Supplementary Material to "NKFAC: A Fast and Stable KFAC Optimizer for Deep Neural Networks"

Ying Sun, Hongwei Yong, and Lei Zhang

The Hong Kong Polytechnic University, Hong Hum, Kowloon, Hong Kong, China.
`csysun@comp.polyu.edu.hk`, `hongwei.yong@polyu.edu.hk`, `cslzhang@comp.polyu.edu.hk`

Abstract. In this supplementary file, we first report the related hyper-parameters of different optimizers on CIFAR100/10 and ImageNet in Sections 4.2 and 4.3. Then we show the ablation studies results about the stepsize of Newton's iteration and the implementations, with experimental verification of the slow change of Fisher information statistics to better study the properties of our proposed NKFAC.

1 Hyper-parameters Settings

In this section, we report the initial learning rate and weight decay of different optimizers on CIFAR100/10 in Table 1 and on ImageNet in Table 2. These hyper-parameters are adopted in the experiments in Sections 4.2 and 4.3.

Table 1. Settings of learning rate (LR) and weight decay (WD) for different optimizers on CIFAR100/10.

Optimizer	SGDM	AdamW	RAdam	Adabelief	KFAC	SKFAC	NKFAC
LR	0.1	0.001	0.001	0.001	0.0005	0.0005	0.05
WD	0.0005	0.5	0.5	0.5	0.1	0.1	0.001

Table 2. Settings of learning rate (LR) and weight decay (WD) for different optimizers on ImageNet.

Optimizer	SGDM	AdamW	RAdam	Adabelief	KFAC	SKFAC	NKFAC
LR	0.1	0.001	0.001	0.001	0.0005	0.0005	0.05
WD	0.0001	0.1	0.1	0.5	0.02	0.02	0.0002

2 Ablation Study on the stepsize of Newton's iteration

Recall that for a given real matrix $\mathbf{A}$, our **Algorithm 1** (i.e., the Newton's method for solving matrix inverse) obtains a sequence $\{\mathbf{B}_k\}$ that performed simply by

$$\mathbf{B}_{k+1} = (1+\alpha_{k+1})\mathbf{B}_k - \alpha_{k+1}\mathbf{B}_k\mathbf{A}\mathbf{B}_k, \tag{1}$$

with $\alpha_{k+1} = 1$ when $\|\mathbf{I} - \mathbf{A}\mathbf{B}_0\| < 1$ and $\alpha_{k+1} = \frac{1}{\|\mathbf{A}\mathbf{B}_k\|}$ when $\|\mathbf{I} - \mathbf{A}\mathbf{B}_0\| > 1$. When choosing α_{k+1} in the latter case, the most natural idea is to use the length that drops the deviation most along this descent direction as the stepsize, specifically, as in the below Proposition 1.

Proposition 1. *Denote the descent direction* $\mathbf{D}_k := \mathbf{B}_k(\mathbf{A}\mathbf{B}_k - \mathbf{I})$ *and the function* $f_k(\alpha) := \frac{1}{2}\|\mathbf{A}(\mathbf{B}_k - \alpha\mathbf{D}_k) - \mathbf{I}\|^2$ *for each iteration* k*. Then the optimization problem* $\alpha^*_{k+1} := \arg\min_\alpha f_k(\alpha)$ *takes the optimal* $\alpha^*_{k+1} = \frac{\langle \mathbf{A}\mathbf{B}_k - \mathbf{I}, \mathbf{A}\mathbf{B}_k(\mathbf{A}\mathbf{B}_k - \mathbf{I})\rangle}{\|\mathbf{A}\mathbf{B}_k(\mathbf{A}\mathbf{B}_k - \mathbf{I})\|^2}$ *in iteration* k*.*

Table 3. Detection results of Faster-RCNN on COCO. Δ means the improvement of α_{k+1} compared with α^*_{k+1}.

Backbone, LR	Stepsize	AP	AP$_{.5}$	AP$_{.75}$	AP$_s$	AP$_m$	AP$_l$
ResNet50, 1×	α^*_{k+1}	38.7	59.5	42.0	22.9	42.3	50.3
	α_{k+1}	39.7	60.7	43.0	23.4	43.3	51.5
	Δ	↑1.0	↑1.2	↑1.0	↑0.5	↑1.0	↑1.2
ResNet101, 1×	α^*_{k+1}	39.1	59.5	42.6	22.6	42.9	51.2
	α_{k+1}	41.3	62.0	45.0	24.4	44.7	54.9
	Δ	↑2.2	↑ 2.5	↑2.4	↑1.8	↑1.8	↑3.7

Table 4. Detection and segmentation results of Mask-RCNN on COCO. Δ means the improvement of α_{k+1} compared with α^*_{k+1}.

Backbone, LR	Algorithm	AP^b	AP$^b_{.5}$	AP$^b_{.75}$	AP^m	AP$^m_{.5}$	AP$^m_{.75}$
ResNet50, 1×	α^*_{k+1}	39.5	60.4	43.1	36.3	57.5	38.9
	α_{k+1}	40.1	60.9	43.9	36.6	57.9	39.2
	Δ	↑0.6	↑ 0.5	↑0.8	↑0.3	↑0.4	↑0.3
ResNet101, 1×	α^*_{k+1}	41.4	61.9	45.4	37.4	59.0	40.2
	α_{k+1}	41.8	62.1	45.9	37.8	59.2	40.3
	Δ	↑0.4	↑ 0.2	↑0.5	↑0.4	↑0.2	↑0.1

Fig. 1. Testing accuracies (%) and time cost by computing dampening and inversion (s) of different optimizers **with implementations** on CIFAR100.

	Accuracy			Time Cost		
Optimizer	KFAC*	SKFAC*	NKFAC	KFAC*	SKFAC*	NKFAC
DenseNet121	81.04 ± .22	80.66 ± .13	**81.13** ± .16	362.14	331.04	**132.00**
ResNet50	**81.89** ± .21	81.08 ± .18	81.78 ± .06	306.88	191.33	**134.21**

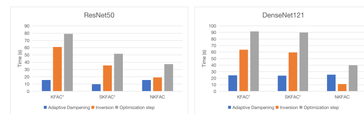

Fig. 2. Time cost of optimization steps for KFAC*, SKFAC* and NKFAC on CIFAR100.

However, our experiments show that this α^*_{k+1} may not works as good as $\alpha_{k+1} = \frac{1}{\|\mathbf{AB}_k\|}$, which is chosen from our experience. Here we show some of the comparison results on detection and segmentation tasks in Table 3 and Table 4. In summary, the stepsize chosen in Newton's iteration indeed leave impact on the performance of NKFAC. Luckily, our stepsize $\alpha_{k+1} = \frac{1}{\|\mathbf{AB}_k\|}$ that chosen in **Algorithm 1** performs good throughout our experiments.

3 Ablation Study on Implementations

With our experiment results in Section 4, we are motivated to add our implementations to KFAC and SKFAC, denoted by KFAC* and SKFAC*, respectively. In this section, we compare the implemented KFAC*, SKFAC* with NKFAC on CIFAR100[1] and ImageNet[2] datasets.

Here, the learning rate and weight decay of KFAC*, SKFAC* and NKFAC are set to be the same, specifically, 0.05 and 0.001, respectively. We see NKFAC still shows its advantage of time cost from Table 1 and achieves higher generalization performances than SKFAC*, while stably reducing the inversion cost by 56% and 64%. KFAC* shows its efficiency benefited from our useful implementations and achieves the highest accuracy in ResNet50 while NKFAC is still the best in DenseNet121. Thus, as a balance, NKFAC can be a good choice in applications.

To clearly compare the time cost of inversion and adaptive dampening in each step, we plot Figure 2 to show the time cost proportion of each part in a single optimization step using the experiment results on CIFAR100. To eliminate randomness, the time reported in Figure 2 is the cumulative time of the first 50 epochs in the training process. SKFAC can also shorten the time in computing the adaptive dampening after dimension reduction, as shown in Figure 2, while NKFAC shows the least time in the optimization step for both the DNNs tested.

4 Supplementary Results about the Change of Statistics

Although we have clearly explained in the paper why the Fisher information statistics $\mathbf{L}_t$ and $\mathbf{R}_t$ do not vary much, to have a more intuitive explanation, we draw the changing curves of these two statistics during the training process for ResNet18 on CIFAR100. Here in the Fig. 3, the plot value represents the norm of the difference of the statistics between an interval of T_{stat} (takes 20 here) and is divided by the current statistics norm. We take the average of the results of four experiments to plot the figure. We can see from Fig. 3 that, except for the very beginning stage, all the values are small, which verifies the slow change of our Fisher information statistics.

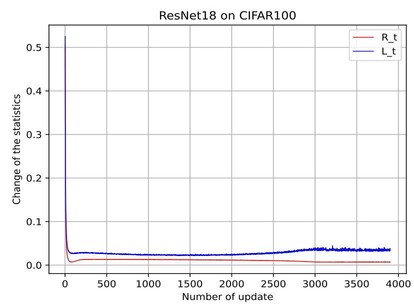

Fig. 3. Change of $\mathbf{L}_t$ and $\mathbf{R}_t$ in training ResNet18 on CIFAR100.

References

1. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
2. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision **115**(3), 211–252 (2015)