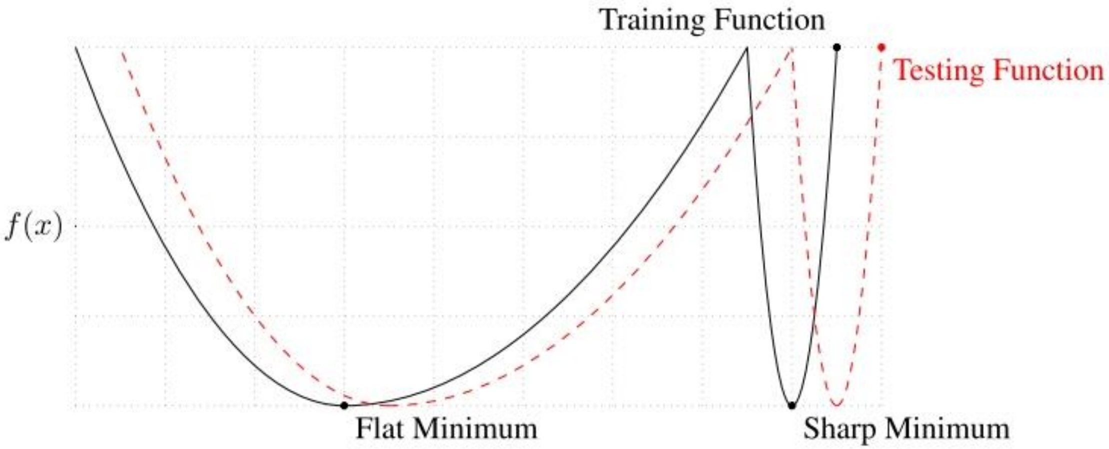
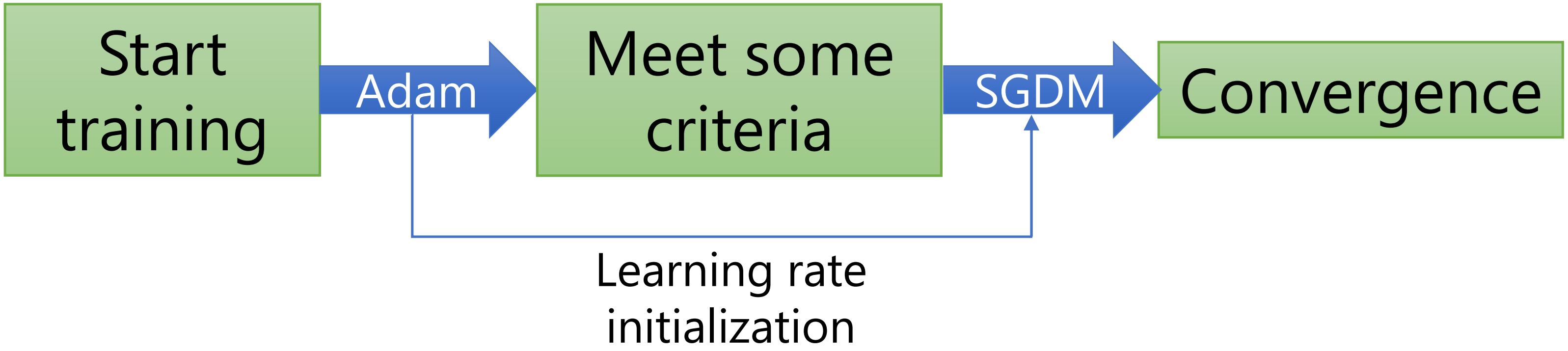


Figure 1: A Conceptual Sketch of Flat and Sharp Minima. The Y-axis indicates value of the loss function and the X-axis the variables (parameters)

Simply combine Adam with SGDM?

- SWATS [Keskar, et al., arXiv'17]

Begin with Adam(fast), end with SGDM


Start training

Adam

Meet some criteria

SGDM

Convergence

Learning rate initialization

Towards Improving Adam…

- Trouble shooting

$$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t + \varepsilon}} \hat{m}_t$$

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_{t-1}, \beta_1 = 0.$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2)(g_{t-1})^2, \beta_2 = 0.999$$

The "memory" of v_t keeps roughly 1000 steps!!

In the final stage of training, most gradients are small and non-informative, while some mini-batches provide large informative gradient rarely.

time step	**…**	**100000**	**100001**	**100002**	**100003**	**…**	**100999**	**101000**
gradient		1	1	1	1		100000	1
movement		η	η	η	η		$10\sqrt{10}\eta$	$10^{-3.5}\eta$

Towards Improving Adam…

- Trouble shooting

Maximum movement distance for one single update is roughly upper bounded by $\sqrt{\frac{1}{1-\beta_2}}\eta$

Non-informative gradients contribute more than informative gradients

time step	…	100000	100001	100002	100003	…	100999	101000
gradient		1	1	1	1		100000	1
movement		η	η	η	η		$10\sqrt{10}\eta$	$10^{-3.5}\eta$

1000η

Towards Improving Adam…

- AMSGrad [Reddi, et al., ICLR'18]

$$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t + \varepsilon}} m_t$$

$$\hat{v}_t = \max(\hat{v}_{t-1}, v_t)$$

ePark》(说唱节目)主理人
唱文化推广者
不對欸

Reduce the influence of non-informative gradients

Remove de-biasing due to the max operation

Monotonically decreasing learning rate

Remember Adagrad vs RMSProp?

Towards Improving Adam…

- Trouble shooting

In the final stage of training, most gradients are small and non-informative, while some mini-batches provide large informative gradient rarely.

Learning rates are either extremely large(for small gradients) or extremely small(for large gradients).

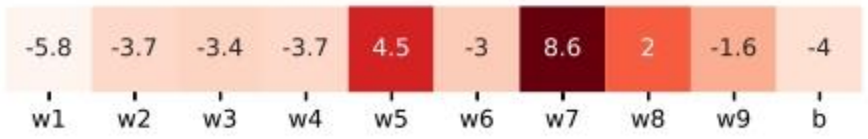


Figure 1: Learning rates of sampled parameters. Each cell contains a value obtained by conducting a logarithmic operation on the learning rate. The lighter cell stands for the smaller learning rate.

Towards Improving Adam…

- AMSGrad only handles large learning rates
- AdaBound [Luo, et al., ICLR'19]

$$\theta_t = \theta_{t-1} - Clip(\frac{\eta}{\sqrt{\hat{v}_t + \varepsilon}})\widehat{m}_t$$

$$Clip(x) = Clip(x, 0.1 - \frac{0.1}{(1-\beta_2)t+1}, 0.1 + \frac{0.1}{(1-\beta_2)t})$$

That's not "adaptive" at all…

That is RUDE

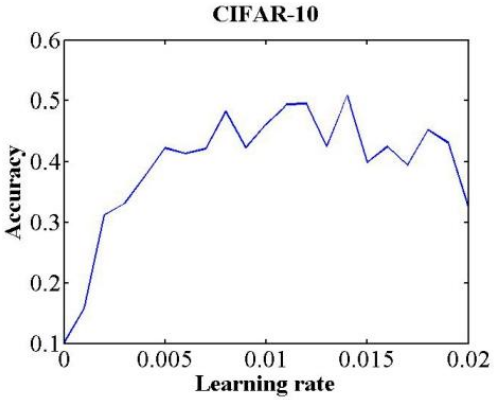
Towards Improving SGDM…

- Adaptive learning rate algorithms：dynamically adjust learning rate over time
- SGD-type algorithms：fix learning rate for all updates… too slow for small learning rates and bad result for large learning rates

There might be a “best” learning rate?

Towards Improving SGDM

- LR range test [Smith, WACV’17]



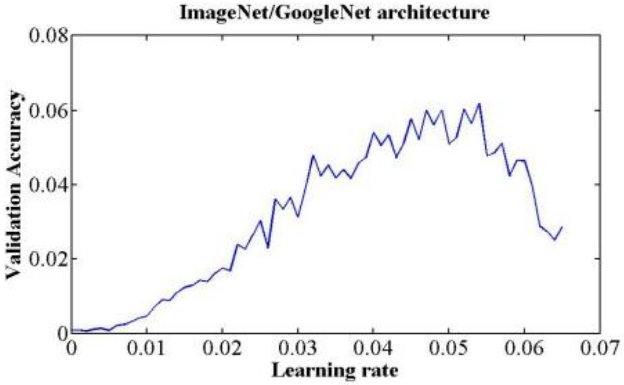


Figure 11. GoogleNet LR range test; validation classification accuracy as a function of increasing learning rate.

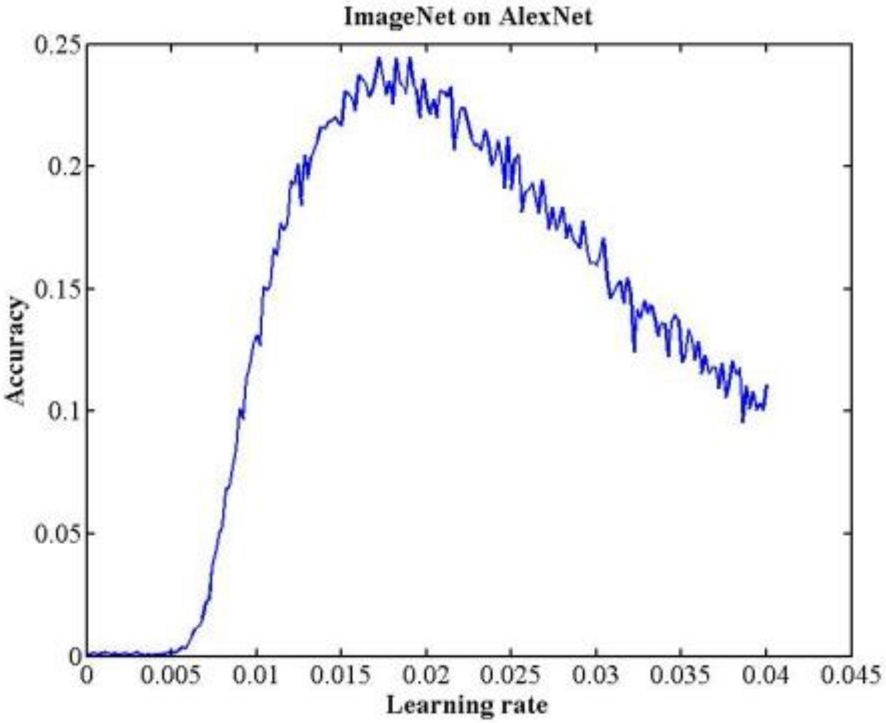
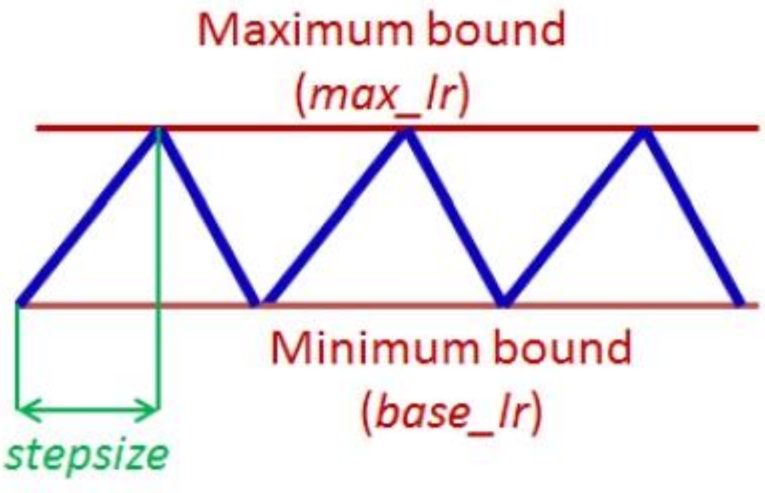


Figure 7. AlexNet LR range test; validation classification accuracy as a function of increasing learning rate.

Towards Improving SGDM

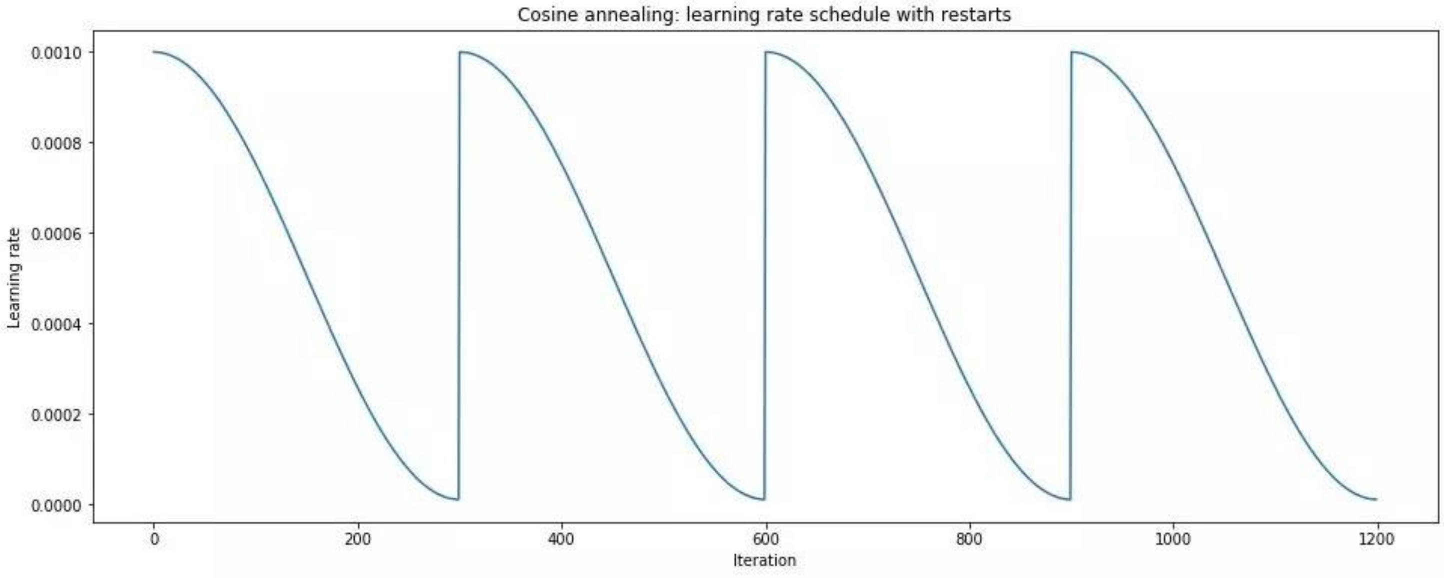
- Cyclical LR [Smith, WACV’17]
- learning rate：decide by LR range test
- stepsize：several epochs
- avoid local minimum by varying learning rate


Maximum bound
(max_lr)
Minimum bound
(base_lr)
stepsize

The more exploration the better!

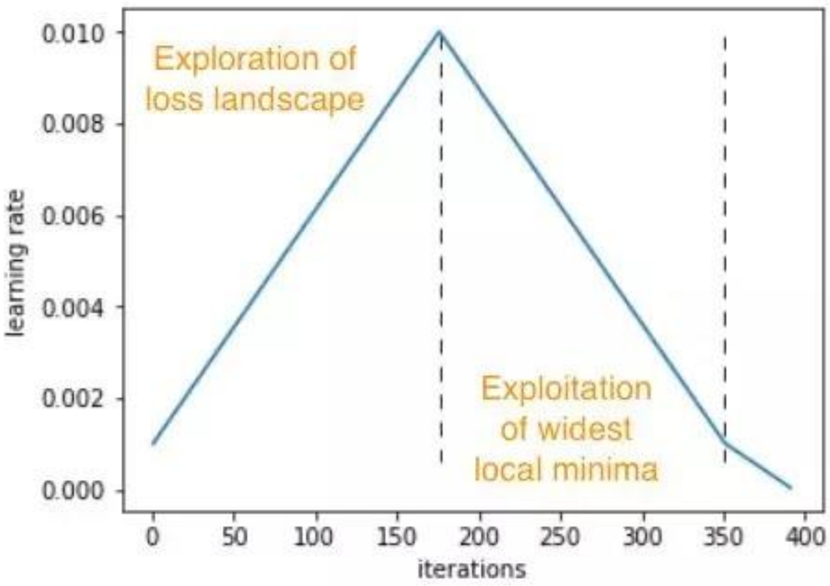
Towards Improving SGDM

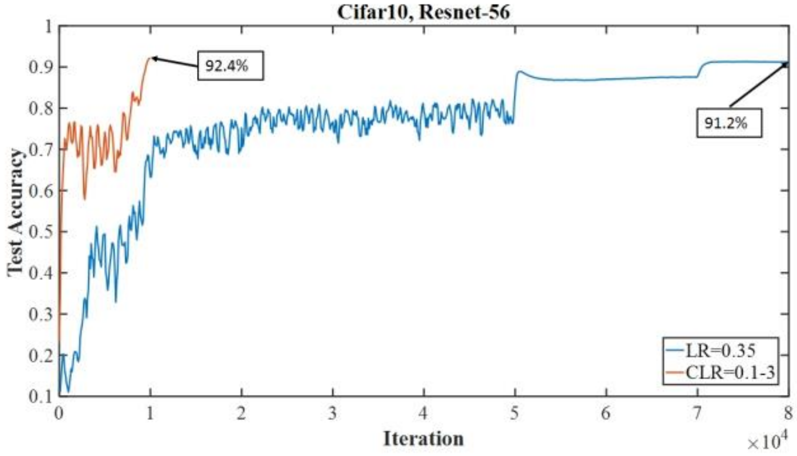
- SGDR [Loshchilov, et al., ICLR'17]



Towards Improving SGDM

- One-cycle LR [Smith, et al., arXiv’17]
- warm-up + annealing + fine-tuning





(a) Comparison of test accuracies of super-convergence example to a typical (piecewise constant) training regime.

Does Adam need warm-up?

Of Course!

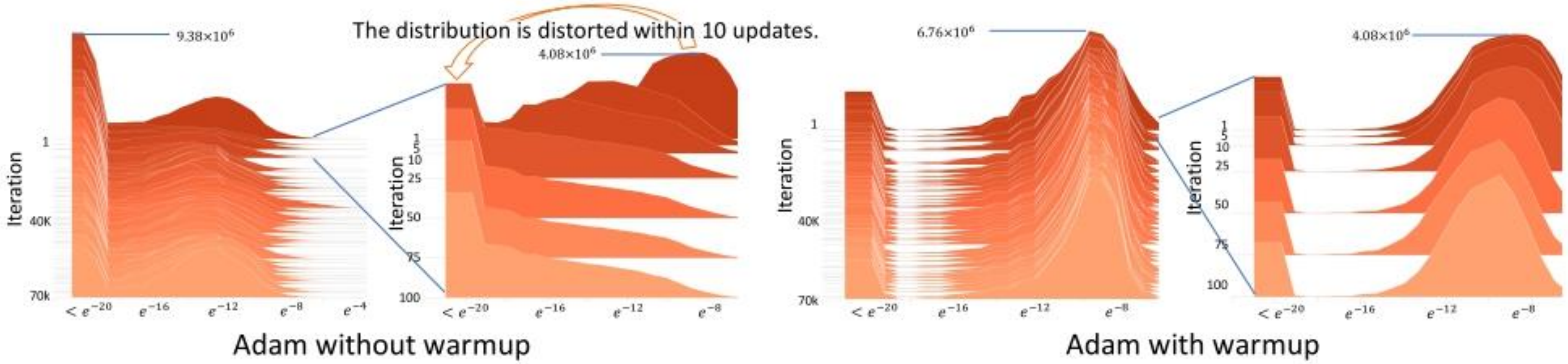
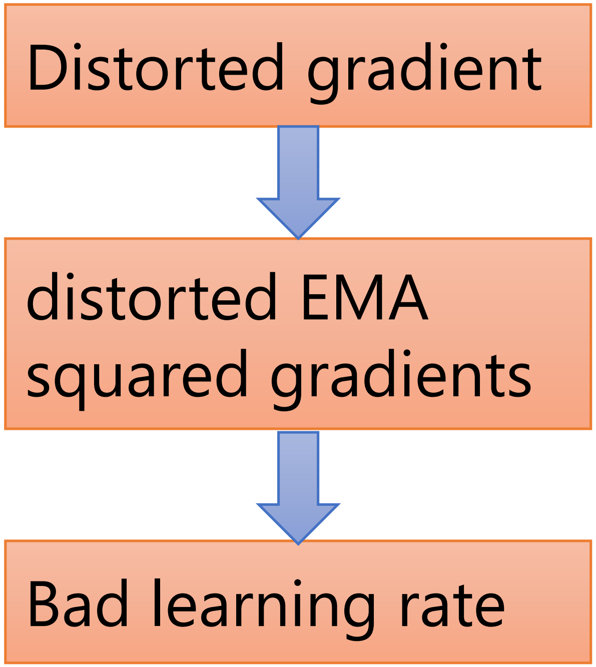


Figure 2: The absolute gradient histogram of the Transformers on the De-En IWSLT’ 14 dataset during the training (stacked along the y-axis). X-axis is absolute value in the log scale and the height is the frequency. Without warmup, the gradient distribution is distorted in the first 10 steps.

Experiments show that the gradient distribution distorted in the first 10 steps

Does Adam need warm-up?



Distorted gradient

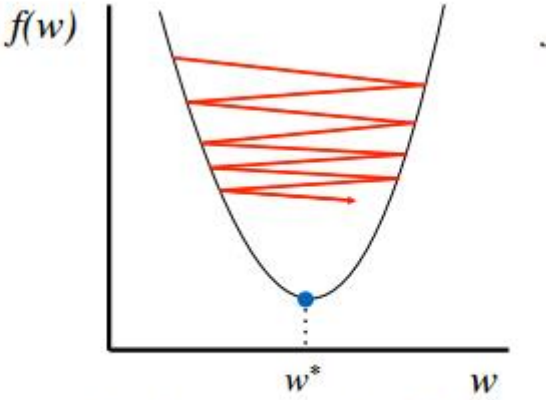
distorted EMA squared gradients

Bad learning rate

$$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t + \varepsilon}} \hat{m}_t$$

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

Keep your step size small at the beginning of training helps to reduce the variance of the gradients



Too big: overshoot and even diverge

Does Adam need warm-up?

- RAdam [Liu, et al., ICLR'20]

$$\rho_t = \rho_\infty - \frac{2t{\beta_2}^t}{1 - {\beta_2}^t}$$

$$\rho_\infty = \frac{2}{1 - \beta_2} - 1$$

$$r_t = \sqrt{\frac{\rho_t - 4)(\rho_t - 2)\rho_\infty}{(\rho_\infty - 4)(\rho_\infty - 2)\rho_t}}$$

effective memory size of EMA

max memory size (t → ∞)

approximated $\frac{Var[\frac{1}{\hat{v}_\infty}]}{Var[\frac{1}{\hat{v}_t}]}$ (for ρ_t>4)

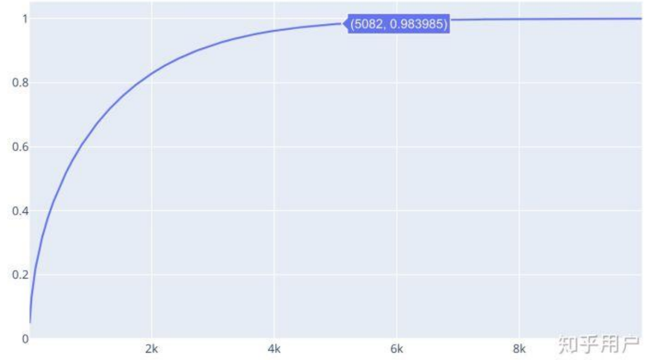
When $\rho_t \leq 4$ (first few steps of training)

$$\theta_t = \theta_{t-1} - \eta \hat{m}_t$$

When $\rho_t > 4$

$$\theta_t = \theta_{t-1} - \frac{\eta r_t}{\sqrt{\hat{v}_t + \varepsilon}} \hat{m}_t$$

r_t is increasing through time!



RAdam vs SWATS

	RAdam	**SWATS**
Inspiration	Distortion of gradient at the beginning of training results in inaccurate adaptive learning rate	non-convergence and generalization gap of Adam, slow training of SGDM
How?	Apply warm-up learning rate to reduce the influence of inaccurate adaptive learning rate	Combine their advantages by applying Adam first, then SGDM
Switch	SGDM to RAdam	Adam to SGDM
Why switch	The approximation of the variance of $\hat{v}_t$ is invalid at the beginning of training	To pursue better convergence
Switch point	When the approximation becomes valid	Some human-defined criteria

k step forward, *1* step back

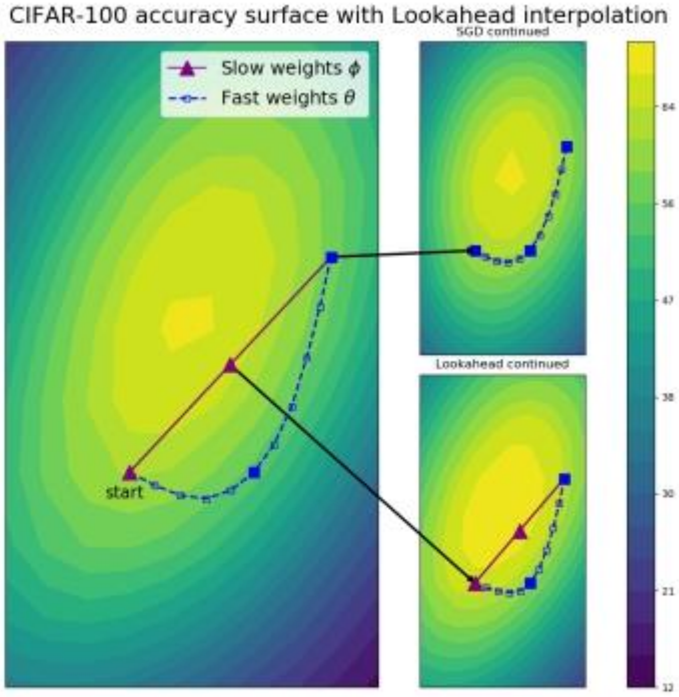
- Lookahead [Zhang, et al., arXiv'19]

universal wrapper for all optimizers

```
For $t = 1, 2, \ldots$ (outer loop)
    $\theta_{t,0} = \phi_{t-1}$
    For $i = 1, 2, \ldots k$ (inner loop)
        $\theta_{t,i} = \theta_{t,i-1} + Optim(Loss, data, \theta_{t,i-1})$
    $\phi_t = \phi_{t-1} + \alpha(\theta_{t,k} - \phi_{t-1})$
```

Similar to Reptile?

$Optim$ can be any optimizer.
E.g. Ranger=
RAdam+Lookahead


CIFAR-100 accuracy surface with Lookahead interpolation

k step forward, *1* step back

- Lookahead [Zhang, et al., arXiv'19]

1 step back: avoid too dangerous exploration

Look for a more flatten minimum

More stable

Better generalization

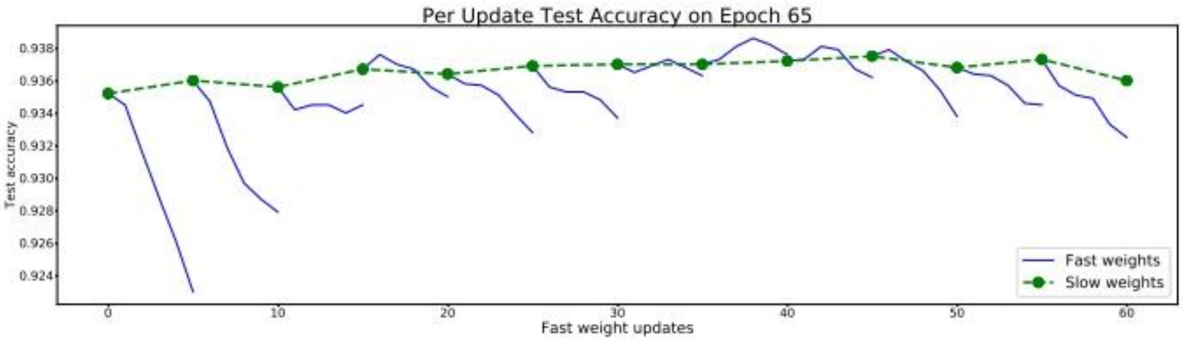
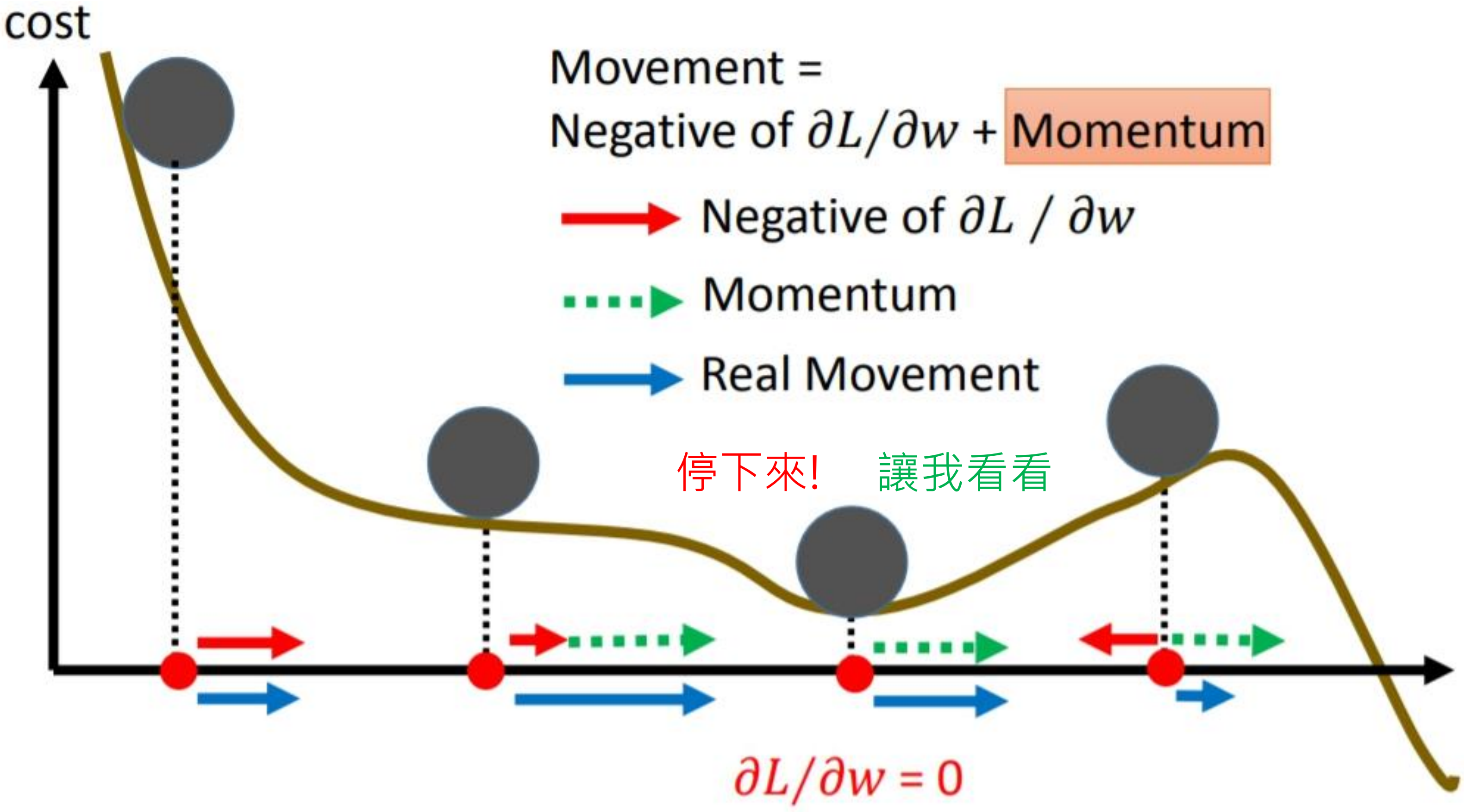


Figure 10: Visualizing Lookahead accuracy for 60 fast weight updates. We plot the test accuracy after every update (the training accuracy and loss behave similarly). The inner loop update tends to degrade both the training and test accuracy, while the interpolation recovers the original performance.

More than momentum…

- Momentum recap



Movement =
Negative of $\partial L/\partial w$ + **Momentum**

→ Negative of $\partial L / \partial w$

⇢ Momentum

→ Real Movement

停下來!

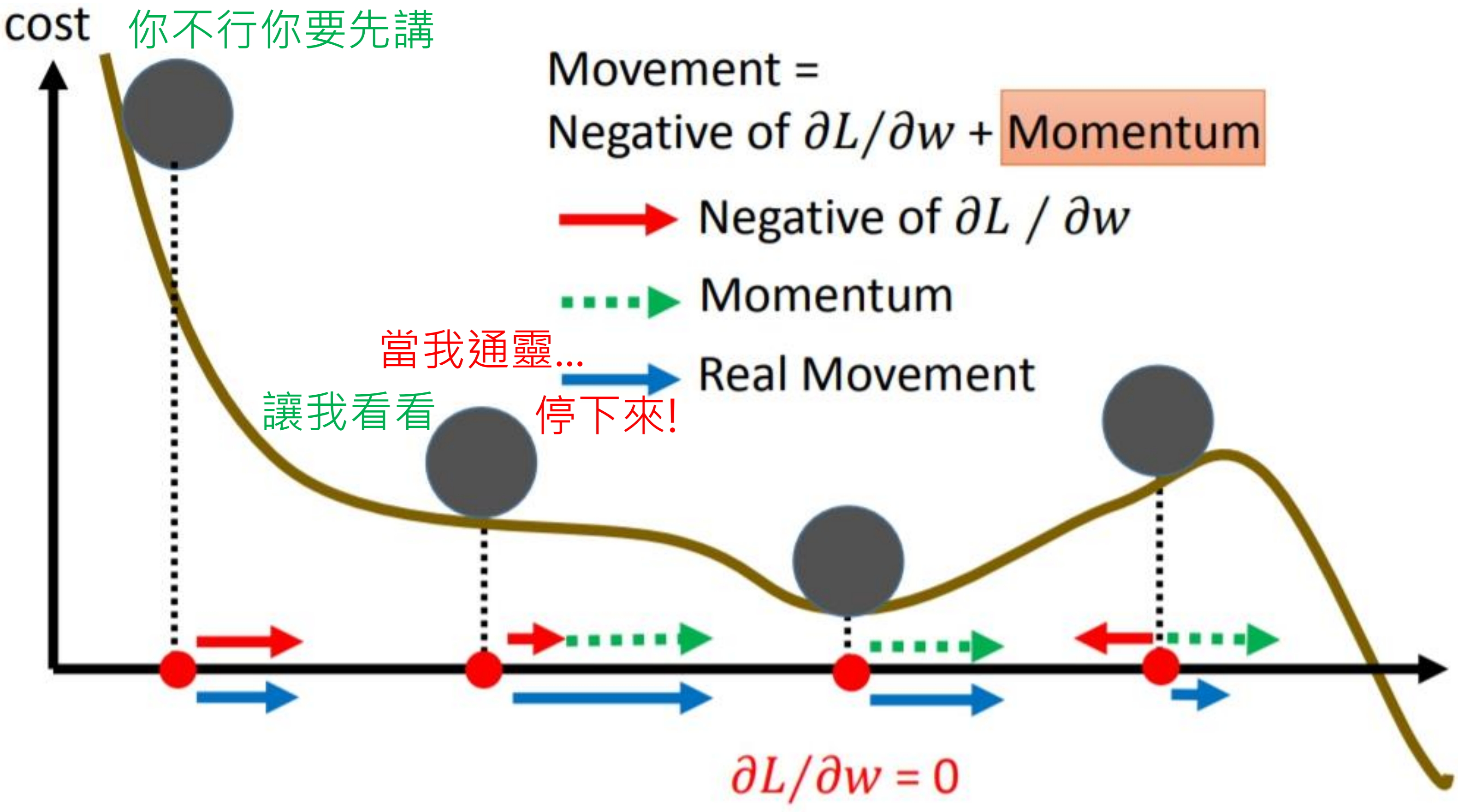
讓我看看

$\partial L/\partial w = 0$

Credit to 李宏毅老師上課投影片

More than momentum...

- Momentum recap


cost
你不行你要先講
當我通靈...
讓我看看
停下來!
$\partial L / \partial w = 0$

Movement =
Negative of $\partial L / \partial w$ + **Momentum**

→ Negative of $\partial L / \partial w$
⇢ Momentum
→ Real Movement

Credit to 李宏毅老師上課投影片

Can we look into the future?

- Nesterov accelerated gradient (NAG)

[Nesterov, jour Dokl. Akad. Nauk SSSR'83]

- SGDM

$$\theta_t = \theta_{t-1} - m_t$$
$$m_t = \lambda m_{t-1} + \eta \nabla L(\theta_{t-1})$$

- Look into the future…

$$\theta_t = \theta_{t-1} - m_t$$
$$m_t = \lambda m_{t-1} + \eta \nabla L(\theta_{t-1} - \lambda m_{t-1})$$

Need to maintain a duplication of model parameters?

Math Warning

Can we look into the future?

- Nesterov accelerated gradient (NAG)

$$\theta_t = \theta_{t-1} - m_t$$
$$m_t = \lambda m_{t-1} + \eta \nabla L(\theta_{t-1} - \lambda m_{t-1})$$

$$\begin{aligned} Let\ \theta_t{}' &= \theta_t - \lambda m_t \\ &= \theta_{t-1} - m_t - \lambda m_t \\ &= \theta_{t-1} - \lambda m_t - \lambda m_{t-1} - \eta \nabla L(\theta_{t-1} - \lambda m_{t-1}) \\ &= \theta_{t-1}{}' - \lambda m_t - \eta \nabla L(\theta_{t-1}{}') \end{aligned}$$
$$m_t = \lambda m_{t-1} + \eta \nabla L(\theta_{t-1}{}')$$

No need to maintain a duplication of model parameters

你也懂超前部署？

- SGDM

$$\theta_t = \theta_{t-1} - m_t$$
$$m_t = \lambda m_{t-1} + \eta \nabla L(\theta_{t-1})$$

or

$$\theta_t = \theta_{t-1} - \lambda m_{t-1} - \eta \nabla L(\theta_{t-1})$$
$$m_t = \lambda m_{t-1} + \eta \nabla L(\theta_{t-1})$$

Adam in the future

- Nadam [Dozat, ICLR workshop'16]

$$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t + \varepsilon}} \hat{m}_t$$

$$\hat{m}_t = \frac{\beta_1 m_t}{1 - {\beta_1}^{t+1}} + \frac{(1 - \beta_1) g_{t-1}}{1 - {\beta_1}^{t}}$$

- SGDM

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

$$= \frac{\beta_1 m_{t-1}}{1 - {\beta_1}^{t}} + \frac{(1 - \beta_1) g_{t-1}}{1 - {\beta_1}^{t}}$$

Do you really know your optimizer?

- A story of L2 regularization…

$$L_{l_2}(\theta) = L(\theta) + \gamma||\theta||^2$$

SGD

$$\begin{aligned}\theta_t &= \theta_{t-1} - \nabla L_{l_2}(\theta_{t-1}) \\ &= \theta_{t-1} - \nabla L(\theta_{t-1}) - \gamma\theta_{t-1}\end{aligned}$$

SGDM

$$\theta_t = \theta_{t-1} - \lambda m_{t-1} - \eta(\nabla L(\theta_{t-1}) + \gamma\theta_{t-1})$$
$$m_t = \lambda m_{t-1} + \eta(\nabla L(\theta_{t-1}) + \gamma\theta_{t-1})\ ?$$
$$m_t = \lambda m_{t-1} + \eta(\nabla L(\theta_{t-1}))\ ?$$

Adam

$$m_t = \lambda m_{t-1} + \eta(\nabla L(\theta_{t-1}) + \gamma\theta_{t-1})\ ?$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2)(\nabla L(\theta_{t-1}) + \gamma\theta_{t-1})^2\ ?$$

Do you really know your optimizer?

- AdamW & SGDW with momentum [Loshchilov, arXiv'17]

SGDWM

$$\theta_t = \theta_{t-1} - m_t - \gamma\theta_{t-1}$$
$$m_t = \lambda m_{t-1} + \eta(\nabla L(\theta_{t-1}))$$

AdamW

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla L(\theta_{t-1})$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2)(\nabla L(\theta_{t-1}))^2$$
$$\theta_t = \theta_{t-1} - \eta(\frac{1}{\sqrt{\hat{v}_t+\varepsilon}}\hat{m}_t + \gamma\theta_{t-1})$$

L2 regularization or weight decay?

Something helps optimization…

- Shuffling
- Dropout
- Gradient noise [Neelakantan, et al., arXiv’15]

$$g_{t,i} = g_{t,i} + N(0, {\sigma_t}^2)$$

$$\sigma_t = \frac{c}{(1+t)^\gamma}$$

The more exploration, the better!

Something helps optimization…

- Warm-up
- Curriculum learning [Bengio, et al., ICML'09]

  Train your model with easy data(e.g. clean voice) first, then difficult data.

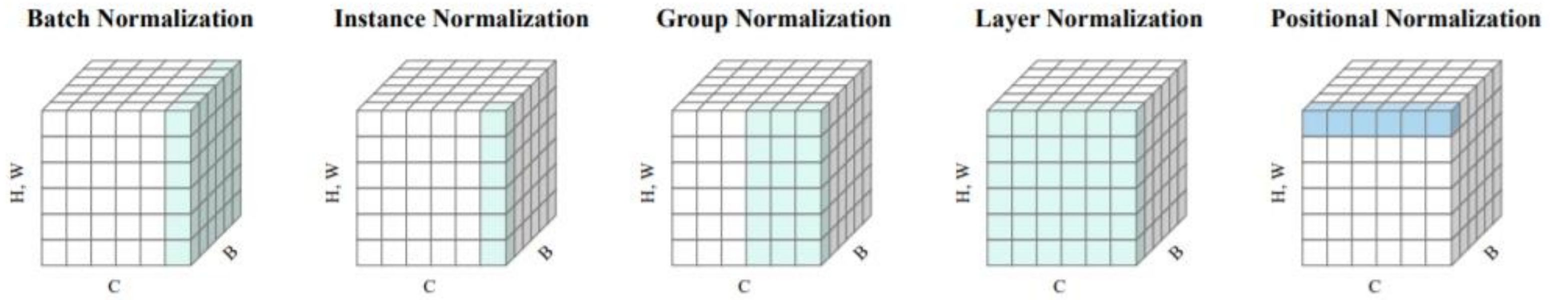
  Perhaps helps to improve generalization

- Fine-tuning

Teach your model patiently!

Something helps optimization…

- Normalization



Batch Normalization

Instance Normalization

Group Normalization

Layer Normalization

Positional Normalization

Figure 2: Positional Normalization together with previous normalization methods. In the figure, each subplot shows a feature map tensor, with B as the batch axis, C as the channel axis, and (H, W) as the spatial axis. The entries colored in green or blue (ours) are normalized by the same mean and standard deviation. Unlike previous methods, our method processes each position independently, and compute both statistics across the channels.

- Regularization

What we learned today?

Team SGD

- SGD
- SGDM
- Learning rate scheduling
- NAG
- SGDWM

Team Adam

- Adagrad
- RMSProp
- Adam
- AMSGrad
- AdaBound
- Learning rate scheduling
- RAdam
- Nadam
- AdamW

Extreme values of learning rate

SWATS

Lookahead

What we learned today?

SGDM	Adam
Slow	Fast
Better convergence	Possibly non-convergence
Stable	Unstable
Smaller generalization gap	Larger generalization gap

Advices

SGDM

- Computer vision
  image classification
  segmentation
  object detection

Adam

- NLP
  QA
  machine translation
  summary
- Speech synthesis
- GAN
- Reinforcement learning

Universal Optimizer?

No Way!!!

What I think I am going to learn in this class

What I actually learned in this class

Reference

- [Ruder, arXiv'17] Sebastian Ruder, "An Overview of Gradient Descent Optimization Algorithms", arXiv, 2017
- [Rumelhart, et al., Nature'86] David E. Rumelhart, Geoffrey E. Hinton and Ronald J. Williams, "Learning Representations by Back-Propagating Errors", Nature, 1986
- [Duchi, et al., JMLR'11] John Duchi, Elad Hazan and Yoram Singer, "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization", JMLR, 2011
- [Hinton, et al., Lecture slides, 2013] Geoffrey Hinton, Nitish Srivastava and Kevin Swersky, "RMSProp", Lecture slides, 2013
- [Kingma, et al., ICLR'15] Diederik P. Kingma and Jimmy Ba, "A Method for Stochastic Optimization", ICLR, 2015

Reference

- [Keskar, et al., arXiv’17] Nitish Shirish Keskar and Richard Socher, “Improving Generalization Performance by Switching from Adam to SGD”, arXiv, 2017
- [Reddi, et al., ICLR’18] Sashank J. Reddi, Satyen Kale and Sanjiv Kumar, “On the Convergence of Adam and Beyond”, ICLR, 2018
- [Luo, et al., ICLR’19] Liangchen Luo, Yuanhao Xiong, Yan Liu and Xu Sun, “Adaptive Gradient Methods with Dynamic Bound of Learning Rate”, ICLR, 2019
- [Smith, WACV’17] Leslie N. Smith, “Cyclical Learning Rates for Training Neural Networks”, WACV, 2017
- [Loshchilov, et al., ICLR’17] Ilya Loshchilov and Frank Hutter, “SGDR: Stochastic Gradient Descent with Warm Restarts”, ICLR, 2017

Reference

- [Smith, et al., arXiv'17] Leslie N. Smith and Nicholay Topin, "Super-Convergence: Very Fast Training of Neural Networks Using Large Learning Rates", arXiv, 2017
- [Liu, et al., ICLR'20] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao and Jiawei Han, "On the Variance of the Adaptive Learning Rate and Beyond", ICLR, 2020
- [Zhang, et al., arXiv'19] Michael R. Zhang, James Lucas, Geoffrey Hinton and Jimmy Ba, "Lookahead Optimizer: k Step Forward, 1 Step Back", arXiv, 2019
- [Nesterov, jour Dokl. Akad. Nauk SSSR'83] Yu. E. Nesterov, "A Method of Solving a Convex Programming Problem with Convergence Rate O(1/k^2)""

Reference

- [Dozat, ICLR workshop’16] Timothy Dozat, “Incorporating Nesterov Momentum into Adam”, ICLR workshop, 2016
- [Loshchilov, arXiv’17] Ilya Loshchilov and Frank Hutter, “Fixing Weight Decay Regularization in Adam”, arXiv, 2017
- [Neelakantan, et al., arXiv’15] Arvind Neelakantan, Luke Vilnis, Quoc V. Le, Ilya Sutskever, Lukasz Kaiser, Karol Kurach and James Martens, “Adding Gradient Noise Improves Learning for Very Deep Networks“, arXiv, 2015
- [Bengio, et al., ICML’09] Yoshua Bengio, Jèrôme Louradour, Ronan Collobert and Jason Weston, “Curriculum Learning”, ICML, 2015