OPT2022: 14th Annual Workshop on Optimization for Machine Learning

Dimension-Reduced Adaptive Gradient Method

Jingyang Li LI_JINGYANG@U.NUS.EDU
National University of Singapore

Pan Zhou ZHOUPAN@SEA.COM
Sea AI Lab

Kuangyu Ding KUANGYUD@U.NUS.EDU
National University of Singapore

Kim-Chuan Toh MATTOHKC@NUS.EDU.SG
National University of Singapore

Yinyu Ye YYYE@STANFORD.EDU
Stanford University

Abstract

Adaptive gradient methods, such as Adam, have shown faster convergence speed than SGD across various kinds of network models at the expense of inferior generalization performance. In this work, we proposed a Dimension-Reduced Adaptive Gradient Method (DRAG) to eliminate the generalization gap. DRAG makes an elegant combination of SGD and Adam by adopting a trust-region like framework. We observe that 1) Adam adjusts stepsizes for each gradient coordinate according to some loss curvature, and indeed decomposes the n-dimensional gradient into n standard basis directions to search; 2) SGD uniformly scales gradient for all gradient coordinates and actually has only one descent direction to minimize. Accordingly, DRAG reduces the high degree of freedom of Adam and also improves the flexibility of SGD via optimizing the loss along k ($\ll n$) descent directions, *e.g.* the gradient direction and momentum direction used in this work. Then per iteration, DRAG finds the best stepsizes for k descent directions by solving a trust-region subproblem whose computational overhead is negligible since the trust-region subproblem is low-dimensional, *e.g.* $k = 2$ in this work. DRAG is compatible with the common deep learning training pipeline without introducing extra hyper-parameters and with negligible extra computation. Moreover, we prove the convergence property of DRAG for non-convex stochastic problems that often occur in deep learning training. Experimental results on representative benchmarks testify the fast convergence speed and also superior generalization of DRAG.

1. Introduction

Training neural networks can be seen as solving the following non-convex optimization problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x}), \tag{1}$$

where f is the loss function and $\boldsymbol{x} \in \mathbb{R}^n$ is the variable. Among all optimizers, Adam [6] is one of the most popular algorithm to solve problem (1). At each training iteration, Adam maintains an exponential moving average (EMA) of first and second moments of stochastic gradient $\boldsymbol{v}_t$ and $\boldsymbol{u}_t$ as

$$\boldsymbol{v}_t = \beta_1\boldsymbol{v}_{t-1} + (1-\beta_1)\boldsymbol{g}_{t-1}, \quad \boldsymbol{u}_t = \beta_2\boldsymbol{u}_{t-1} + (1-\beta_2)\boldsymbol{g}_{t-1}^2,$$

where $\beta_1, \beta_2 \in [0,1]$ are constant and $\boldsymbol{g}_{t-1} := \tilde{\nabla} f(\boldsymbol{x}_{t-1})$ is the stochastic gradient. It adaptively scales the learning rates for each gradient coordinate, and actually minimizes the loss function along n descent directions

$$\boldsymbol{x}_t = \boldsymbol{x}_{t-1} - \eta \frac{\hat{\boldsymbol{v}}_t}{\sqrt{\hat{\boldsymbol{u}}_t} + \nu} = \boldsymbol{x}_{t-1} - \sum_{i=1}^{n} \frac{\eta}{\sqrt{\hat{\boldsymbol{u}}_{t,i}} + \nu} (\hat{\boldsymbol{v}}_{t,i} \boldsymbol{e}_i), \tag{2}$$

where $\hat{\boldsymbol{v}}_t, \hat{\boldsymbol{u}}_t$ are bias-corrected $\boldsymbol{v}_t, \boldsymbol{u}_t$, $\boldsymbol{e}_i$ is the standard basis vector with 1 for dimension i and 0 for all other dimensions. Specifically, Adam adopts a stepsize of $\frac{\eta}{\sqrt{\hat{\boldsymbol{u}}_{t,i}}+\nu}$ for the i-th gradient descent direction $-\hat{\boldsymbol{v}}_{t,i}\boldsymbol{e}_i$.

While adaptive stepsize boosts the convergence of Adam, it weakens the generalization performance due to noise and overfitting. In contrast, SGD generalizes well because it uses a single stepsize for all gradient coordinates and indeed optimizes the loss function only along the gradient direction. One interpretation for their different generalization performance is that Adam's update direction no longer falls into the subspace spanned by all stochastic gradients $\mathrm{span}\{\boldsymbol{g}_0, \cdots, \boldsymbol{g}_t\}$ [17, 23], while SGD do. Actually, Wilson et al. [17] proved that on a binary classification problem, SGD converges to the max-margin solution because its update at each step is linear combination of stochastic gradients, while adaptive gradient methods converge to solutions that generalize poorly because adaptivity makes the algorithm susceptible to noises and therefore causes overfitting.

To overcome the issue just mentioned, motivated by DRSOM [22], we proposed DRAG algorithm to optimize the loss function in (1) from the gradient direction and the momentum direction. It maintains flexibility in the update direction while inheriting the generalization capacity of SGD. At each step, it searches for the optimal stepsizes along these two directions by solving a two-dimensional trust-region subproblem. Therefore, from the optimization perspective, it conducts the optimal update within the two-dimensional subspace spanned by gradient direction and momentum direction. Moreover, while DRAG adopts the trust-region framework, it is compatible with the dominant deep learning training pipeline without introducing extra hyperparameters.

2. Method

As described in Algorithm 1, at each iteration DRAG first computes stochastic gradient $\boldsymbol{g}_{t-1}$, and use it to update the first moment $\boldsymbol{v}_t$ and second moment $\boldsymbol{u}_t$ of stochastic gradient like Adam. Then, we introduce the bias-corrected second moment $\hat{\boldsymbol{u}}_t$ to approximate the Hessian. In this way, DRAG constructs the trust-region subproblem in line 9 of Algorithm 1. While solving this trust-region subproblem in high-dimensional parameter space is computational expensive, DRAG solves it in the two-dimensional subspace spanned by bias-corrected first moment direction $\hat{\boldsymbol{v}}_t$ and momentum direction $\boldsymbol{d}_{t-1}$, making the computational overhead negligible. Here we intuitively set the trust-region radius as $\eta\|\hat{\boldsymbol{v}}_t\|$, and the benefits of this setting is described in Section 2.1. After calculating the solution α_{1t} and α_{2t} of the subproblem, we get an optimal update $\boldsymbol{p} = -\alpha_{1t}\hat{\boldsymbol{v}}_t + \alpha_{2t}\boldsymbol{d}_{t-1}$ in the two-dimensional subspace. Finally, we follow [9] and conduct a decoupled weight decay step. This is the overall framework of our DRAG.

The only extra computational overhead of DRAG compared with Adam is solving the two-dimensional trust-region subproblem in line 9 of Algorithm 1. The trust-region subproblem can be

Algorithm 1 Dimension-Reduced Adaptive Gradient Method (DRAG)

```
Input: Total number of training epoch m, learning rate η, exponential moving average coefficients β_1, β_2, weight decay scale γ, margin coefficient ν.
Initialize: Set x_0, v_0 = 0, u_0 = 0.
for t = 1, ··· , m do
    Compute stochastic gradient g_{t-1} = ∇̃f(x_{t-1}).
    v_t = β_1 v_{t-1} + (1 − β_1) g_{t-1}, v̂_t = v_t/(1 − β_1^t)
    u_t = β_2 u_{t-1} + (1 − β_2) g_{t-1}^2, û_t = u_t/(1 − β_2^t)
    H_t = diag(√û_t + ν)
    d_{t-1} = x_{t-1} − x_{t-2} if t ≥ 2 else d_{t-1} = 0.
    (α_{1t}, α_{2t}) = argmin_p {⟨v̂_t, p⟩ + 1/2 ⟨p, H_t p⟩ | ‖p‖ ≤ η‖v̂_t‖, p = −α_1 v̂_t + α_2 d_{t-1}}.
    x_t = x_{t-1} − α_{1t} v̂_t + α_{2t} d_{t-1}
    x_t = x_t − ηγ x_{t-1} (Conduct weight decay)
end for
Output: x_1, ··· , x_m
```

formally formulated as follows:

$$
\begin{aligned}
\min_{\alpha_1,\alpha_2} \quad & \langle \hat{\boldsymbol{v}}_t, -\alpha_1\hat{\boldsymbol{v}}_t + \alpha_2\boldsymbol{d}_{t-1}\rangle + \frac{1}{2}\langle -\alpha_1\hat{\boldsymbol{v}}_t + \alpha_2\boldsymbol{d}_{t-1}, \boldsymbol{H}_t(-\alpha_1\hat{\boldsymbol{v}}_t + \alpha_2\boldsymbol{d}_{t-1})\rangle \\
= \quad & \begin{bmatrix}\alpha_1 & \alpha_2\end{bmatrix}\begin{bmatrix}-\hat{\boldsymbol{v}}_t^T\hat{\boldsymbol{v}}_t \\ \hat{\boldsymbol{v}}_t^T\boldsymbol{d}_{t-1}\end{bmatrix} + \frac{1}{2}\begin{bmatrix}\alpha_1 & \alpha_2\end{bmatrix}\begin{bmatrix}\hat{\boldsymbol{v}}_t^T\boldsymbol{H}_t\hat{\boldsymbol{v}}_t & -\hat{\boldsymbol{v}}_t^T\boldsymbol{H}_t\boldsymbol{d}_{t-1} \\ -\hat{\boldsymbol{v}}_t^T\boldsymbol{H}_t\boldsymbol{d}_{t-1} & \boldsymbol{d}_{t-1}^T\boldsymbol{H}_t\boldsymbol{d}_{t-1}\end{bmatrix}\begin{bmatrix}\alpha_1 \\ \alpha_2\end{bmatrix} \\
\text{s.t.} \quad & \|-\alpha_1\hat{\boldsymbol{v}}_t + \alpha_2\boldsymbol{d}_{t-1}\| \le \eta\|\hat{\boldsymbol{v}}_t\|,
\end{aligned}
$$

where $\boldsymbol{H}_t = \mathrm{diag}(\sqrt{\hat{\boldsymbol{u}}_t} + \nu)$ as defined in Algorithm 1. This two-dimensional subproblem can be solved efficiently by using its global minimal condition. In Appendix A, we transform this subproblem into a standard trust-region subproblem, and then an ϵ-global primal-dual solution satisfying KKT condition can be found in $\mathcal{O}(\log\log(\frac{1}{\epsilon}))$ time [10]. See more details in Appendix A.

2.1. Benefits of our algorithm

Flexibility of update As in Algorithm 1, DRAG updates the variable $\boldsymbol{x}$ along EMA of gradient direction $\hat{\boldsymbol{v}}_t$ and momentum direction $\boldsymbol{d}_{t-1}$. This update direction choice acts as a trade-off between the whole space search of Adam and one direction search of SGD. Moreover, the update of DRAG lies in the subspace $\mathrm{span}\{\hat{\boldsymbol{v}}_t, \boldsymbol{d}_{t-1}\} \in \mathrm{span}\{\boldsymbol{g}_0, \cdots, \boldsymbol{g}_{t-1}\}$. This means that the parameter update direction is always a combination of stochastic gradients. According to Wilson et al. [17], this property makes DRAG always converge to the max-margin solution of the binary classification problem, which has the best generalization capacity. This helps to explain DRAG's excellent generalization performance in practice.

Optimal stepsizes DRAG solves the dimension-reduced subproblem at each training epoch and finds the best update along the gradient direction and momentum direction. This optimal update is evaluated by the quadratic approximation to the loss function, where the Hessian is approximated by second moment $\sqrt{\hat{\boldsymbol{u}}_t}$ and gradient is approximated by first moment $\hat{\boldsymbol{v}}_t$. Since DRAG conducts optimal update along gradient and momentum direction within the learning rate we set, it converges faster than SGD on training dataset and is comparable with adaptive gradient methods.

Heuristic trust-region radius We set the trust-region radius for the subproblem as $\eta\|\hat{\boldsymbol{v}}_t\|$. The intuition is that when gradient is large, we hope our algorithm can make a larger step to minimize the loss function significantly. While when gradient is small, we hope our method to be stable and don't change the parameters too much. This heuristic design not only frees us from changing the radius at each step as trust region method does, but also make our algorithm compatible well with dominant deep learning training pipeline without introducing extra hyperparameters.

2.2. Convergence Analysis

For the analysis of stochastic non-convex algorithm, we follow the works Guo et al. [2], Zhuang et al. [24] and make the following necessary assumption.

Assumption 1 *For non-convex problem* $\min_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x})$*, we assume the loss* $f(\boldsymbol{x})$ *satisfies*

- f *is L-Lipschitz smooth.*
- *The gradient estimation* $\boldsymbol{g}$ *is unbiased, namely* $\mathbb{E}[\boldsymbol{g}_t] = \nabla f(\boldsymbol{x}_t)$*, and its variance can be bounded as* $\mathbb{E}[\|\boldsymbol{g}_t - \nabla f(\boldsymbol{x}_t)\|^2] \le \sigma^2$.

Then we can derive the convergence of our proposed algorithm and also provide its stochastic gradient complexity to find an ϵ-approximate first-order stationary point.

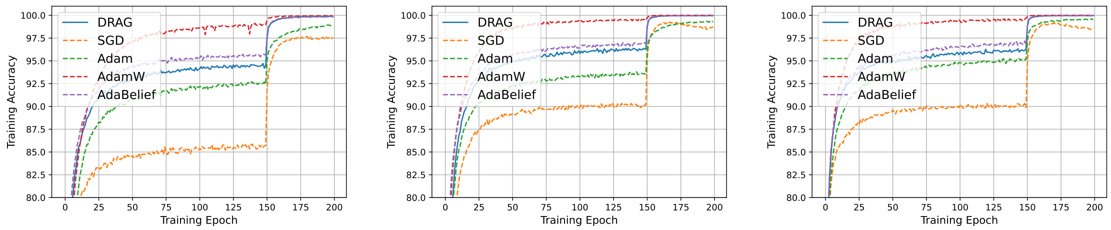
Theorem 1 *Suppose Assumption 1 holds. Let* $\beta_t = \beta$ *and* $\eta_t = \eta$ *for all* t*. Assume there exist constants* $\alpha, G > 0$*, such that* $\alpha \le \min_t \alpha_{1t}$ *and* $\alpha_{1t} \le \eta G$*,* $|\alpha_{2t}| \le \eta G$*. In addition,* $\eta \le \min\left\{\frac{1}{2LG}, \left(\frac{(1-\beta)^2\alpha}{8GL^2}\right)^{\frac{1}{3}}, \left(\frac{\alpha^2}{96G^2}\right)^{\frac{1}{4}}, \left(\frac{\alpha}{48LG^2}\right)^{\frac{1}{4}}, \left(\frac{\alpha}{192L^2G^3}\right)^{\frac{1}{5}}\right\}$*. Then, if* $1-\beta \le \frac{\epsilon^2}{3C_2\sigma^2}$ *and* $T \ge \max\left\{\frac{3C_1}{\alpha\epsilon^2}, \frac{3C_3}{(1-\beta)\epsilon^2}\right\}$*, DRAG can achieve*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\nabla f(\boldsymbol{x}_t)\|^2\right] \le \epsilon^2,\ \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\boldsymbol{v}_t\|^2\right] \le 8\epsilon^2, \tag{3}$$

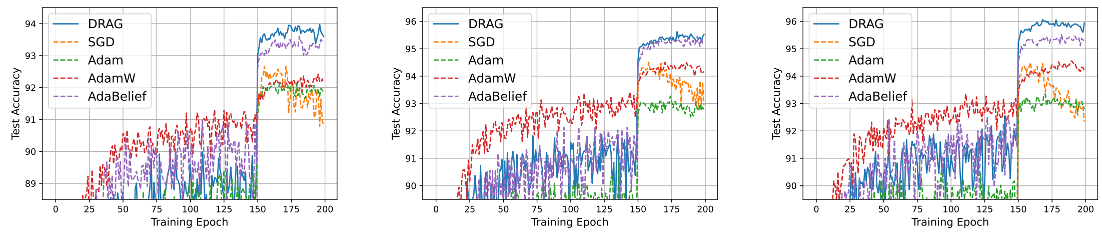
where $C_1 = 4\left(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*)\right)$*,* $C_2 = \frac{4\eta G}{\alpha}$ *and* $C_3 = \frac{2\eta G\mathbb{E}[\|\nabla f(\boldsymbol{x}_0)-(1-\beta_0)\boldsymbol{g}_0\|^2]}{\alpha}$.

Remark 1 *Theorem 1 with its proof in Appendix C demonstrates that by properly selecting constant trust-region radius* η_t *and constant momentum parameter* β_t *(correspond to* β_1 *in Algorithm 1), DRAG can converge to an* ϵ*-approximate first-order stationary point of the non-convex stochastic problem with stochastic gradient complexity* $\mathcal{O}(\epsilon^{-4})$*. Note that the assumptions on* α_{1t} *and* α_{2t} *are mild with the design of DRAG, see details in Appendix B. The complexity of DRAG is of the same order as the lower bound provided by Arjevani et al. [1]. A similar complexity has also been obtained in, for example, LAMB [20], Adam-family [2]. In the analysis of DRAG, we only need a unbiased and variance-bounded stochastic gradient, without any large mini-batch sizes requirement as in LARS [19] and LAMB [20]. In addition, some previous works [8, 11, 14, 21] require the momentum parameter* β_t *to be very close or decreasing to zero. In contrast, DRAG requires* β_t *to be close to one, which is more consistent with the practice.*

Proof of Theorem 1 and more convergence analysis of DRAG can be found in Appendix C.



(a) VGG16 on CIFAR10. (b) ResNet34 on CIFAR10. (c) DenseNet121 on CIFAR10.



(a) VGG16 on CIFAR10. (b) ResNet34 on CIFAR10. (c) DenseNet121 on CIFAR10.

Figure 1: Training and test accuracy of CNNs on CIFAR10 dataset.

3. Experiments

We conduct experiments on several representative benchmarks, including VGG [15], ResNet [3], DenseNet [5] on CIFAR10, CIFAR100 dataset [7], and LSTM [4] on the Penn Treebank dataset [12]. We compare our algorithm DRAG with some popular deep learning optimizers, including SGD [13], Adam [6], AdamW [9], AdaBound [11], AdaBelief [24], RAdam [8], Yogi [21]. Experimental results show that DRAG has faster convergence speed compared with SGD and it achieves state-of-the-art generalization performance. We also conduct ablation study to show 1) two search directions (DRAG) performs better than one direction and multiple directions and 2) DRAG is robust to different learning rate schedules. Details of the ablation study is in Appendix D.

3.1. CNNs on image classification

We conducted experiments for VGG16 with Batch Normalization, ResNet34, and DenseNet121 on CIFAR10 an CIFAR100 dataset. The experimental setting is borrowed from AdaBelief [24] and we also use their default setting for all the hyperparameters. For DRAG, we choose its learning rate to be the same as in SGD, which is 0.1, and weight decay factor is 0.0015 for CIFAR10 and 0.0025 for CIFAR100. Other hyperparameters of DRAG is the same as the default setting ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$). As Figure 1 shows, DRAG has convergence speed comparable with adaptive gradient methods and it attains the best generalization performance. To be specific, DRAG obtains more than 0.5% generalization accuracy gain over AdaBelief [24] on most tasks. The detailed test accuracy is summarized in Table 1.

The possible reasons for this improvement on the convergence speed and generalization capacity is 1) DRAG searches for the optimal update along two directions and thus converges faster, 2) DRAG confines the search of update within the two-dimensional subspace spanned by gradient and momentum direction to avoid overfitting and alleviating the influence of noises, therefore it generalizes better.

Table 1: Top-1 test accuracy (%) of VGG16, ResNet34, DenseNet121 on CIFAR10 and CIFAR100.

		DRAG	SGD	Adam	AdamW	AdaBelief
CIFAR10	VGG16	**94.0**	92.7	92.2	92.4	93.6
	ResNet34	**95.6**	94.5	93.3	94.5	95.4
	DenseNet121	**96.1**	94.5	93.3	94.6	95.5
CIFAR100	VGG16	**72.8**	69.7	62.2	68.5	72.2
	ResNet34	**77.6**	75.6	73.0	70.9	76.1
	DenseNet121	**79.2**	77.8	73.7	74.3	78.2

Table 2: Test perplexity (lower is better) of 1-layer, 2-layer, and 3-layer LSTM on PTB dataset. All results except DRAG and SGD are reported by Adabelief [24].

	DRAG	SGD	AdaBound	Adam	AdamW	AdaBelief	RAdam	Yogi
1-layer	**82.5**	83.0	84.3	85.1	87.7	84.8	86.5	86.5
2-layer	**65.6**	66.1	67.5	67.4	72.8	66.3	72.3	71.3
3-layer	**61.0**	61.8	63.6	64.3	69.9	61.8	70.0	67.5

3.2. LSTMs on language modeling

We experiment with LSTM on the Penn Treebank dataset and record the perplexity (lower is better). We follow the exact experimental setting in Adabelief [24] and use their default hyperparameters except for SGD. For SGD, we use the same hyperparameters as DRAG to make a fair comparison between the two. For SGD and DRAG, we set their learning rate as 25, 75, 75 for 1,2,3-layer LSTM and weight decay factor as 2.5×10^{-6}. SGD's generalization performance in our setting is better than the results provided by Zhuang et al. [24]. From Table 2, we can see that DRAG attains more than 0.5 less perplexity than other optimizers. The good generalization performance may be due to DRAG's two-direction search. The gradient direction inherits SGD's good generalization property and the extra momentum direction further improves its performance.

4. Conclusion

In this paper we propose the DRAG algorithm, which finds the optimal update of the parameters along gradient and momentum directions at each iteration. Compared with Adam, DRAG reduces the flexibility of update direction from searching in the whole parameter space to updating in a two-dimensional subspace, therefore is less susceptible to overfitting and has better generalization performance. Compared with SGD, DRAG inherits the gradient update direction and also update along an extra momentum direction, thus it has faster convergence speed and comparable generalization capacity. Our algorithm can be further generalized to any number of search directions and any choice of Hessian approximation.

References

[1] Yossi Arjevani, Yair Carmon, John C Duchi, Dylan J Foster, Nathan Srebro, and Blake Woodworth. Lower bounds for non-convex stochastic optimization. *Mathematical Programming*, pages 1–50, 2022.

[2] Zhishuai Guo, Yi Xu, Wotao Yin, Rong Jin, and Tianbao Yang. A novel convergence analysis for algorithms of the adam family. *arXiv preprint arXiv:2112.03459*, 2021.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[4] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9 (8):1735–1780, 1997.

[5] G. Huang, Z. Liu, L. Van Der Maaten, and K. Weinberger. Densely connected convolutional networks. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 4700–4708, 2017.

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

[7] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[8] 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. In *International Conference on Learning Representations*, 2019.

[9] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018.

[10] David G Luenberger, Yinyu Ye, et al. *Linear and nonlinear programming*, volume 2. Springer, 1984.

[11] Liangchen Luo, Yuanhao Xiong, Yan Liu, and Xu Sun. Adaptive gradient methods with dynamic bound of learning rate. In *International Conference on Learning Representations*, 2018.

[12] Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Using Large Corpora*, 273, 1994.

[13] H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

[14] Naichen Shi, Dawei Li, Mingyi Hong, and Ruoyu Sun. Rmsprop converges with proper hyperparameter. In *International Conference on Learning Representations*, 2020.

[15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[16] Mengdi Wang, Ethan X Fang, and Han Liu. Stochastic compositional gradient descent: algorithms for minimizing compositions of expected-value functions. *Mathematical Programming*, 161(1):419–449, 2017.

[17] Ashia C Wilson, Rebecca Roelofs, Mitchell Stern, Nati Srebro, and Benjamin Recht. The marginal value of adaptive gradient methods in machine learning. *Advances in Neural Information Processing Systems*, 30, 2017.

[18] Stephen Wright, Jorge Nocedal, et al. Numerical optimization. *Springer Science*, 35(67-68): 7, 1999.

[19] Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.

[20] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962*, 2019.

[21] Manzil Zaheer, Sashank Reddi, Devendra Sachan, Satyen Kale, and Sanjiv Kumar. Adaptive methods for nonconvex optimization. *Advances in Neural Information Processing Systems*, 31, 2018.

[22] Chuwen Zhang, Dongdong Ge, Bo Jiang, and Yinyu Ye. Drsom: A dimension reduced second-order method and preliminary analyses. *arXiv preprint arXiv:2208.00208*, 2022.

[23] Zijun Zhang, Lin Ma, Zongpeng Li, and Chuan Wu. Normalized direction-preserving adam. *arXiv preprint arXiv:1709.04546*, 2017.

[24] Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar C Tatikonda, Nicha Dvornek, Xenophon Papademetris, and James Duncan. Adabelief optimizer: Adapting stepsizes by the belief in observed gradients. *Advances in Neural Information Processing Systems*, 33:18795–18806, 2020.

[25] Fangyu Zou, Li Shen, Zequn Jie, Weizhong Zhang, and Wei Liu. A sufficient condition for convergences of adam and rmsprop. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11127–11135, 2019.

Appendix A. Solve the trust-region subproblem

Recall the trust region subproblem

$$\begin{aligned} \min_{\alpha} \quad & \langle \alpha, C_t \rangle + \frac{1}{2}\langle \alpha, Q_t \alpha \rangle \\ \text{s.t.} \quad & \sqrt{\langle \alpha, G_t \alpha \rangle} \leq \eta \|\hat{v}_t\|, \end{aligned}$$

where $\alpha := \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix}$, $C_t := \begin{bmatrix} -\hat{v}_t^T \hat{v}_t \\ \hat{v}_t^T d_{t-1} \end{bmatrix}$, $Q_t := \begin{bmatrix} \hat{v}_t^T H_t \hat{v}_t & -\hat{v}_t^T H_t \\ -d_{t-1}^T H_t \hat{v}_t & d_{t-1}^T H_t d_{t-1} \end{bmatrix}$, and $G_t := \begin{bmatrix} \hat{v}_t^T \hat{v}_t & -\hat{v}_t^T d_{t-1} \\ -d_{t-1}^T \hat{v}_t & d_{t-1}^T d_{t-1} \end{bmatrix}$, $H_t = \mathrm{diag}(\sqrt{\hat{u}_t} + \nu)$.

In order to solve this trust region subproblem, we transform it into a standard trust region subproblem with L_2-norm constraint.

When matrix G_t is positive definite, we have

$$\begin{aligned} & G_t = L_t L_t^T \text{ (Cholesky Decomposition)} \\ & \sqrt{\alpha^T G_t \alpha} = \sqrt{(L_t^T \alpha)^T L_t^T \alpha} = \|L_t^T \alpha\| \leq \eta \|\hat{v}_t\|. \end{aligned}$$

So we let $y = L_t^T \alpha$, then $\alpha = L_t^{-T} y$ and the subproblem becomes

$$\begin{aligned} & \min_{y} \quad \langle C_t, L_t^{-T} y \rangle + \frac{1}{2}\langle L_t^{-T} y, Q_t L_t^{-T} y \rangle \\ & \text{s.t.} \quad \|y\| \leq \eta \|\hat{v}_t\| \\ \Longleftrightarrow \quad & \min_{y} \quad \langle L_t^{-1} C_t, y \rangle + \frac{1}{2}\langle y, L_t^{-1} Q_t L_t^{-T} y \rangle \\ & \text{s.t.} \quad \|y\| \leq \eta \|\hat{v}_t\|. \end{aligned}$$

In this way, the trust region subproblem is transformed to a standard spherical constrained quadratic optimization problem and it can be solved efficiently [18].

When $|G_t| = 0$, this means $\hat{v}_t$ is linearly dependent with d_{t-1}. In this case, we solve the one-dimensional subproblem as described in Section 2.

Appendix B. Mild assumptions on α_{1t}, α_{2t}

The trust-region subproblem to be solved in Algorithm 1 has global optimality condition [10] given by

$$\begin{cases} (Q_t + \lambda G_t)\alpha + C_t = 0 \\ Q_t + \lambda G_t \succeq 0 \\ \lambda(\|\alpha\|_{G_t} - \eta \|\hat{v}_t\|) = 0, \quad \lambda \geq 0. \end{cases}$$

By its construction, we know that G_t is positive semidefinite. In practice, numerical issues sometimes make it indefinite, leaving the trust-region subproblem insoluble. Thus, we make an adjustment to G_t

$$G_t = \begin{cases} G_t & \text{if } \lambda_{min} \geq \varepsilon_0 \text{ or } |G_t| = 0 \\ \varepsilon_0 I & \text{o.w.} \end{cases}$$

where λ_{min} is the smallest eigenvalue of G_t. In this way, when $|G_t| \neq 0$, we have

$$\|\alpha\| \leq \|G_t^{-1/2}\| \|\alpha\|_{G_t} \leq \eta \|G_t^{-1/2}\| \|\hat{v}_t\| \leq \eta \frac{\|\hat{v}_t\|}{\sqrt{\varepsilon_0}},$$

which means

$$|\frac{\alpha_{1t}}{\eta}|, |\frac{\alpha_{2t}}{\eta}| \leq \frac{\|\hat{v}_t\|}{\sqrt{\varepsilon_0}}.$$

With the common additional assumption that stochastic gradient $g_t = \tilde{\nabla} f(x_t)$ has bounded L_∞ norm, i.e.$\|g_t\|_\infty \leq G_\infty$, then $\hat{v}_t$ as an moving average of g_t also has bounded norm $\|\hat{v}_t\|$. Therefore, we can see that $|\frac{\alpha_{1t}}{\eta}|, |\frac{\alpha_{2t}}{\eta}|$ are upper bounded by a constant.

When $|G_t| = 0$, which means d_{t-1} is parallel with $\hat{v}_t$. Then we only need to find the optimal update within the trust-region along gradient direction $\hat{v}_t$. In this case, we manually set $\alpha_{2t} = 0$ in our implementation of DRAG, and then α_1 satisfies $|\alpha_1| \leq \eta$.

From discussions above, we can see the assumption that $|\frac{\alpha_{1t}}{\eta}|, |\frac{\alpha_{2t}}{\eta}|$ are upper bounded in Theorem 1 and Theorem 2 is satisfied given the common assumption that stochastic gradient $g_t = \tilde{\nabla} f(x_t)$ has bounded L_∞ norm. For the simplicity of notations, we directly make assumptions for α_{1t} and α_{2t} in Theorem 1 and Theorem 2.

For the assumption that α_{1t} is positive and $\frac{\alpha_{1t}}{\eta}$ is lower bounded by a constant, we give an explanation here by intuition and empirical results. Gradient direction is what we considered the most important update direction locally, because by the training pipeline of neural networks, stochastic gradients of training parameters are the new information we gain at each iteration. Thus, we consider the update should at least move towards the gradient descent direction rather than move towards the gradient ascent direction. Moreover, from the observations of α_{1t} under all the experimental settings, α_{1t} is always positive and $\frac{\alpha_{1t}}{\eta}$ is always larger than 0.1. Therefore, this assumption on α_{1t} is reasonable based on common sense and holds true in practice.

Appendix C. Convergence analysis in non-convex stochastic optimization

To clarify Assumption 1, we give the following definitions.

Definition 1 *For a differentiable function f, $\boldsymbol{x}$ is said to be an ϵ-approximate first-order stationary point if it satisfies $\|\nabla f(\boldsymbol{x})\| \leq \epsilon$.*

Definition 2 *For a differentiable funtion $f(x)$, it is called L-Lipschitz smooth if it statisfies $\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\| \leq L\|\boldsymbol{x} - \boldsymbol{y}\|$ for a constant $L > 0$ and any $\boldsymbol{x}, \boldsymbol{y}$ in domain of f.*

Except for Theorem 1, we also have the following result which establishes an $\mathcal{O}(\log T/\sqrt{T})$ sub-linear convergence rate for DRAG.

Theorem 2 *Suppose Assumption 1 holds. Assume there exist constants $\delta, G > 0$, such that $0 < \delta \leq \frac{\alpha_{1t}}{\eta_t} \leq G$, $\frac{|\alpha_{2t}|}{\eta_t} \leq G$. Set $\eta_t = \frac{c_\eta}{\sqrt{t+2}}$, $1 - \beta_t = \frac{Cc_\eta}{\sqrt{t+1}}$, for any c_η and C satisfying $C \geq L\sqrt{\frac{8G}{\delta}}$, and $c_\eta \leq \left\{ \frac{1}{\sqrt{2}LG}, \left(\frac{\delta^2}{96G^2}\right)^{\frac{1}{2}}, \left(\frac{\delta}{48LG^2}\right)^{\frac{1}{3}}, \left(\frac{\delta}{192L^2G^3}\right)^{\frac{1}{4}} \right\}$. Then there exist two constant C_1 and C_2 which are independent with T, such that*

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}[\|\nabla f(\boldsymbol{x}_t)\|^2] \leq \frac{C_1}{\sqrt{T}} + \frac{C_2 \log T}{\sqrt{T}}, \quad \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}[\|\boldsymbol{v}_{t+1}\|^2] \quad \leq \frac{8C_1}{\sqrt{T}} + \frac{8C_2 \log T}{\sqrt{T}}.$$

Given a tolerance $\epsilon > 0$, if $T \geq \tilde{\mathcal{O}}(\frac{1}{\epsilon^4})$, we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla f(\boldsymbol{x}_t)\|^2] \leq \epsilon^2,\ \ \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\|\boldsymbol{v}_{t+1}\|^2] \leq 8\epsilon^2.$$

Remark 2 *Theorem 2 establishes an $\mathcal{O}(\log T/\sqrt{T})$ sub-linear convergence rate for DRAG by choosing a decreasing η_t and $1-\beta_t$ with the order $\mathcal{O}(1/\sqrt{t})$. Similar sub-linear convergence rates are also established by Zou et al. [25] for Adam and Guo et al. [2] for Adam-type optimizers. While Zou et al. [25] has restrictions on the second moment momentum parameter β_2. In Theorem 2, we only need β_t (corresponds β_1 in Algorithm 1) to increase to one.*

One key ingredient in our analysis is an existing variance recursion of the stochastic estimator based on moving average, which is given by the following lemma.

Lemma 3 (Variance Recursion [16]) *Suppose Assumption 1 holds, then we have*

$$\mathbb{E}_t[\|v_{t+1} - \nabla f(x_t)\|^2] \leq \beta\|v_t - \nabla f(x_{t-1})\|^2 + 2(1-\beta)^2\mathbb{E}_t[\|g_t - \nabla f(x_t)\|^2] + \frac{L^2\|d_t\|^2}{1-\beta},$$

where $\mathbb{E}_t[\cdot]$ denotes the conditional expectation with respect to all randomness before g_t.

Before proving Theorem 1, we need to prove the following auxiliary lemma.

Lemma 4 *Suppose Assumption 1 holds. Assume there exist $\alpha, \eta, \delta, G > 0$, such that $\alpha \leq \min_t \alpha_{1t}$, $\max_t \eta_t \leq \eta$, and $0 < \delta \leq \frac{\alpha}{\eta} \leq \frac{\alpha_{1t}}{\eta_t} \leq G$, $\frac{|\alpha_{2t}|}{\eta_t} \leq G$, (δ, G) are constants independent with t. In addition, $\eta \leq \min\left\{\frac{1}{2LG}, \frac{1-\beta}{2L}\sqrt{\frac{\delta}{2G}}, \frac{\delta}{4\sqrt{6}G}, \left(\frac{\delta}{48LG^2}\right)^{\frac{1}{3}}, \left(\frac{\delta}{192L^2G^3}\right)^{\frac{1}{4}}\right\}$. Then there exist positive constants C_1, C_2 and C_3, which are all independent with T, such that the following estimation holds:*

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\nabla f(x_t)\|^2\right] &\leq \frac{C_1}{T\alpha} + C_2(1-\beta)\sigma^2 + \frac{C_3}{T(1-\beta)},\\ \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|v_t\|^2\right] &\leq \frac{8C_1}{T\alpha} + 8C_2(1-\beta)\sigma^2 + \frac{8C_3}{T(1-\beta)}.\end{aligned} \tag{4}$$

Proof Since F is L-smooth, we have

$$\begin{aligned}f(x_{t+1}) &\leq f(x_t) + \langle\nabla f(x_t), -\alpha_{1t}v_{t+1} + \alpha_{2t}d_t\rangle + \frac{L}{2}\|-\alpha_{1t}v_{t+1} + \alpha_{2t}d_t\|^2\\ &= f(x_t) - \alpha_{1t}\langle\nabla f(x_t), v_{t+1}\rangle + \alpha_{2t}\langle\nabla f(x_t), d_t\rangle + \frac{L\alpha_{1t}^2}{2}\|v_{t+1}\|^2 + \frac{L\alpha_{2t}^2}{2}\|d_t\|^2 - L\alpha_{1t}\alpha_{2t}\langle v_{t+1}, d_t\rangle\\ &= f(x_t) + \frac{\alpha_{1t}}{2}\|\nabla f(x_t) - v_{t+1}\|^2 - \frac{\alpha_{1t}(1-L\alpha_{1t})}{2}\|v_{t+1}\|^2 - \frac{\alpha_{1t}}{2}\|\nabla f(x_t)\|^2 + \alpha_{2t}\langle\nabla f(x_t), d_t\rangle\\ &\quad + \frac{L\alpha_{2t}^2}{2}\|d_t\|^2 - L\alpha_{1t}\alpha_{2t}\langle v_{t+1}, d_t\rangle.\end{aligned} \tag{5}$$

By Lemma 3, we can obtain

$$\sum_{t=1}^{T}\mathbb{E}[\|\nabla f(x_{t-1})-v_t\|^2]\leq\frac{1}{1-\beta}\mathbb{E}[\|\nabla f(x_0)-v_1\|^2]+2(1-\beta)T\sigma^2+\frac{L^2}{(1-\beta)^2}\mathbb{E}\left[\sum_{t=1}^{T}\|d_t\|^2\right]. \tag{6}$$

Taking expectation for both sides of (5) and taking summation among $t=0,...,T-1$, combining with (6), we have

$$\begin{aligned}&\mathbb{E}\left[f(x_T)-f(x_0)\right]\\ \leq&\frac{\eta G}{2}\left[\frac{\mathbb{E}[\|\nabla f(x_0)-v_1\|^2]}{1-\beta}+2(1-\beta)T\sigma^2+\frac{L^2}{(1-\beta)^2}\sum_{t=1}^{T}\mathbb{E}[\|d_t\|^2]\right]-\sum_{t=0}^{T-1}\frac{\alpha_{1t}}{2}\mathbb{E}[\|\nabla f(x_t)\|^2]\\ &-\sum_{t=0}^{T-1}\frac{\alpha_{1t}(1-L\alpha_{1t})}{2}\mathbb{E}[\|v_{t+1}\|^2]+\sum_{t=0}^{T-1}\left(\mathbb{E}[\alpha_{2t}\langle\nabla f(x_t),d_t\rangle]+\frac{L\alpha_{2t}^2}{2}\|d_t\|^2-\mathbb{E}[L\alpha_{1t}\alpha_{2t}\langle v_{t+1},d_t\rangle]\right).\end{aligned}$$

By AM-GM inequality,

$$\begin{aligned}\alpha_{2t}\langle\nabla f(x_t),d_t\rangle&\leq\frac{\alpha_{1t}}{4}\|\nabla f(x_t)\|^2+\frac{\alpha_{2t}^2}{\alpha_{1t}}\|d_t\|^2,\\ -L\alpha_{1t}\alpha_{2t}\langle v_{t+1},d_t\rangle&\leq\frac{\alpha_{1t}(1-L\alpha_{1t})}{4}\|v_{t+1}\|^2+\frac{L^2\alpha_{1t}\alpha_{2t}^2}{1-L\alpha_{1t}}\|d_t\|^2.\end{aligned} \tag{7}$$

Combining all together, we have

$$\begin{aligned}&\sum_{t=0}^{T-1}\frac{\alpha_{1t}}{4}\mathbb{E}[\|\nabla f(x_t)\|^2]\\ \leq&f(x_0)-f(x^*)+\frac{\eta G}{2(1-\beta)}\mathbb{E}[\|\nabla f(x_0)-v_1\|^2]+\sum_{t=1}^{T}\frac{\eta GL^2}{2(1-\beta)^2}\mathbb{E}[\|d_t\|^2]\\ &+\eta G(1-\beta)T\sigma^2+\sum_{t=0}^{T-1}\left(\frac{\alpha_{2t}^2}{\alpha_{1t}}+\frac{L\alpha_{2t}^2}{2}+\frac{L^2\alpha_{1t}\alpha_{2t}^2}{1-L\alpha_{1t}}\right)\mathbb{E}[\|d_t\|^2]-\sum_{t=0}^{T-1}\frac{\alpha_{1t}(1-L\alpha_{1t})}{4}\mathbb{E}[\|v_{t+1}\|^2],\end{aligned} \tag{8}$$

where x^* is one of the global minimizer of F. Since $\alpha_{1t}\leq\eta G\leq\frac{1}{2L}$, we have $\frac{\alpha_{1t}(1-L\alpha_{1t})}{4}\geq\frac{\alpha_{1t}}{8}$. By the conditions for η and α, we have $\frac{\alpha_{1t}}{16}\geq\frac{\alpha}{16}\geq\frac{\eta^3GL^2}{2(1-\beta)^2}\geq\frac{\eta GL^2\eta_{t+1}^2}{2(1-\beta)^2}$, $\frac{\alpha_{1t}}{96}\geq\frac{\alpha}{96}\geq\frac{\eta^4G^2}{\alpha}\geq\frac{\alpha_{2,t+1}^2\eta_{t+1}^2}{\alpha_{1,t+1}}$, $\frac{\alpha_{1t}}{96}\geq\frac{\alpha}{96}\geq\frac{L\eta^4G^2}{2}\geq\frac{L\alpha_{2,t+1}^2\eta_{t+1}^2}{2}$, and $\frac{\alpha_{1t}}{96}\geq\frac{\alpha}{96}\geq 2L^2\eta^5G^3\geq\frac{L^2\alpha_{1,t+1}\alpha_{2,t+1}^2\eta_{t+1}^2}{1-L\alpha_{1,t+1}}$. By $\|d_t\|\leq\eta_t\|v_t\|$. Since $v_0=0$, we have

$$\begin{aligned}&\sum_{t=0}^{T-1}\left(\frac{\alpha_{2t}^2}{\alpha_{1t}}+\frac{L\alpha_{2t}^2}{2}+\frac{L^2\alpha_{1t}\alpha_{2t}^2}{1-L\alpha_{1t}}\right)\|d_t\|^2+\sum_{t=1}^{T}\frac{\eta GL^2}{2(1-\beta)^2}\|d_t\|^2-\sum_{t=0}^{T-1}\frac{\alpha_{1t}(1-L\alpha_{1t})}{4}\|v_{t+1}\|^2\\ \leq&-\frac{\alpha}{8}\sum_{t=0}^{T-1}\|v_{t+1}\|^2+\sum_{t=0}^{T-1}\left(\frac{\alpha_{2t}^2\eta_t^2}{\alpha_{1t}}+\frac{L\alpha_{2t}^2\eta_t^2}{2}+\frac{L^2\alpha_{1t}\alpha_{2t}^2\eta_t^2}{1-L\alpha_{1t}}\right)\|v_t\|^2+\sum_{t=0}^{T-1}\frac{\eta GL^2\eta_{t+1}^2}{2(1-\beta)^2}\|v_{t+1}\|^2\\ \leq&-\frac{\alpha}{32}\sum_{t=0}^{T-1}\|v_{t+1}\|^2.\end{aligned} \tag{9}$$

Combining (8) and (9), we can obtain

$$\sum_{t=0}^{T-1}\frac{\alpha_{1t}}{4}\mathbb{E}[\|\nabla f(x_t)\|^2] \leq f(x_0) - f(x^*) + \frac{\eta G}{2(1-\beta)}\mathbb{E}[\|\nabla f(x_0) - v_1\|^2] + \eta G(1-\beta)T\sigma^2,$$
$$\frac{\alpha}{32}\sum_{t=0}^{T-1}\mathbb{E}[\|v_{t+1}\|^2] \leq f(x_0) - f(x^*) + \frac{\eta G}{2(1-\beta)}\mathbb{E}[\|\nabla f(x_0) - v_1\|^2] + \eta G(1-\beta)T\sigma^2.$$

Dividing the above two inequalities by $\frac{\alpha T}{4}$ and $\frac{\alpha T}{32}$ respectively, we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla f(x_t)\|^2] \leq \frac{4\left(f(x_0) - f(x^*)\right)}{T\alpha} + \frac{2G\mathbb{E}[\|\nabla f(x_0) - v_1\|^2]}{\delta(1-\beta)T} + \frac{4G(1-\beta)\sigma^2}{\delta},$$
$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\|v_{t+1}\|^2] \leq \frac{32\left(f(x_0) - f(x^*)\right)}{T\alpha} + \frac{16G\mathbb{E}[\|\nabla f(x_0) - v_1\|^2]}{\delta(1-\beta)T} + \frac{32G(1-\beta)\sigma^2}{\delta},$$

which completes the proof by letting $C_1 = 4\left(f(x_0) - f(x^*)\right)$, $C_2 = \frac{4G}{\delta}$, $C_3 = \frac{2G\mathbb{E}[\|\nabla f(x_0)-v_1\|^2]}{\delta}$.
■

Proof of Theorem 1

Proof By the selections of α and η_t in Theorem 1, let $\delta = \alpha/\eta$. By Lemma 4, we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\nabla f(x_t)\|^2\right] \leq \frac{C_1}{T\alpha} + C_2(1-\beta)\sigma^2 + \frac{C_3}{T(1-\beta)}.$$

The conditions $1-\beta \leq \frac{\epsilon^2}{3C_2\sigma^2}$ and $T \geq \max\left\{\frac{3C_1}{\alpha\epsilon^2}, \frac{3C_3}{(1-\beta)\epsilon^2}\right\}$ lead to $\frac{C_1}{T\alpha} \leq \frac{\epsilon^2}{3}$, $C_2(1-\beta)\sigma^2 \leq \frac{\epsilon^2}{3}$, $\frac{C_3}{T(1-\beta)} \leq \frac{\epsilon^2}{3}$. This completes the proof. ■

Proof of Theorem 2

Proof From (5) in Lemma 4, we have

$$\begin{aligned} f(x_{t+1}) \leq f(x_t) &+ \frac{\alpha_{1t}}{2}\|\nabla f(x_t) - v_{t+1}\|^2 - \frac{\alpha_{1t}(1-L\alpha_{1t})}{2}\|v_{t+1}\|^2 - \frac{\alpha_{1t}}{2}\|\nabla f(x_t)\|^2 + \alpha_{2t}\langle \nabla f(x_t), d_t\rangle \\ &+ \frac{L\alpha_{2t}^2}{2}\|d_t\|^2 - L\alpha_{1t}\alpha_{2t}\langle v_{t+1}, d_t\rangle. \end{aligned} \tag{10}$$

By Lemma 3, we have

$$(1-\beta_t)\|v_t - \nabla f(x_{t-1})\|^2 \leq \|v_t - \nabla f(x_{t-1})\|^2 - \mathbb{E}_t[\|v_{t+1} - \nabla f(x_t)\|^2] + 2(1-\beta_t)^2\mathbb{E}_t[\|\nabla f(x_t) - g_t\|^2] + \frac{L^2\|d_t\|^2}{1-\beta_t}.$$

Taking expectation and summation for $t = 1, ..., T$, we get

$$\sum_{t=0}^{T-1}\mathbb{E}[(1-\beta_{t+1})\|\nabla f(x_t) - v_{t+1}\|^2] \leq \mathbb{E}[\|v_1 - \nabla f(x_0)\|^2] + \sum_{t=1}^{T}\left(2(1-\beta_t)^2\sigma^2 + \frac{L^2\mathbb{E}[\|d_t\|^2]}{1-\beta_t}\right). \tag{11}$$

Note that $1-\beta_{t+1}=C\eta_t$, so $\frac{\alpha_{1t}}{2}\le\frac{G}{2}\eta_t=\frac{G}{2C}(1-\beta_{t+1})$. Taking expectation for both sides of (10) and taking summation among $t=0,...,T-1$, combining with (11), we can obtain

$$\begin{aligned}
&\mathbb{E}\left[f(x_T)-f(x_0)\right]\\
\le&\frac{G}{2C}\left[\mathbb{E}[\|v_1-\nabla f(x_0)\|^2]+\sum_{t=1}^{T}\left(2(1-\beta_t)\sigma^2+\frac{L^2\|d_t\|^2}{1-\beta_t}\right)\right]-\sum_{t=0}^{T-1}\frac{\alpha_{1t}(1-L\alpha_{1t})}{2}\mathbb{E}[\|v_{t+1}\|^2]\\
&-\sum_{t=0}^{T-1}\frac{\alpha_{1t}}{2}\mathbb{E}[\|\nabla f(x_t)\|^2]+\sum_{t=0}^{T-1}\left(\mathbb{E}[\alpha_{2t}\langle\nabla f(x_t),d_t\rangle]+\frac{L\alpha_{2t}^2}{2}\|d_t\|^2-\mathbb{E}[L\alpha_{1t}\alpha_{2t}\langle v_{t+1},d_t\rangle]\right).
\end{aligned}\tag{12}$$

From (7) and (12), we can get

$$\begin{aligned}
&\sum_{t=0}^{T-1}\frac{\alpha_{1t}}{4}\mathbb{E}[\|\nabla f(x_t)\|^2]\\
\le&f(x_0)-f(x^*)+\frac{G}{2C}\mathbb{E}[\|\nabla f(x_0)-v_1\|^2]+\sum_{t=1}^{T}\frac{G}{C}(1-\beta_t)^2\sigma^2+\sum_{t=1}^{T}\frac{GL^2\|d_t\|^2}{2C(1-\beta_t)}\\
&+\sum_{t=0}^{T-1}\left(\frac{\alpha_{2t}^2}{\alpha_{1t}}+\frac{L\alpha_{2t}^2}{2}+\frac{L^2\alpha_{1t}\alpha_{2t}^2}{1-L\alpha_{1t}}\right)\mathbb{E}[\|d_t\|^2]-\sum_{t=0}^{T-1}\frac{\alpha_{1t}(1-L\alpha_{1t})}{4}\mathbb{E}[\|v_{t+1}\|^2],
\end{aligned}\tag{13}$$

By the conditions for c_η and C, we have $\alpha_{1t}\le\eta_tG\le\frac{1}{2L}$, $\frac{\alpha_{1t}(1-L\alpha_{1t})}{4}\ge\frac{\alpha_{1t}}{8}$. By similar arguments in the proof of Lemma 4, we have $\frac{\alpha_{1t}}{16}\ge\frac{\delta\eta_t}{16}\ge\frac{\eta_{t+1}^2GL^2}{2\eta_tC^2}=\frac{GL^2\eta_{t+1}^2}{2(1-\beta_{t+1})C}$, $\frac{\alpha_{1t}}{96}\ge\frac{\delta\eta_t}{96}\ge\frac{\alpha_{2,t+1}^2\eta_{t+1}^2}{\alpha_{1,t+1}}$, $\frac{\alpha_{1t}}{96}\ge\frac{\delta\eta_t}{96}\ge\frac{L\alpha_{2,t+1}^2\eta_{t+1}^2}{2}$, and $\frac{\alpha_{1t}}{96}\ge\frac{\delta\eta_t}{96}\ge 2L^2\eta_{t+1}^5G^3\ge\frac{L^2\alpha_{1,t+1}\alpha_{2,t+1}^2\eta_{t+1}^2}{1-L\alpha_{1,t+1}}$. By $\|d_t\|\le\eta_t\|v_t\|$. Since $v_0=0$, we can get

$$\begin{aligned}
&\sum_{t=0}^{T-1}\left(\frac{\alpha_{2t}^2}{\alpha_{1t}}+\frac{L\alpha_{2t}^2}{2}+\frac{L^2\alpha_{1t}\alpha_{2t}^2}{1-L\alpha_{1t}}\right)\|d_t\|^2+\sum_{t=1}^{T}\frac{GL^2}{2C(1-\beta_t)}\|d_t\|^2-\sum_{t=0}^{T-1}\frac{\alpha_{1t}(1-L\alpha_{1t})}{4}\|v_{t+1}\|^2\\
\le&-\sum_{t=0}^{T-1}\frac{\alpha_{1t}}{8}\|v_{t+1}\|^2+\sum_{t=0}^{T-1}\left(\frac{\alpha_{2t}^2\eta_t^2}{\alpha_{1t}}+\frac{L\alpha_{2t}^2\eta_t^2}{2}+\frac{L^2\alpha_{1t}\alpha_{2t}^2\eta_t^2}{1-L\alpha_{1t}}\right)\|v_t\|^2+\sum_{t=0}^{T-1}\frac{GL^2\eta_{t+1}^2}{2C(1-\beta_{t+1})}\|v_{t+1}\|^2\\
\le&-\sum_{t=0}^{T-1}\frac{\alpha_{1t}}{32}\|v_{t+1}\|^2.
\end{aligned}\tag{14}$$

Combining (13) and (14), we can obtain

$$\begin{aligned}
\sum_{t=0}^{T-1}\frac{\delta\eta_t}{4}\mathbb{E}[\|\nabla f(x_t)\|^2]\le\sum_{t=0}^{T-1}\frac{\alpha_{1t}}{4}\mathbb{E}[\|\nabla f(x_t)\|^2]\le f(x_0)-f(x^*)+\frac{G\mathbb{E}[\|\nabla f(x_0)-v_1\|^2]}{2C}+\sum_{t=1}^{T}\frac{G\sigma^2}{C}(1-\beta_t)^2,\\
\sum_{t=0}^{T-1}\frac{\delta\eta_t}{32}\mathbb{E}[\|v_{t+1}\|^2]\le\sum_{t=0}^{T-1}\frac{\alpha_{1t}}{32}\mathbb{E}[\|v_{t+1}\|^2]\le f(x_0)-f(x^*)+\frac{G\mathbb{E}[\|\nabla f(x_0)-v_1\|^2]}{2C}+\sum_{t=1}^{T}\frac{G\sigma^2}{C}(1-\beta_t)^2.
\end{aligned}$$

Then, the final assertion can be obtained by $\sum_{t=1}^{T}\frac{1}{t+1}=\mathcal{O}(\log T)$. This completes the proof. ■

Appendix D. Ablation Study

Different search directions We compare the performance of algorithms that solve the trust-region subproblem in one-dimensional, two-dimensional (DRAG), and three-dimensional subspaces as described in Section 2. As show in Table 3, DRAG generalizes better than its one search direction and three search direction counterparts. The reason is that DRAG updates in more directions than the one search direction counterpart while its subproblem can be solved more accurately than the three direction counterpart, since low-dimensional subproblem can be solved with less numerical errors in single precision arithmetic by GPU.

Table 3: Test accuracy of algorithms solving the trust-region subproblem with one, two, and three search directions on CIFAR10.

	VGG16	ResNet34	DenseNet121
1 direction	93.8	95.3	96.0
DRAG	**94.0**	**95.6**	**96.1**
3 directions	93.8	95.4	95.7

Robustness to learning rate schedule DRAG is robust to different choices of learning rate schedule. Except for letting the learning rate decay at epoch 150 as in Section 3.1, we also conduct experiments on decaying the learning rate at epoch 120 and adopting cosine annealing learning rate schedule. The only change of hyperparameter setting from Section 3.1 is we increase the learning rate of DRAG from 0.1 to 0.12 in cosine annealing schedule. The intuition is that when the trust-region radius is decreased during the training process, we need a larger initial radius to converge to a better local minima. We compared DRAG's test performance with other optimizers with VGG16 on CIFAR10, details are presented in Table 4, which shows that DRAG enjoys the best generalization performance for all the learning rate schedules.

Table 4: Test accuracy of VGG16 on CIFAR-10 with three different learning rate schedules.

	DRAG	SGD	Adam	AdamW	Adabelief
Cosine Annealing	**94.3**	94.0	92.2	92.4	94.1
Decay at 120 epoch	**93.8**	92.5	91.8	92.6	93.6
Decay at 150 epoch	**94.0**	92.7	92.2	92.4	93.6