New Optimization

Chung-Ming Chien

2020/4/9
Background Knowledge

• $\mu$-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

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

• Find a $\theta$ to get the lowest $\sum_x L(\theta; x)$ !!
• Or, Find a $\theta$ to get the lowest $L(\theta)$ !!
On-line vs Off-line

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

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

\[ g_t \]

• The rest of this lecture will focus on the off-line cases
Start at position $\theta^0$
Compute gradient at $\theta^0$
Move to $\theta^1 = \theta^0 - \eta \nabla L(\theta^0)$
Compute gradient at $\theta^1$
Move to $\theta^2 = \theta^1 - \eta \nabla L(\theta^1)$
\[ \vdots \]
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

\[ \nabla L(\theta^0) \]

\[ \nabla L(\theta^1) \]

\[ \nabla L(\theta^2) \]

\[ \nabla L(\theta^3) \]

Start at point \( \theta^0 \)
Movement \( v^0 = 0 \)
Compute gradient at \( \theta^0 \)
Movement \( v^1 = \lambda v^0 - \eta \nabla L(\theta^0) \)
Move to \( \theta^1 = \theta^0 + v^1 \)
Compute gradient at \( \theta^1 \)
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 \( \theta^0 \)
Movement \( v^0 = 0 \)
Compute gradient at \( \theta^0 \)
Movement \( v^1 = \lambda v^0 - \eta \nabla L(\theta^0) \)
Move to \( \theta^1 = \theta^0 + v^1 \)
Compute gradient at \( \theta^1 \)
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

$\partial L / \partial w = 0$

Credit to 李宏毅老師上課投影片
Adagrad

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

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

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

\[
v_1 = g_0^2
\]

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

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

• SGDM

\[ \begin{align*}
\theta_t &= \theta_{t-1} - \eta m_t \\
m_t &= \beta_1 m_{t-1} + (1 - \beta_1) g_{t-1}
\end{align*} \]

\[ \theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{v_t} + \varepsilon} \hat{m}_t \]

• RMSProp

\[ \begin{align*}
\theta_t &= \theta_{t-1} - \eta \left( \frac{1}{\sqrt{v_t}} \right) v_{t-1} \\
v_1 &= g_0^2 \\
v_t &= \beta_2 v_{t-1} + (1 - \beta_2) (g_{t-1})^2
\end{align*} \]

\[ \begin{align*}
\hat{m}_t &= \frac{m_t}{1 - \beta_1^t} \\
\hat{v}_t &= \frac{v_t}{1 - \beta_2^t} \\
\beta_1 &= 0.9 \\
\beta_2 &= 0.999 \\
\varepsilon &= 10^{-8}
\end{align*} \]
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

- BERT
- ADAM
- Transformer
- Tacotron

ADAM

BER

ADAM

ADAM
Optimizers: Real Application

Mask R-CNN

SGDM

ResNet
Optimizers: Real Application

ADAM

sample

G

sample

Random Noise z

Big-GAN

MEMO

ADAM

MAML
Back to 2014...
Back to 2014...
Adam vs SGDM

Original article
Adam vs SGDM
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*
Towards Improving Adam...

- Trouble shooting

\[
\begin{align*}
\theta_t &= \theta_{t-1} - \frac{\eta}{\sqrt{v_{t-1} + \epsilon}} \hat{m}_t \\
  m_t &= \beta_1 m_{t-1} + (1 - \beta_1) g_{t-1}, \beta_1 = 0. \\
  v_t &= \beta_2 v_{t-1} + (1 - \beta_2) (g_{t-1})^2, \beta_2 = 0.999
\end{align*}
\]

*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.

<table>
<thead>
<tr>
<th>time step</th>
<th>...</th>
<th>100000</th>
<th>100001</th>
<th>100002</th>
<th>100003</th>
<th>...</th>
<th>100999</th>
<th>101000</th>
</tr>
</thead>
<tbody>
<tr>
<td>gradient</td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>100000</td>
<td>1</td>
</tr>
<tr>
<td>movement</td>
<td>( \eta )</td>
<td>( \eta )</td>
<td>( \eta )</td>
<td>( \eta )</td>
<td>( 10\sqrt{10}\eta )</td>
<td>( 10^{-3.5} \eta )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>time step</th>
<th>...</th>
<th>100000</th>
<th>100001</th>
<th>100002</th>
<th>100003</th>
<th>...</th>
<th>100999</th>
<th>101000</th>
</tr>
</thead>
<tbody>
<tr>
<td>gradient</td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>100000</td>
<td>1</td>
</tr>
<tr>
<td>movement</td>
<td>( \eta )</td>
<td>( \eta )</td>
<td>( \eta )</td>
<td>( \eta )</td>
<td>( \eta )</td>
<td>( 10\sqrt{10}\eta )</td>
<td>( 10^{-3.5}\eta )</td>
<td></td>
</tr>
</tbody>
</table>

\[1000\eta\]
Towards Improving Adam...

- **AMSGrad** [Reddi, et al., ICLR’18]

  \[ \theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{\hat{\nu}_t + \epsilon}} m_t \]
  \[ \hat{\nu}_t = \max(\hat{\nu}_{t-1}, \nu_t) \]

  - 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]

\[
\theta_t = \theta_{t-1} - \text{Clip}(\frac{\eta}{\sqrt{\hat{v}_t + \epsilon}}) \hat{m}_t \\
\text{Clip}(x) = \text{Clip}(x, 0.1 - \frac{0.1}{(1 - \beta_2)t + 1}, 0.1 + \frac{0.1}{(1 - \beta_2)t})
\]

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

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

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

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

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) (g_{t-1})^2$$
Does Adam need warm-up?

- **RAdam** [Liu, et al., ICLR’20]

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

\[
\rho_\infty = \frac{2}{1 - \beta_2} - 1
\]

\[
r_t = \sqrt{\frac{(\rho_t - 2)\rho_\infty}{(\rho_\infty - 4)^2}}
\]

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!

- effective memory size of EMA
- max memory size (\(t \to \infty\))
- approximated \(\frac{\text{var}\left[\frac{1}{\rho_\infty}\right]}{\text{var}\left[\frac{1}{\hat{v}_t}\right]}\) (for \(\rho_t > 4\))
### RAdam vs SWATS

<table>
<thead>
<tr>
<th></th>
<th>RAdam</th>
<th>SWATS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Inspiration</strong></td>
<td>Distortion of gradient at the beginning of training results in inaccurate adaptive learning rate</td>
<td>non-convergence and generalization gap of Adam, slow training of SGDM</td>
</tr>
<tr>
<td><strong>How?</strong></td>
<td>Apply warm-up learning rate to reduce the influence of inaccurate adaptive learning rate</td>
<td>Combine their advantages by applying Adam first, then SGDM</td>
</tr>
<tr>
<td><strong>Switch</strong></td>
<td>SGDM to RAdam</td>
<td>Adam to SGDM</td>
</tr>
<tr>
<td><strong>Why switch</strong></td>
<td>The approximation of the variance of $\hat{\nu}_t$ is invalid at the beginning of training</td>
<td>To pursue better convergence</td>
</tr>
<tr>
<td><strong>Switch point</strong></td>
<td>When the approximation becomes valid</td>
<td>Some human-defined criteria</td>
</tr>
</tbody>
</table>
$k$ 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)
\[
\theta_{t,i} = \theta_{t,i-1} + \text{Optim}(\text{Loss}, \text{data}, \theta_{t,i-1})
\]
\[
\phi_t = \phi_{t-1} + \alpha(\theta_{t,k} - \phi_{t-1})
\]

Similar to Reptile?

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

Cost

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

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

停下来！让我看看

$\partial L / \partial w = 0$

Credit to 李宏毅老师上课投影片
More than momentum...

- Momentum recap

\[
\text{Movement} = \text{Negative of } \frac{\partial L}{\partial w} + \text{Momentum}
\]

\[
\frac{\partial L}{\partial w} = 0
\]

Credit to 李宏毅老師上課投影片
Can we look into the future?

• Nesterov accelerated gradient (NAG)  

• SGDM  
  \[ \theta_t = \theta_{t-1} - m_t \]  
  \[ m_t = \lambda m_{t-1} + \eta \nabla L(\theta_{t-1}) \]

• Look into the future...  
  \[ \theta_t = \theta_{t-1} - m_t \]  
  \[ m_t = \lambda m_{t-1} + \eta \nabla L(\theta_{t-1} - \lambda m_{t-1}) \]

Math Warning

Need to maintain a duplication of model parameters?
Can we look into the future?

• Nesterov accelerated gradient (NAG)

\[
\begin{align*}
\theta_t &= \theta_{t-1} - m_t \\
m_t &= \lambda m_{t-1} + \eta \nabla L(\theta_{t-1} - \lambda m_{t-1})
\end{align*}
\]

Let \( \theta_t' = \theta_t - \lambda m_t \)

\[
\begin{align*}
&= \theta_{t-1} - m_t - \lambda m_t \\
&= \theta_{t-1} - \lambda m_t - \lambda m_{t-1} - \eta \nabla L(\theta_{t-1} - \lambda m_{t-1}) \\
&= \theta_{t-1}' - \lambda m_t - \eta \nabla L(\theta_{t-1}')
\end{align*}
\]

\[
\begin{align*}
m_t &= \lambda m_{t-1} + \eta \nabla L(\theta_{t-1}')
\end{align*}
\]

• SGDM

\[
\begin{align*}
\theta_t &= \theta_{t-1} - m_t \\
m_t &= \lambda m_{t-1} + \eta \nabla L(\theta_{t-1})
\end{align*}
\]

or

\[
\begin{align*}
\theta_t &= \theta_{t-1} - \lambda m_{t-1} - \eta \nabla L(\theta_{t-1}) \\
m_t &= \lambda m_{t-1} + \eta \nabla L(\theta_{t-1})
\end{align*}
\]
Adam in the future

- **Nadam**  [Dozat, ICLR workshop’16]

\[
\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} \hat{m}_t \\
\hat{m}_t = \frac{\beta_1 m_t}{1 - \beta_1^{t+1}} + \frac{(1 - \beta_1) g_{t-1}}{1 - \beta_1^t}
\]

- **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
\[
\theta_t = \theta_{t-1} - \nabla L_{l_2}(\theta_{t-1}) \\
= \theta_{t-1} - \nabla L(\theta_{t-1}) - \gamma \theta_{t-1}
\]

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**

\[
\begin{align*}
\theta_t &= \theta_{t-1} - m_t - \gamma \theta_{t-1} \\
\theta_t &= \theta_{t-1} - \eta(\nabla L(\theta_{t-1}))
\end{align*}
\]

**AdamW**

\[
\begin{align*}
\nabla L(\theta_{t-1}) &= \beta_1 m_{t-1} + (1 - \beta_1) \nabla L(\theta_{t-1}) \\
\nabla L(\theta_{t-1}) &= \beta_2 v_{t-1} + (1 - \beta_2) (\nabla L(\theta_{t-1}))^2 \\
\theta_t &= \theta_{t-1} - \eta \left( \frac{1}{\sqrt{v_t + \epsilon}} \hat{m}_t + \gamma \theta_{t-1} \right)
\end{align*}
\]

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

![Normalization Diagram]

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

SWATS

Extreme values of learning rate

Lookahead
**What we learned today?**

<table>
<thead>
<tr>
<th>SGDM</th>
<th>Adam</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slow</td>
<td>Fast</td>
</tr>
<tr>
<td>Better convergence</td>
<td>Possibly non-convergence</td>
</tr>
<tr>
<td>Stable</td>
<td>Unstable</td>
</tr>
<tr>
<td>Smaller generalization gap</td>
<td>Larger generalization gap</td>
</tr>
</tbody>
</table>
## Advices

<table>
<thead>
<tr>
<th>SGDM</th>
<th>Adam</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Computer vision</td>
<td>• NLP</td>
</tr>
<tr>
<td>image classification</td>
<td>QA</td>
</tr>
<tr>
<td>segmentation</td>
<td>machine translation</td>
</tr>
<tr>
<td>object detection</td>
<td>summary</td>
</tr>
<tr>
<td></td>
<td>• Speech synthesis</td>
</tr>
<tr>
<td></td>
<td>• GAN</td>
</tr>
<tr>
<td></td>
<td>• Reinforcement learning</td>
</tr>
</tbody>
</table>
Universal Optimizer?

No Way!!!
What I think I am going to learn in this class

Adam
What I actually learned in this class
Reference


Reference

- [Smith, WACV’17] Leslie N. Smith, “Cyclical Learning Rates for Training Neural Networks”, WACV, 2017
Reference

- [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
Reference


