Adam (Adaptive Moments) I

- The update rule (Kingma and Ba, 2015)

$$
\begin{aligned}
\boldsymbol{g} &\leftarrow \frac{\boldsymbol{\theta}}{C} + \frac{1}{|S|}\nabla_{\boldsymbol{\theta}} \sum_{i:i\in S} \xi(\boldsymbol{\theta}; \boldsymbol{y}^i, Z^{1,i}) \\
\boldsymbol{s} &\leftarrow \rho_1 \boldsymbol{s} + (1-\rho_1)\boldsymbol{g} \\
\boldsymbol{r} &\leftarrow \rho_2 \boldsymbol{r} + (1-\rho_2)\boldsymbol{g} \odot \boldsymbol{g} \\
\hat{\boldsymbol{s}} &\leftarrow \frac{\boldsymbol{s}}{1-\rho_1^t} \\
\hat{\boldsymbol{r}} &\leftarrow \frac{\boldsymbol{r}}{1-\rho_2^t} \\
\boldsymbol{\theta} &\leftarrow \boldsymbol{\theta} - \frac{\epsilon}{\sqrt{\hat{\boldsymbol{r}}} + \delta} \odot \hat{\boldsymbol{s}}
\end{aligned}
$$

Adam (Adaptive Moments) II

- t is the current iteration index
- Roughly speaking, Adam is the combination of
  - Momentum
  - RMSprop
- From Goodfellow et al. (2016),

$$\frac{\epsilon}{\sqrt{\hat{\boldsymbol{r}}} + \delta} \odot \hat{\boldsymbol{s}}$$

(i.e., the use of momentum combined with rescaling) "does not have a clear theoretical motivation"

Adam (Adaptive Moments) III

- How about Adam's practical performance?
- From Goodfellow et al. (2016), "generally regarded as being fairly robust to the choice of hyperparmeters, though the learning rate may need to be changed from the default"
- However, from the web page we referred to for deriving the bias correction, "The original paper ... showing huge performance gains in terms of speed of training. However, after a while people started noticing, that in some cases Adam actually finds worse solution than stochastic gradient"

Adam (Adaptive Moments) IV

- One example of showing the above is Wilson et al. (2017)

Bias Correction in Adam I

- The two steps in Adam

$$\hat{\boldsymbol{s}} \leftarrow \frac{\boldsymbol{s}}{1-\rho_1^t}$$

$$\hat{\boldsymbol{r}} \leftarrow \frac{\boldsymbol{r}}{1-\rho_2^t}$$

  are called "bias correction"
- Why do we need this "bias correction"?
- Note that $\boldsymbol{s}$ is the direction used to update $\boldsymbol{\theta}$.

Bias Correction in Adam II

- We hope that its expectation is similar to the expected gradient

$$E[\boldsymbol{s}_t] = E[\boldsymbol{g}_t]$$

and

$$E[\boldsymbol{r}_t] = E[\boldsymbol{g}_t \odot \boldsymbol{g}_t],$$

where t is the iteration index

- The problem is that due to the moving average, the vector is biased toward the initial value
- Note that our initial $\boldsymbol{s}$ is $\mathbf{0}$

Bias Correction in Adam III

- For $\boldsymbol{s}_t$, we have

$$\begin{aligned}
\boldsymbol{s}_t &= \rho_1 \boldsymbol{s}_{t-1} + (1-\rho_1)\boldsymbol{g}_t \\
&= \rho_1(\rho_1 \boldsymbol{s}_{t-2} + (1-\rho_1)\boldsymbol{g}_{t-1}) + (1-\rho_1)\boldsymbol{g}_t \\
&= (1-\rho_1)\sum_{i=1}^{t} \rho_1^{t-i}\boldsymbol{g}_i
\end{aligned}$$

Bias Correction in Adam IV

- Then

$$\begin{aligned} E[\boldsymbol{s}_t] &= E[(1-\rho_1)\sum_{i=1}^{t}\rho_1^{t-i}\boldsymbol{g}_i] \\ &= E[\boldsymbol{g}_t](1-\rho_1)\sum_{i=1}^{t}\rho_1^{t-i} \end{aligned}$$

- Note that we assume

$$E[\boldsymbol{g}_i], \forall i \geq 1$$

are the same

Bias Correction in Adam V

- Next,

$$
\begin{aligned}
&(1-\rho_1)\sum_{i=1}^{t}\rho_1^{t-i}\\
=&(1-\rho_1)(1+\cdots+\rho_1^{t-1})\\
=&1-\rho_1^t
\end{aligned}
$$

- Thus

$$E[\boldsymbol{s}_t] = E[\boldsymbol{g}_t](1-\rho_1^t)$$

and they do

$$\hat{\boldsymbol{s}} \leftarrow \frac{\boldsymbol{s}}{1-\rho_1^t}$$

Bias Correction in Adam VI

- The above derivation on bias correction partially follows from `https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimiz`
- The situation for $E[\boldsymbol{g}_t \odot \boldsymbol{g}_t]$ is similar

The Importance of Bias Correction I

- An interesting story is that BERT (Devlin et al., 2019), an important NLP technique using Adam, forgot to do bias correction
- This seems to cause lengthy iterations
- See Zhang et al. (2021) for discussing this issue

Weight Decay I

- Recall in our earlier description, the simple stochastic gradient update is

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta\left(\frac{\boldsymbol{\theta}}{C} + \frac{1}{|S|}\nabla_{\boldsymbol{\theta}} \sum_{i:i\in S} \xi(\boldsymbol{\theta}; \boldsymbol{y}^i, Z^{1,i})\right)$$

- In this rule,

$$\frac{\boldsymbol{\theta}}{C}$$

comes from the regularization term $\boldsymbol{\theta}^T\boldsymbol{\theta}/(2C)$ in $f(\boldsymbol{\theta})$

Weight Decay II

- The use of regularization follows from standard machine learning settings
- However, in the area of neural networks, this term may come from a setting called weight decay (Hanson and Pratt, 1988)

$$\boldsymbol{\theta} \leftarrow (1-\lambda)\boldsymbol{\theta} - \eta\left(\frac{1}{|S|}\nabla_{\boldsymbol{\theta}} \sum_{i:i\in S} \xi(\boldsymbol{\theta}; \boldsymbol{y}^i, Z^{1,i})\right)$$

  where λ is the rate of weight decay
- In fact, Hanson and Pratt (1988) did not give good reasons for decaying the weight of $\boldsymbol{\theta}$

Weight Decay III

- Clearly, if

$$\lambda = \frac{\eta}{C}$$

  then weight decay is the same as regularization
- However, as pointed out in Loshchilov and Hutter (2019), the equivalence does not hold if adaptive learning rate is used

Weight Decay IV

- For example, in AdaGrad, the update rule is

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \frac{\epsilon}{\sqrt{\boldsymbol{r}} + \delta} \odot \left(\frac{1}{|S|} \nabla_{\boldsymbol{\theta}} \sum_{i:i \in S} \xi(\boldsymbol{\theta}; \boldsymbol{y}^i, Z^{1,i})\right)$$
$$- \frac{\epsilon}{\sqrt{\boldsymbol{r}} + \delta} \odot \frac{\boldsymbol{\theta}}{C}$$

  so the regularization term is scaled in a component-wise way

- Loshchilov and Hutter (2019) advocate to decouple the weight decay step

Weight Decay V

- For example, for the momentum algorithm

$$\boldsymbol{v} \leftarrow \alpha \boldsymbol{v} - \eta\left(\frac{\boldsymbol{\theta}}{C} + \frac{1}{|S|}\nabla_{\boldsymbol{\theta}} \sum_{i:i\in S} \xi(\boldsymbol{\theta}; \boldsymbol{y}^i, Z^{1,i})\right)$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \boldsymbol{v}$$

they prefer the following equivalent form

$$\boldsymbol{v} \leftarrow \alpha \boldsymbol{v} - \eta\left(\frac{1}{|S|}\nabla_{\boldsymbol{\theta}} \sum_{i:i\in S} \xi(\boldsymbol{\theta}; \boldsymbol{y}^i, Z^{1,i})\right)$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \boldsymbol{v} - \eta\frac{\boldsymbol{\theta}}{C}$$

Weight Decay VI

- Based on this, Loshchilov and Hutter (2019) proposed AdamW

AdamW I

$$
\begin{aligned}
\boldsymbol{g} &\leftarrow \frac{1}{|S|}\nabla_\theta \sum_{i:i\in S} \xi(\boldsymbol{\theta}; \boldsymbol{y}^i, Z^{1,i}) \\
\boldsymbol{s} &\leftarrow \rho_1 \boldsymbol{s} + (1-\rho_1)\boldsymbol{g} \\
\boldsymbol{r} &\leftarrow \rho_2 \boldsymbol{r} + (1-\rho_2)\boldsymbol{g} \odot \boldsymbol{g} \\
\hat{\boldsymbol{s}} &\leftarrow \frac{\boldsymbol{s}}{1-\rho_1^t} \\
\hat{\boldsymbol{r}} &\leftarrow \frac{\boldsymbol{r}}{1-\rho_2^t} \\
\boldsymbol{\theta} &\leftarrow \boldsymbol{\theta} - \frac{\epsilon}{\sqrt{\hat{\boldsymbol{r}}}+\delta} \odot \hat{\boldsymbol{s}} - \epsilon\frac{\boldsymbol{\theta}}{C}
\end{aligned}
$$

AdamW II

- This is not equivalent to Adam because in Adam, $\boldsymbol{\theta}/C$ has been used in calculating $\boldsymbol{g}$ and then scaled after
- Why is the decoupled setting better? Some discussions are in Section 3 of Loshchilov and Hutter (2019)

Choosing Stochastic Gradient Algorithms

- From Goodfellow et al. (2016), "there is currently no consensus"
- Further, "the choice ... seemed to depend on the user's familiarity with the algorithm"

Why Stochastic Gradient Widely Used? I

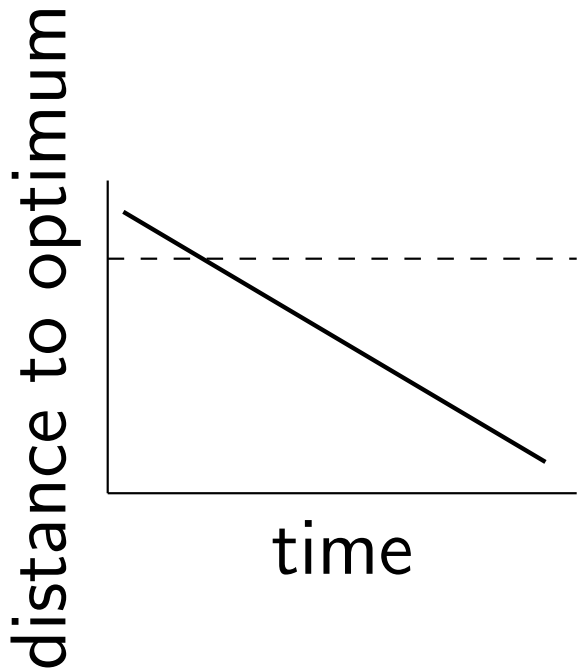
- In machine learning fast final convergence may not be important
  - An optimal solution $\boldsymbol{\theta}^*$ may not lead to the best model
  - Further, we don't need a point close to $\boldsymbol{\theta}^*$. In prediction we find

$$\arg\max_k z_k^{L+1}(\boldsymbol{\theta})$$

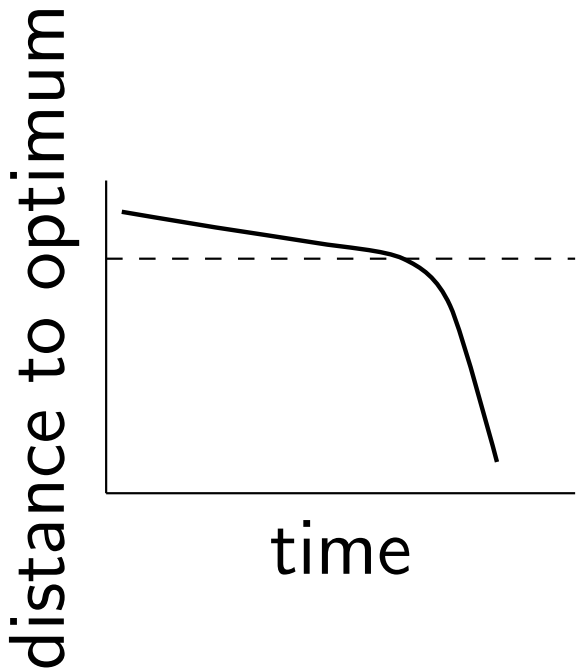
A not-so-accurate $\boldsymbol{\theta}$ may be good enough

An illustration (modified from Tsai et al. (2014))

Why Stochastic Gradient Widely Used? II



Slow final convergence



Fast final convergence

Why Stochastic Gradient Widely Used? III

- The special property of data classification is essential

$$E(\nabla_\theta \xi(\boldsymbol{\theta}; \boldsymbol{y}^i, Z^{1,i})) = \frac{1}{l} \nabla_\theta \sum_{i=1}^{l} \xi(\boldsymbol{\theta}; \boldsymbol{y}^i, Z^{1,i})$$

- We can cheaply get a good approximation of the gradient
- Indeed stochastic gradient is less used outside machine learning

Why Stochastic Gradient Widely Used? IV

- Easy implementation. It's simpler than methods using, for example, second derivative

  Now for complicated networks, (subsampled) gradient is calculated by automatic differentiation
- We will explain more about this
- Non-convexity plays a role
  - For convex, other methods may possess advantages to more efficiently find the global minimum
  - But for non-convex, efficiency to reach a stationary point is less useful

Why Stochastic Gradient Widely Used? V

- A global minimum usually gives a good model (as loss is minimized), but for a stationary point we are less sure
- Some variants of SG have been proposed to improve the robustness or the convergence
- All these explain why SG is popular for deep learning

References I

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In J. Burstein, C. Doran, and T. Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*, pages 4171–4186, 2019. doi: 10.18653/v1/n19-1423.

I. J. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. The MIT Press, 2016.

S. Hanson and L. Pratt. Comparing biases for minimal network construction with back-propagation. In *Advances in Neural Information Processing Systems*, volume 1, 1988.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2015.

I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *Proceedings of International Conference on Learning Representations*, 2019.

C.-H. Tsai, C.-Y. Lin, and C.-J. Lin. Incremental and decremental training for linear classification. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/ws/inc-dec.pdf`.

A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht. The marginal value of adaptive gradient methods in machine learning. In *Advances in Neural Information Processing Systems*, pages 4148–4158, 2017.

T. Zhang, F. Wu, A. Katiyar, K. Q. Weinberger, and Y. Artzi. Revisiting few-sample BERT fine-tuning. In *Proceedings of International Conference on Learning Representations*, 2021.