# Topic 27: Optimization Basics

Yanwu Gu

Dept. of Math, HKUST

21/11/2023

# Abstract

Optimizers and learning rate schedules used for training generative AI.

- AdaGrad is a milestone paper of adaptive gradient methods, which is especially efficient for learning on rarely occurred features.
- Its momentum extensions named Adam is currently the dominant optimizer in deep learning. AdamW proposes the correct way of combining Adam and weight decay, which is currently the most popular optimizer used in training transformer models.
- Cosine decay (SGDR) is one of the most used modern learning rate schedule.

# Gradient Descent

---

**Algorithm 1** Gradient Descent

---

**Require:** Initial parameter $w_1$, learning rate $\eta_t$, number of iterations $T$, loss function $f_t$

1: **for** $t = 1$ to $T$ **do**
2: Receive data $(x_t, y_t)$ from observation
3: Compute the gradient: $g_t \leftarrow \nabla_w f_t(x_t, y_t; w_t)$
4: Update the parameters: $w_{t+1} \leftarrow \Pi_{\mathcal{K}}(w_t - \eta g_t)$
5: **end for**
6: **return** final parameter vector $w_{T+1}$

---

The gradient information.

# AdaGrad

**Algorithm 2** AdaGrad with full matrices

**Require:** $\eta > 0,\ \delta \geq 0, S_t, H_t, G_t \in \mathbb{R}^{d\times d},\ x_1 = 0, S_0 = H_0 = G_d = 0$

1: **for** $t = 1$ to $T$ **do**

2: Receive loss $f_t(x_t)$, subgradient $g_t \in \partial f_t(x_t)$

3: Upgrade: $G_t = G_{t-1} + g_t g_t^T, S_t = G_t^{1/2}$

4: Set $H_t = \delta I + S_t, \Psi_t(x) = \frac{1}{2}\langle x, H_t x\rangle$

5: Primal-dual subgradient update:

$$x_{t+1} = \arg\min_{x\in\mathcal{X}}\{\eta\langle\frac{1}{t}\sum_{\tau=1}^{t} g_\tau, x\rangle + \eta\varphi(x) + \frac{1}{t}\psi_t(x)\}$$

6: **end for**

7: **return** final parameter vector $w_{T+1}$

The second-order information of the gradient.

# AdaGrad

Dychi, Hazan and Singer [1] state that the regret of AdaGrad satisfies that

$$R_\phi(T) \leq \frac{\delta}{\eta}\|x^*\|_2^2 + \frac{1}{\eta}\|x^*\|_2^2 \mathsf{tr}(G_T^{1/2}) + \eta \mathsf{tr}(G_T^{1/2})$$

**Algorithm 3** Adaptive Moment Estimation (Adam)

**Require:** setpsize $\alpha$, decay rates $\beta_1, \beta_2 \in [0,1)$, loss function $f_t$
**Require:** $\theta_0 = 0$, $m_0 = 0$ and $v_0 = 0$

1: **for** $t = 1$ to $T$ **do**
2: $\quad g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$
3: $\quad m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1) g_t$
4: $\quad v_t \leftarrow \beta_2 v_{t-1} + (1-\beta_2) \|g_t\|_2^2$
5: $\quad \hat{m}_t \leftarrow m_t / (1-\beta_1^t)$
6: $\quad \hat{v}_t \leftarrow v_t / (1-\beta_2^t)$
7: $\quad \theta_t \leftarrow \theta_{t-1} - \alpha \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$
8: **end for**
9: **return** $\theta_T$

The gradient momentum information.

# Adam

King ma and Ba [2] state that the regret of Adam satisfies that

$$
\begin{aligned}
R(T) \leq & \frac{D^2}{2\alpha(1-\beta_1)} \sum_{i=1}^{d} \sqrt{T \hat{v}_{T,i}} \\
& + \frac{\alpha(1+\beta_1)G_\infty}{(1-\beta)\sqrt{1-\beta_2}(1-\gamma)^2} \sum_{i=1}^{d} \|g_{1:T,i}\|_2 \\
& + \sum_{i=1}^{d} \frac{D_\infty^2 G_\infty \sqrt{1-\beta_2}}{2\alpha(1-\beta_1)(1-\lambda)^2}
\end{aligned}
$$

**Algorithm 4** Adam with factored second moments

**Require:** Initial point $X_0 \in \mathcal{R}^{n\times m}$, stepsize $\{\alpha_t\}_{t=1}^T$, secound moment decay $\beta$, regularization constant $\epsilon$, $R_0 = 0$, $C_0 = 0$

1: **for** $t = 1$ to $T$ **do**

2: $\quad G_t = \nabla f_t(X_{t-1})$

3: $\quad R_t = \beta_2 R_{t-1} + (1-\beta_2)(G_t^2)1_m$

4: $\quad C_t = \beta_2 C_{t-1} + (1-\beta_2)1_n^T(G_t^2)$

5: $\quad \hat{V}_t = (R_t C_t / 1_n^T R_t)/(1-\beta_2^T)$

6: $\quad X_t = X_{t-1} - \alpha_t G_t / (\sqrt{\hat{V}_t + \epsilon})$

7: **end for**

# AMSGrad

**Algorithm 5** AMSGrad

**Require:** $x_1 \in \mathcal{F}$, stepsize $\{\alpha_t\}_{t=1}^T, \{\beta_{1t}\}_{t=1}^T, \beta_2$

Set $m_0 = 0, v_0 = 0, \hat{v}_0 = 0$

**for** $t = 1$ to $T$ **do**

$\quad g_t = \nabla f_t(x_t)$

$\quad m_t = \beta_{1t} m_{t-1} + (1 - \beta_{1t}) g_t$

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

$\quad \hat{v}_t = \max(\hat{v}_{t-1}, v_t)$ and $\hat{V}_t = \text{diag}(\hat{v}_t)$

$\quad x_{t+1} = \Pi_{\mathcal{F}, \sqrt{\hat{V}_t}}(x_t - \alpha_t m_t / \sqrt{\hat{v}_t})$

**end for**

# AMSGrad

Reddi, Kale and Kumar [6] state that the regret of AMSGrad satisfies that

$$
\begin{aligned}
R(T) \leq & \frac{D_\infty^2}{\alpha(1-\beta_1)} \sum_{i=1}^{d} \sqrt{T \hat{v}_{T,i}} \\
& + \frac{\alpha(1+\beta_1)\sqrt{1+\log T}}{(1-\beta_1)^2 \sqrt{1-\beta_2}(1-\gamma)} \sum_{i=1}^{d} \|g_{1:T,i}\|_2 \\
& + \frac{D_\infty^2}{(1-\beta_1)^2} \sum_{t=1}^{T} \sum_{i=1}^{d} \frac{\beta_{1t} \hat{v}_{t,i}^{1/2}}{\alpha_t}
\end{aligned}
$$

# AdamW Optimizer

Some analysis of Adam:

- $L_2$ regularization and weight decay are not identical.
- $L_2$ regularization is not effective in Adam.
- Weight decay is equally effective in both SGD and Adam.
- Optimal weight decay depends on the total number of batch passes/weight updates.
- Adam can substantially benefit from a scheduled learning rate multiplier.

# AdamW Optimizer

- The main contribution of Loshchilov and Hutter [5] is to improve regularization in Adam by decoupling the weight decay from the gradient-based update.

# AdamW Optimizer

**Proposition (1)**

*(Weight decay $=L_2$ reg for standard SGD). Standard SGD with base learning rate $\alpha$ executes the same steps on batch loss functions $f_t(\theta)$ with weight decay $\lambda$ as it executes without weight decay on $f_t^{reg}(\theta)=f_t(\theta)+\lambda/2\alpha\cdot\|\theta\|_2^2$.*

$$
\begin{aligned}
\theta_{t+1}^{WD} &= (1-\lambda)\theta_t - \alpha\nabla f_t(\theta_t) \\
\theta_{t+1}^{reg} &= \theta_t - \alpha\nabla f_t^{reg}(\theta_t)
\end{aligned}
$$

# AdamW Ooptimizer

**Algorithm 6** Adam with $L_2$ reg and decoupled weight decay

**Require:** setpsize $\alpha$, decay rates $\beta_1, \beta_2 \in [0, 1)$, loss function $f_t$, $\lambda \in \mathbb{R}$

**Require:** $\theta_0 = 0$, $m_0 = 0$ and $v_0 = 0$

**for** $t = 1$ to $T$ **do**

  $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1}) + \lambda\theta_{t-1}$

  $m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1) g_t$

  $v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2) \|g_t\|_2^2$

  $\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$

  $\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$

  $\theta_t \leftarrow \theta_{t-1} - \alpha(\hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon) + \lambda\theta_{t-1})$

**end for**

**return** $\theta_T$

# AdamW Optimizer

**Proposition (2)**

*(Weight decay $\neq L_2$ reg for adaptive gradients).*
*Let $O_1, O_2$ denote $\theta_{t+1} \leftarrow \theta_t - \alpha M_t \nabla f_t(\theta_t)$ without weight decay, and $\theta_{t+1} \leftarrow (1-\lambda)\theta_t - \alpha M_t \nabla f_t(\theta_t)$ with weight decay, respectively, with $M_t \neq kI$. Then, there exists no $L_2$ coefficient $\lambda'$ such that running $O_1$ on batch loss $f_t^{reg}(\theta) = f_t(\theta) + \lambda'/2 \cdot \|\theta\|_2^2$ without weight decay is equivalent to running $O_2$ on $f_t(\theta)$ with decay $\lambda \in \mathbb{R}^+$.*

# AdamW Optimizer

- For the adaptive gradient algorithm, with regularization $L_2$, the sum of the gradient of **loss function** and **regularizer** are adapted, while with decoupled weight decay, only the sum of the gradient of **loss function** is adapted.
- With $L_2$ regularization both types of gradients are normalized by their typical (summed) magnitude; decoupled weight decay regularizes all weights with the same rate $\lambda$.

# AdamW Optimizer

## Proposition (3)

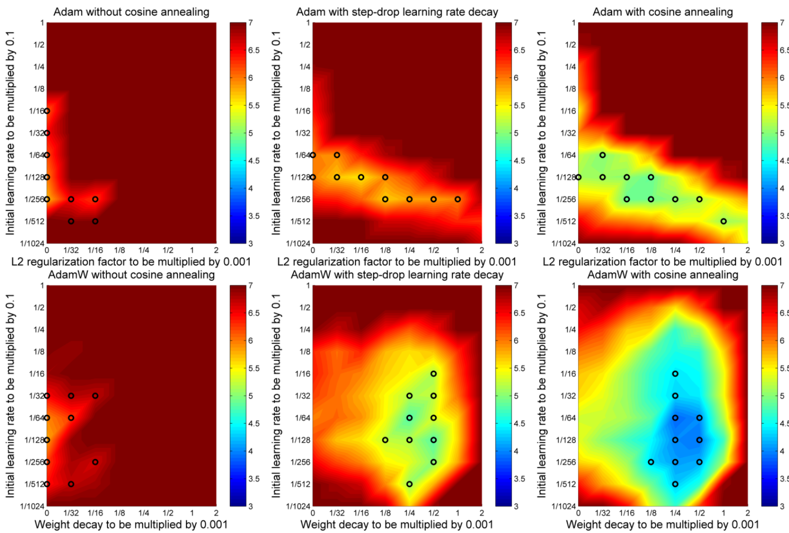
*(Weight decay = scale-adjusted $L_2$ reg for adaptive gradient algorithm with fixed preconditioner).*
*Using a fixed preconditioner matrix $M_t = \text{diag}(s) - 1$ (with $s_i > 0$). Then, $O$ with base learning rate $\alpha$ executes the same steps on batch loss functions $f_t(\theta)$ with weight decay $\lambda$ as it executes without weight decay on the scale-adjusted regularized batch loss*

$$f_t^{sreg}(\theta) = f_t(\theta) + \frac{\lambda'}{2\alpha}\|\theta \odot \sqrt{s}\|_2^2,$$

*where $\lambda' = \lambda/\alpha$ .*

# AdamW Optimizer

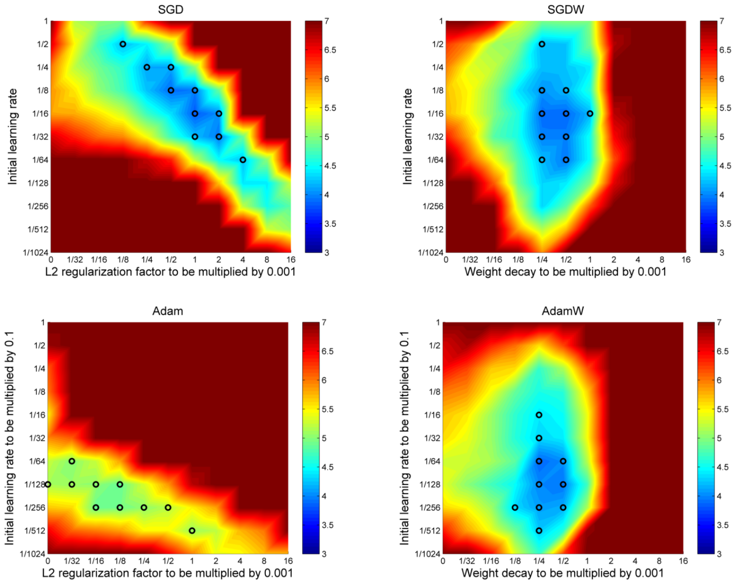
## Evaluating Decoupled Weight Decay With Different Learning Rate Schedules




- Test on a 26 2x64d ResNet on CIFAR-10 after 100 epochs.
- AdamW performs better than Adam.
- AdamW leads a more separable hyperparameter search space.
- Cosine annealing > step-drop learning rate decay > fixed learning rate.

# AdamW Optimizer

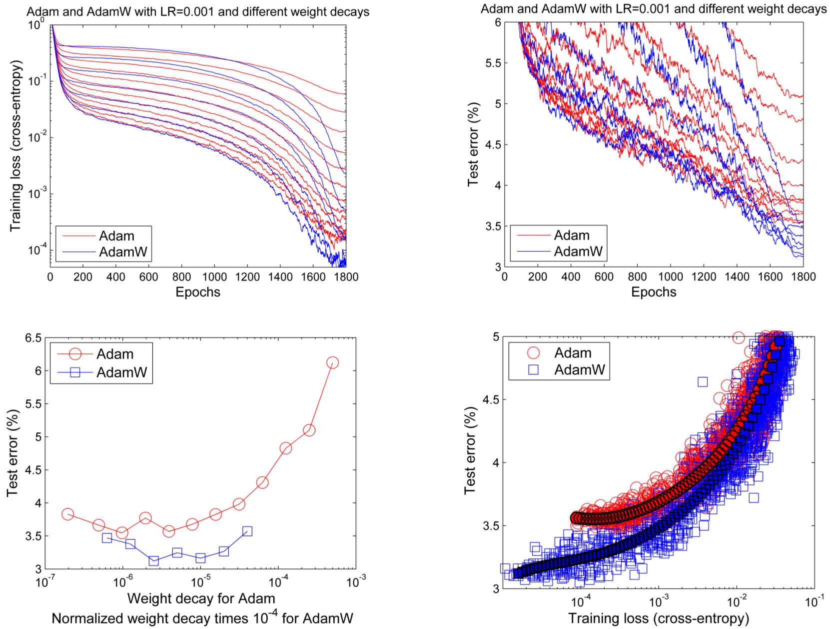
## Decoupling the Weight Decay and Initial Learning Rate Parameters




- Compare the performance of SGD, SGDW, Adam, and AdamW.
- Results support the author's hypothesis that the weight decay and learning rate hyperparameters can be decoupled, simplifying the problem of hyperparameter tuning in SGD, and improving Adam's performance to be competitive w.r.t. SGDM.

# AdamW Optimizer

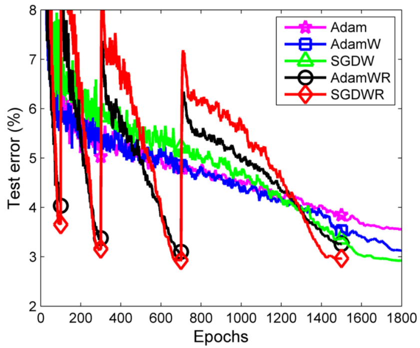
## Better Generalization of AdamW

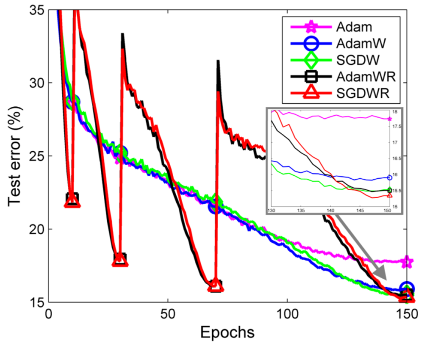



- The learning curves of Adam and AdamW coincided for the first half of the training run, AdamW often led to lower training loss and test errors.
- The results in the bottom right suggest that AdamW did not only yield better training loss but also yielded better generalization performance for similar training loss values.

# AdamW Optimizer

## AdamWR With Warm Restarts for Better Anytime Performance




- AdamWR greatly speeds up AdamW on CIFAR-10 and ImageNet32x32, up to a factor of 10.
- For the default learning rate of 0.001, AdamW achieved 15% relative improvement in test error compared to Adam.
- AdamWR achieved the same improved results, but with much better performance at all times. These improvements closed most of the gap between Adam and SGDWR.

# AdamW Optimizer
## Conclusion

- Identified and exposed the inequivalence of $L_2$ regularization and weight decay for Adam.
- Adam with decoupled weight decay provides significantly improved generalization performance compared to the typical implementation of Adam with $L_2$ regularization.
- Using warm restarts for Adam to improve its anytime performance.

# LAMB Optimizer

**Algorithm 7** LARS (Layerwise Adaptive Rate Scaling)

**Require:** $x_1 \in \mathbb{R}^d$, lr $\eta_t$, $0 < \beta_1 < 1$, scaling func $\phi$, $\epsilon > 0, m_0 = 0$

1: **for** $t = 1$ to $T$ **do**

2: Draw $b$ sample $S_t$ from $\mathbb{P}$

3: $g_t \leftarrow \frac{1}{|S_t|} \sum_{s_t \in S_t} \nabla \ell(x_t, s_t)$

4: $m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1)(g_t + \lambda x_t)$

5: $x_{t+1}^{(i)} = x_t^{(i)} - \eta_t \frac{\phi(\|x_t^{(i)}\|)}{\|m_t^{(i)}\|} m_t^{(i)}$ for all $i \in [h]$

6: **end for**

# LAMB Optimizer

**Algorithm 8** LAMB (Layerwise Adaptive with Mini-Batches)

**Require:** $x_1 \in \mathbb{R}^d$, lr $\eta_t$, $0 < \beta_1, \beta_2 < 1$, scaling func $\phi$, $\epsilon > 0, m_0, v_0 = 0$

**for** $t = 1$ to $T$ **do**

  Draw $b$ sample $S_t$ from $\mathbb{P}$

  $g_t \leftarrow \frac{1}{|S_t|} \sum_{s_t \in S_t} \nabla \ell(x_t, s_t)$

  $m_t \leftarrow \beta m_{t-1} + (1 - \beta_1)(g_t + \lambda x_t)$

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

  $m_t = m_t / (1 - \beta_1^t)$

  $v_t = v_t / (1 - \beta_2^t)$

  $r_t = \frac{m_t}{\sqrt{v_t} + \epsilon}$

  $x_{t+1}^{(i)} = x_t^{(i)} - \eta_t \frac{\phi(\|x_t^{(i)}\|)}{\|r_t^{(i)} + \lambda x_t^{(i)}\|} (r_t^{(i)} + \lambda x_t^{(i)})$ for all $i \in [h]$

**end for**

# LAMB

You et.al. [7] give the bound of $x_t$ generated using LAMB as:

1. When $\beta_2 = 0$,

$$(\mathbb{E}[\frac{1}{\sqrt{d}}\|\nabla f(x_a)\|_1])^2 \leq O(\frac{(f(x_1) - f(x^*))L_{avg})}{T} + \frac{\|\tilde{\sigma}\|_1^2}{Th}),$$

2. When $\beta_2 > 0$,

$$\mathbb{E}[\|\nabla f(x_a)\|^2] \leq O(\sqrt{\frac{G^2 d}{h(1-\beta_2)}} \times [\frac{2(f(x_1) - f(x^*))\|L\|_1}{T} + \frac{\|\tilde{\sigma}\|_1}{\sqrt{T}}]),$$

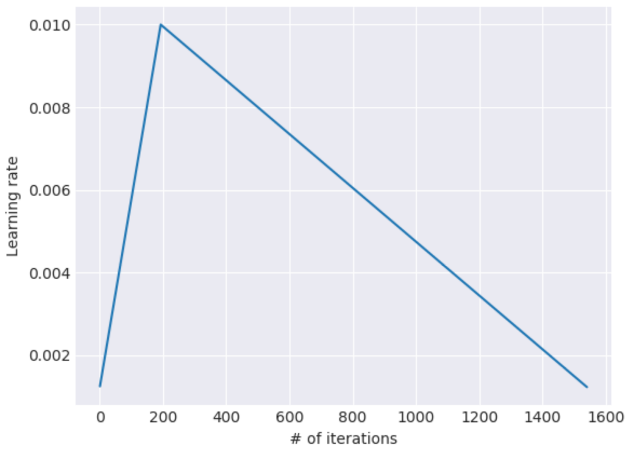
where $x^*$ is an optimal solution to the regularized excepted loss function and $x_a$ is an iterate uniformly randomly chosen from $\{x_1, \ldots, x_T\}$.

# Linear Decay Scheduler

The change of learning rate can be described as:

$$
\begin{aligned}
cut =& \lceil T \cdot cutfrac \rceil \\
p =& \begin{cases} t/cut, & t < cut \\ 1 - \frac{t - cut}{cut \cdot (1/cutfrac - 1)}, & otherwise \end{cases} \\
\eta_t =& \eta_{max} \cdot \frac{1 + p \cdot (ratio - 1)}{ratio}
\end{aligned}
$$

# Linear Decay Scheduler



# SGDR

- The main difficulty in training a DNN is associated with the scheduling of the learning rate and the amount of $L_2$ weight decay regularization employed.
- Loshchilov and Hkutter [4] proposed to periodically simulate warm restarts of SGD, wherein each restart the learning rate is initialized to some value and is scheduled to decrease.
- The results suggest that SGD with warm restarts requires $2\times$ to $4\times$ fewer epochs than the currently used learning rate schedule schemes to achieve comparable or even better results.

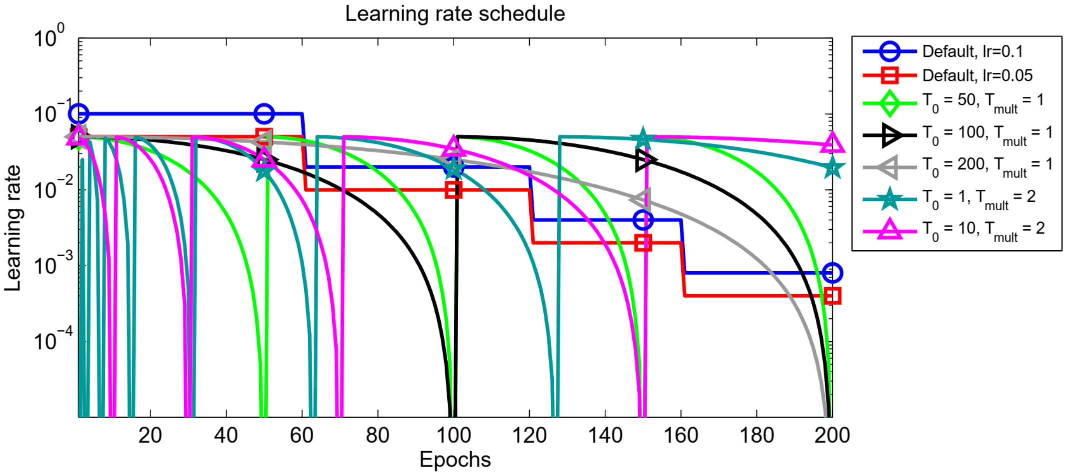
# SGDR

The decay of the learning rate is described as

$$\eta_t = \eta_{min}^i + \frac{1}{2}(\eta_{max}^i - \eta_{min}^i)(1 + \cos(\frac{T_{cur}}{T_i}\pi))$$

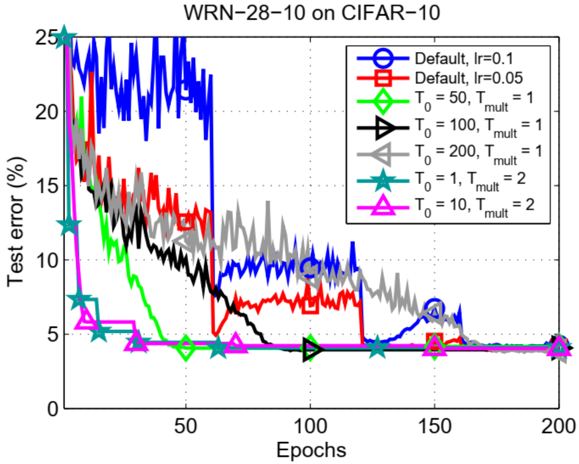
where $\eta_{min}^i$ and $\eta_{max}^i$ is the ranges of the learning rate, and $T_{cur}$ accounts for how many epochs have been performed since the last restart.

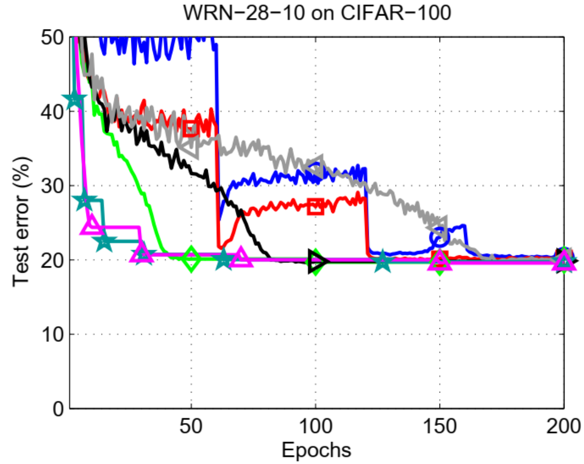
# SGDR



Learning rate schedule
Learning rate
Epochs
$10^0$
$10^{-1}$
$10^{-2}$
$10^{-3}$
$10^{-4}$
20
40
60
80
100
120
140
160
180
200
Default, lr=0.1
Default, lr=0.05
$T_0$ = 50, $T_{mult}$ = 1
$T_0$ = 100, $T_{mult}$ = 1
$T_0$ = 200, $T_{mult}$ = 1
$T_0$ = 1, $T_{mult}$ = 2
$T_0$ = 10, $T_{mult}$ = 2

# SGDR Experiments

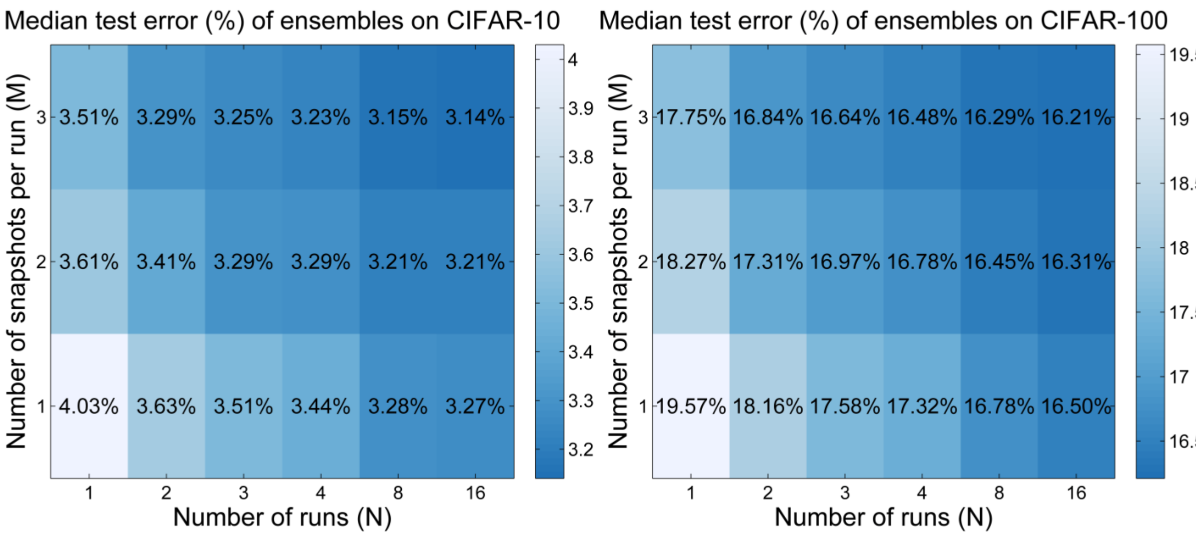


# SGDR Experiments

| | depth-$k$ | # params | # runs | CIFAR-10 | CIFAR-100 |
|---|---|---|---|---|---|
| original-ResNet (He et al., 2015) | 110 | 1.7M | mean of 5 | 6.43 | 25.16 |
| | 1202 | 10.2M | mean of 5 | 7.93 | 27.82 |
| stoc-depth (Huang et al., 2016c) | 110 | 1.7M | 1 run | 5.23 | 24.58 |
| | 1202 | 10.2M | 1 run | 4.91 | n/a |
| pre-act-ResNet (He et al., 2016) | 110 | 1.7M | med. of 5 | 6.37 | n/a |
| | 164 | 1.7M | med. of 5 | 5.46 | 24.33 |
| | 1001 | 10.2M | med. of 5 | 4.62 | 22.71 |
| WRN (Zagoruyko & Komodakis, 2016) | 16-8 | 11.0M | 1 run | 4.81 | 22.07 |
| | 28-10 | 36.5M | 1 run | 4.17 | 20.50 |
| with dropout | 28-10 | 36.5M | 1 run | n/a | 20.04 |
| WRN (ours) | | | | | |
| default with $\eta_0 = 0.1$ | 28-10 | 36.5M | med. of 5 | 4.24 | 20.33 |
| default with $\eta_0 = 0.05$ | 28-10 | 36.5M | med. of 5 | 4.13 | 20.21 |
| $T_0 = 50, T_{mult} = 1$ | 28-10 | 36.5M | med. of 5 | 4.17 | 19.99 |
| $T_0 = 100, T_{mult} = 1$ | 28-10 | 36.5M | med. of 5 | 4.07 | 19.87 |
| $T_0 = 200, T_{mult} = 1$ | 28-10 | 36.5M | med. of 5 | 3.86 | 19.98 |
| $T_0 = 1, T_{mult} = 2$ | 28-10 | 36.5M | med. of 5 | 4.09 | 19.74 |
| $T_0 = 10, T_{mult} = 2$ | 28-10 | 36.5M | med. of 5 | 4.03 | 19.58 |
| default with $\eta_0 = 0.1$ | 28-20 | 145.8M | med. of 2 | 4.08 | 19.53 |
| default with $\eta_0 = 0.05$ | 28-20 | 145.8M | med. of 2 | 3.96 | 19.67 |
| $T_0 = 50, T_{mult} = 1$ | 28-20 | 145.8M | med. of 2 | 4.01 | 19.28 |
| $T_0 = 100, T_{mult} = 1$ | 28-20 | 145.8M | med. of 2 | **3.77** | 19.24 |
| $T_0 = 200, T_{mult} = 1$ | 28-20 | 145.8M | med. of 2 | **3.66** | 19.69 |
| $T_0 = 1, T_{mult} = 2$ | 28-20 | 145.8M | med. of 2 | 3.91 | **18.90** |
| $T_0 = 10, T_{mult} = 2$ | 28-20 | 145.8M | med. of 2 | **3.74** | **18.70** |

# SGDR Experiments



# SGDR Experiments

| | CIFAR-10 | CIFAR-100 |
|---|---|---|
| $N = 1$ run of WRN-28-10 with $M = 1$ snapshot (median of 16 runs) | 4.03 | 19.57 |
| $N = 1$ run of WRN-28-10 with $M = 3$ snapshots per run | 3.51 | 17.75 |
| $N = 3$ runs of WRN-28-10 with $M = 3$ snapshots per run | 3.25 | 16.64 |
| $N = 16$ runs of WRN-28-10 with $M = 3$ snapshots per run | **3.14** | **16.21** |

# SGDR

## Conclusion

- SGDR simulates warm restarts by scheduling the learning rate to achieve competitive results on CIFAR-10 and CIFAR-100 roughly two to four times faster.
- State-of-the-art results with SGDR are achieved, mainly by using even wider WRNs and ensembles of snapshots from SGDR's trajectory.
- SGDR delivers better results with more restarts and more snapshots of the model.
- SGDR might also reduce the problem of learning rate selection because the annealing and restarts of SGDR scan / consider a range of learning rate values.

# Weakness of Adam

- Adam optimizers cannot adapt to the heterogeneous curvatures in different parameter dimensions, which may occur in LLM pre-traning.
- Liu et.al[3] proposed Sophia, Second-order Clipped Stochastic Optimization, a simple scalable second-order optimizer that uses a light-weight estimate of the diagonal Hessian as the preconditioner, to conquer this problem.

# Acknowledgement

Thank you!

John Duchi, Elad Hazan, and Yoram Singer.
Adaptive subgradient methods for online learning and stochastic optimization.
*Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

Diederik P Kingma and Jimmy Ba.
Adam: A method for stochastic optimization.
*arXiv preprint arXiv:1412.6980*, 2014.

Hong Liu, Zhiyuan Li, David Hall, Percy Liang, and Tengyu Ma.
Sophia: A scalable stochastic second-order optimizer for language model pre-training, 2023.

Ilya Loshchilov and Frank Hutter.
Sgdr: Stochastic gradient descent with warm restarts.
*arXiv preprint arXiv:1608.03983*, 2016.

Ilya Loshchilov and Frank Hutter.
Decoupled weight decay regularization.
In *International Conference on Learning Representations*, 2019.

Sashank J Reddi, Satyen Kale, and Sanjiv Kumar.
On the convergence of adam and beyond.
*arXiv preprint arXiv:1904.09237*, 2019.

et al. Yang You, Jing Li.
Large batch optimization for deep learning- training bert in 76 minutes.
*arXiv preprint arXiv:1904.00962*, 2019.