Breaking the Low-Rank Dilemma of Linear Attention

Supplementary Material

A. More Analysis About α

In this section, we further analyze the rank-boosting effect of α on the KV buffer and conduct additional ablation experiments on α to validate the rationale behind our design.

An analysis of the rank-boosting effect. Assume in $\kappa(K_j^T)V_j, j \in [0, N]$ that the two components $\kappa(K_1^T)V_1$ and $\kappa(K_2^T)V_2$ are linearly correlated. That means:

$$\kappa(K_1^T)V_1 = \lambda\kappa(K_2^T)V_2. \tag{1}$$

In general linear attention mechanisms, a portion of the KV buffer formed by the sum of the two can be expressed as:

$$\kappa(K_1^T)V_1 + \kappa(K_2^T)V_2 = (1+\lambda)\kappa(K_2^T)V_2 \tag{2}$$

After introducing α as a modulation coefficient for each term, their sum becomes:

$$\begin{aligned}\alpha_1\kappa(K_1^T)V_1 + \alpha_2\kappa(K_2^T)V_2 &= \lambda\alpha_1\kappa(K_2^T)V_2 + \alpha_2\kappa(K_2^T)V_2 \\ &= \alpha_2(1+\frac{\lambda\alpha_1}{\alpha_2})\kappa(K_2^T)V_2\end{aligned} \tag{3}$$

Since in our setup the value of α_1 and α_2 are calculated based on the attention scores between global query and keys, it changes with the input samples and the training process. This results in more varied coefficients, meaning that the new matrix composed of these two linearly correlated matrices has a broader range of possible values and greater flexibility. This makes the KV buffer $\sum_{j=1}^{N}\alpha_j\kappa(K_j^T)V_j$ more likely to achieve a full-rank state.

As previously mentioned (Eq. 2 and Eq. 3), two linearly correlated matrices can be summed to form a single matrix with coefficients. Therefore, we consider here only the case where all matrices are linearly independent. Consider the equation:

$$\sum_{j=1}^{N} c_j\kappa(K_j^T)V_j = C \tag{4}$$

Here, $C \in \mathbb{R}^{d\times d}$ is a full-rank matrix. Since all $\kappa(K_j^T)V_j$ are linearly independent, this equation has a **unique solution or no solution**. After decomposing c_j into $d_j\alpha_j$, the original equation becomes:

$$\sum_{j=1}^{N} d_j\alpha_j\kappa(K_j^T)V_j = C \tag{5}$$

Due to the presence of different scalars, the solutions can be multiple, giving the matrix representation a broader scope. This implies that there are more solutions that enable the KV buffer to achieve a full-rank state.

More choices about α. We choose the attention scores between the keys and the global query as the value for α, and here we experiment with different selections. When α is set as a learnable vector, its fixed shape of $1 \times N$ makes it challenging to apply to higher resolution tasks such as object detection. Therefore, we only evaluate its performance on image classification tasks. We conduct experiments using the DeiT-T [9] configuration, and the results are shown in Tab. 1. While the learnable α still significantly enhances the model's performance, its effectiveness is not as impressive as the α based on the attention score.

Model	Params(M)	FLOPs(G)	Acc(%)
DeiT-T	6	1.1	72.2
attn score	6	1.1	75.1
learnable	6	1.1	73.8(-1.3)

Table 1. More choice about α.

B. More Analysis About $\phi(.)$

An analysis of the rank-boosting effect. Consider two matrices A and B, $A, B \in \mathbb{R}^{m\times n}$, and represent these matrices as a linear combination of rank-one matrices. That is:

$$A = \sum_{i=1}^{r} u_i v_i^T, \quad B = \sum_{j=1}^{s} x_j y_j^T \tag{6}$$

where $r = \text{Rank}(A)$, $s = \text{Rank}(B)$, $u_i, x_j \in \mathbb{R}^{m\times 1}$, $v_i, y_j \in \mathbb{R}^{n\times 1}$. Consider the Hardmard product:

$$(A \odot B)_{ij} = A_{ij} \odot B_{ij} \tag{7}$$

we can rewrite it as:

$$\begin{aligned}A \odot B &= (\sum_{i=1}^{r} u_i v_i^T) \odot (\sum_{j=1}^{s} x_j y_j^T) \\ &= \sum_{i=1}^{r}\sum_{j=1}^{s}(u_i \odot x_j)(v_i \odot y_j)^T\end{aligned} \tag{8}$$

This expression shows that $A \odot B$ can be viewed as a linear combination of rs rank-one matrices. Therefore, the total rank of $A \odot B$ does not exceed the number of matrices, which is:

$$\text{Rank}(A \odot B) \leq rs = \text{Rank}(A) \times \text{Rank}(B) \tag{9}$$

The above expression effectively demonstrates that when the ranks of the two matrices, r and s, are both small, the

Hadamard product can effectively raise the upper bound of the matrix rank. Therefore, when augmenting the rank of the output features matrix, the choice of $\phi(.)$ becomes less important; what matters is the use of the Hadamard product. This is consistent with the conclusions we obtained from the ablation experiments in the main text. Regardless of what $\phi(.)$ is, the presence of $\phi(.)$ will always enhance the rank of the matrix.

C. Experimental Details

Image Classification. We adopt the training strategy proposed in DeiT [9] with the only supervision is classification loss. Specifically, all models are trained from scratch for 300 epochs with the input resolution of 224×224. Adam is used with a cosine decay learning rate scheduler and 5 epochs of linear warm-up. The initial learning rate, weight decay, and batch-size are set to 0.001, 0.05, and 1024, respectively. We apply the same data augmentation and regularization as DeiT [9] (RandAugment [4] (randm9-mstd0.5-inc1) , Mixup [12] (prob = 0.8), CutMix [11] (prob = 1.0), Random Erasing (prob = 0.25)).

Object Detection and Instance Segmentation. We apply RetinaNet [7], Mask-RCNN [5], and Cascaded Mask R-CNN [1] as the frameworks based on the MMDetection [2] to evaluate our models. The models are trained under "1 $\times$" (12 training epochs) and "3 $\times$ +MS" (36 training epochs with multi-scale training) settings. For the "1 $\times$" setting, images are resized to the shorter side of 800 pixels while the longer side is within 1333 pixels. For the "3 $\times$ +MS", multi-scale training strategy is applied to randomly resize the shorter side between 480 to 800 pixels. We use the initial learning rate of 1e-4. For RetinaNet, we set the weight decay to 1e-4. For Mask-RCNN and Cascaded Mask R-CNN, we set it to 5e-2.

Semantic Segmentation. we implement UperNet [10] and SemanticFPN [6] based on MMSegmentation [3] to validate the models. For UperNet, we follow the previous setting [8] and train the model for 160k iterations with the input size of 512×512. For SemanticFPN, we also use the input resolution of 512×512 but train the models for 80k iterations.

References

[1] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, 2018. 2

[2] Kai Chen, Jiaqi Wang, Jiangmiao Pang, et al. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 2

[3] MMSegmentation Contributors. Mmsegmentation, an open source semantic segmentation toolbox, 2020. 2

[4] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, et al. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPRW*, 2020. 2

[5] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask r-cnn. In *ICCV*, 2017. 2

[6] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *CVPR*, 2019. 2

[7] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, and Kaiming He andPiotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 2

[8] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 2

[9] Hugo Touvron, Matthieu Cord, Matthijs Douze, et al. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021. 1, 2

[10] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *ECCV*, 2018. 2

[11] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, et al. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019. 2

[12] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, et al. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 2