Bootstrapping ViTs: Towards Liberating Vision Transformers from Pre-training
–*Supplementary Material*–

Haofei Zhang[1,*], Jiarui Duan[1,*], Mengqi Xue[1], Jie Song[1,†], Li Sun[1], Mingli Song[1,2]
[1]Zhejiang University [2] Shanghai Institute for Advanced Study of Zhejiang University

Appendix A
Convolution in Matrix Form

In this section, we formularize convolution in the matrix form consistent with MHSA. Let $\boldsymbol{F}^{\text{in}} \in \mathbb{R}^{H\times W\times d_{\text{in}}}$ denote a 2D input feature map to a $k_h \times k_w$ convolution layer. The receptive field N of the convolution layer is defined as $N = k_h \times k_w$. For the sake of simplicity, we assume that the input feature and the output feature share the same size, *i.e.*, $\boldsymbol{F}^{\text{out}} \in \mathbb{R}^{H\times W\times d_{\text{out}}}$, and the padding value of the convolution is set to zero. The output feature map at position (h, w) only depends on the neighborhood $\Delta_{h,w} = \{(h_i, w_j)\}_{(i,j)\in[k_h]\times[k_w]}$ of the input feature map:

$$\boldsymbol{F}^{\text{out}}_{h,w} = \sum_{(i,j)\in[k_h]\times[k_w]} \Delta_{h,w}[\boldsymbol{F}^{\text{in}}]_{i,j}\boldsymbol{W}_{i,j}, \tag{1}$$

where operation $\Delta_{h,w}[\cdot]$ denotes the selection of neighborhood of input 2D feature by the neighborhood indices, $\boldsymbol{W}_{i,j} \in \mathbb{R}^{d_{\text{in}}\times d_{\text{out}}}$ denotes the linear projection matrix, and $\boldsymbol{W} \in \mathbb{R}^{k_h\times k_w\times d_{\text{in}}\times d_{\text{out}}}$ is the parameter tensor of the convolution layer $\text{Conv}(\cdot; \boldsymbol{W})$.

Similarly, as we flatten the input feature map to 1D visual sequence $X = \text{Flatten}(\boldsymbol{F}^{\text{in}}) \in \mathbb{R}^{n\times d_{\text{in}}}$, $n = HW$, the q-th output visual token $y_q \in \mathbb{R}^{d_{\text{out}}}$ can be calculated by

$$y_q = \sum_{i=1}^{|\Delta_q|} \Delta_q[X]_i \boldsymbol{W}_i, \tag{2}$$

where $\Delta_q = \{p_1, \ldots, p_N\}$ is the sequence of 1D coordinates according to flattened indices $\Delta_{h,w}$, $\Delta_q[X]$ is the extracted subsequence of X, and $\boldsymbol{W}_i$ is the linear projection matrix corresponding to $\boldsymbol{W}_{i,j}$ in Eq. (1). Therefore, the output sequence $Y \in \mathbb{R}^{n\times d_{\text{out}}}$ is the stack of output tokens:

$$Y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \sum_{i=1}^{N} \begin{bmatrix} \Delta_1[X]_i \\ \vdots \\ \Delta_n[X]_i \end{bmatrix} \boldsymbol{W}_i. \tag{3}$$

Here, we abuse some notations for further derivation by setting $\Delta_i[X] = \begin{bmatrix}\Delta_1[X]_i^\top & \cdots & \Delta_n[X]_i^\top\end{bmatrix}^\top$ as the shift of input sequence and $W_i := \boldsymbol{W}_i$. It is worth noting that $\Delta_q[X]_i$ is the selection of one input visual token and can be treated as a linear transformation matrix $\phi_q^{(i)} \in \mathbb{R}^{1\times n}$. Specifically,

$$\phi^{(i)}_{q,p} = \begin{cases} 1, & \text{if } p = p_i = \Delta_q^{(i)}, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Hence, $\phi_q^{(i)}$ has at most one non-zero element ($\phi_q^{(i)} = \boldsymbol{0}$ when p_i is the zero padding index).

As such, the output sequence can be simplified as

$$Y = \sum_{i=1}^{N} \Phi_i X W_i, \tag{5}$$

where

$$\Phi_i = \begin{bmatrix} \phi_1^{(i)} \\ \vdots \\ \phi_n^{(i)} \end{bmatrix} \tag{6}$$

is the constant matrix with hard-coded inductive biases.

Appendix B
Intermediate Supervision Analysis

In this section, we analyze the effects when the intermediate supervisions act upon different layers. Fig. 1 shows the top-1 accuracy variation on CIFAR-100 dataset when the supervision at relative depth is removed from the final loss. The dash lines represent the performance without removing supervision for any layer. For the small settings, the importance increases significantly as going deeper. However, the base model relies on both shallow and deep layers.

Appendix C
Visualization of MHSA

*Equal contribution
†Corresponding author, email: sjie@zju.edu.cn

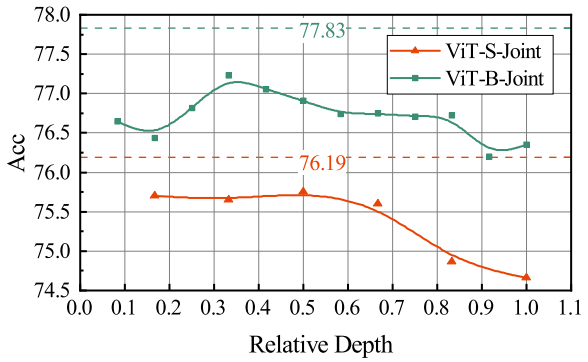
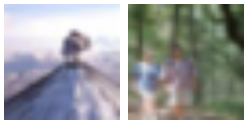


Figure 1. Accuracy variation when the intermediate supervision at different layers is removed.

The visualizations of learned self-attentions are shown in Tab. 3 and Tab. 4 for two randomly selected images from CIFAR-100. The heat maps of layer 2, 4, and 6 reveal that the inductive biases, such as sparsity and localized relationship, have been injected into ViTs, especially for shallow layers.



(a) (b)

Figure 2. Selected images from CIFAR-100 for visualization.

Appendix D
Visualization of Intermediate Feature

The visualizations of intermediate features for both agent CNN and the ViT are shown in Tab. 5 and Tab. 6 for two different input images (Fig. 2a and Fig. 2b). As we can observe, (1) when trained independently, the CNNs tend to produce smooth features while ViTs tend to generate sharp features due to the global attention mechanism; (2) when optimized jointly, as the inductive biases have been injected into the ViTs, they tend to pay more attention to the real object, which is similar to CNNs, and thus more robust to the disturbances.

Appendix E
Network Architecture of Agent CNNs

The network architectures of agent CNNs are given in Tab. 1 and Tab. 2, respectively for the base agent CNN and the res-like agent CNN. Here, 'H9' and 'H6' denotes generalized convolution with receptive field of 9 and 6 respectively. The input images are resized to 224×224 pixels for both agent CNNs and ViTs. In each block, the CONV layer replaces the MHSA layer in ViTs and the following MLP layer is composed of two 1×1 convolution layers. Finally, features from the global average pooling layer are fed into a fully connected (FC) layer for classification.

Table 1. Network architectures of base agent CNNs.

Layer	Agent-S	Agent-B
Input Projection	16×16 Conv, 288, S16	16×16 Conv, 384, S16
Blocks	[H9, CONV, 288 1×1 Conv, 1152 1×1 Conv, 288] ×6	[H6, CONV, 384 1×1 Conv, 1536 1×1 Conv, 384] ×12
Head	Global Average Pooling FC	

Table 2. Network architectures of res-like agent CNNs.

Layer	Agent-S	Agent-B
Input Projection	7×7 Conv, 64, S2 2×2, Max Pooling, S2 3×3 Conv, 288, S2	7×7 Conv, 64, S2 2×2, Max Pooling, S2 3×3 Conv, 384, S2
Blocks	[H9, CONV, 288 1×1 Conv, 1152 1×1 Conv, 288] ×1	[H6, CONV, 384 1×1 Conv, 1536 1×1 Conv, 384] ×2
	Down-sampling	Down-sampling
	[H9, CONV, 288 1×1 Conv, 1152 1×1 Conv, 288] ×1	[H6, CONV, 384 1×1 Conv, 1536 1×1 Conv, 384] ×2
	Down-sampling	Down-sampling
	[H9, CONV, 288 1×1 Conv, 1152 1×1 Conv, 288] ×4	[H6, CONV, 384 1×1 Conv, 1536 1×1 Conv, 384] ×8
Head	Global Average Pooling FC	

Table 3. Visualization of average attention for input Fig. 2a.

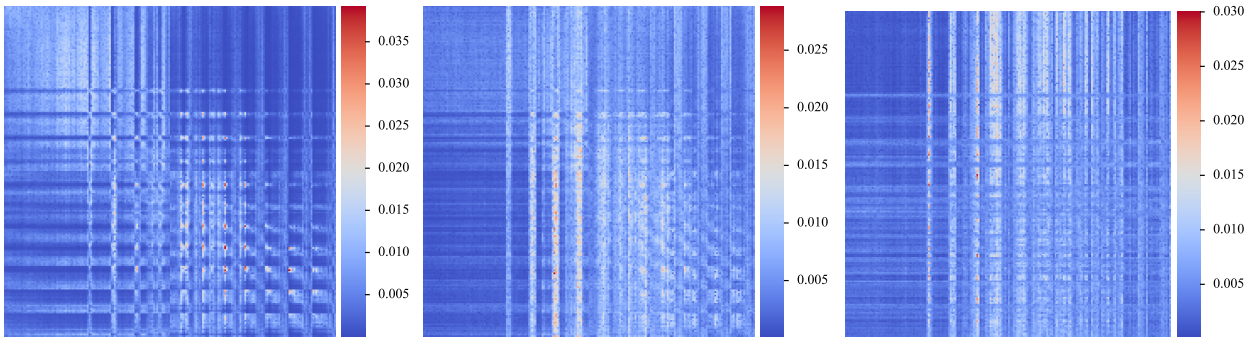
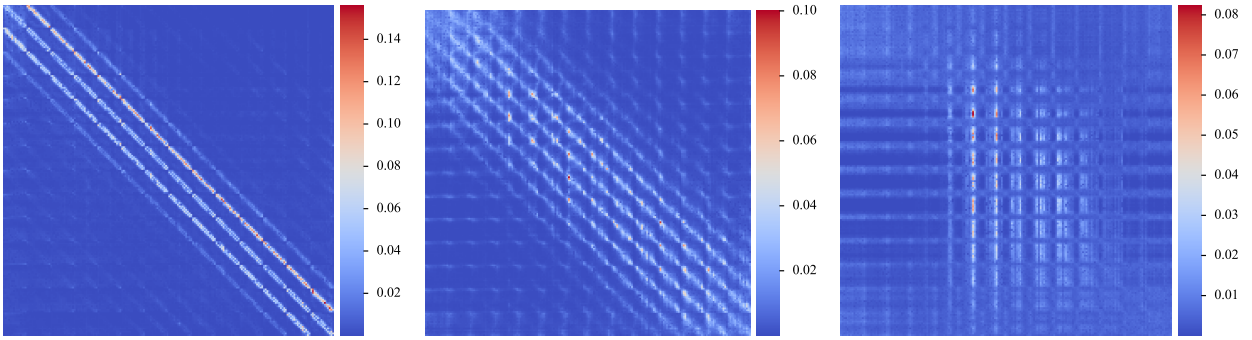
Layer	2	4	6
ViT-S			
Joint ViT-S			

Table 4. Visualization of average attention for input Fig. 2b.

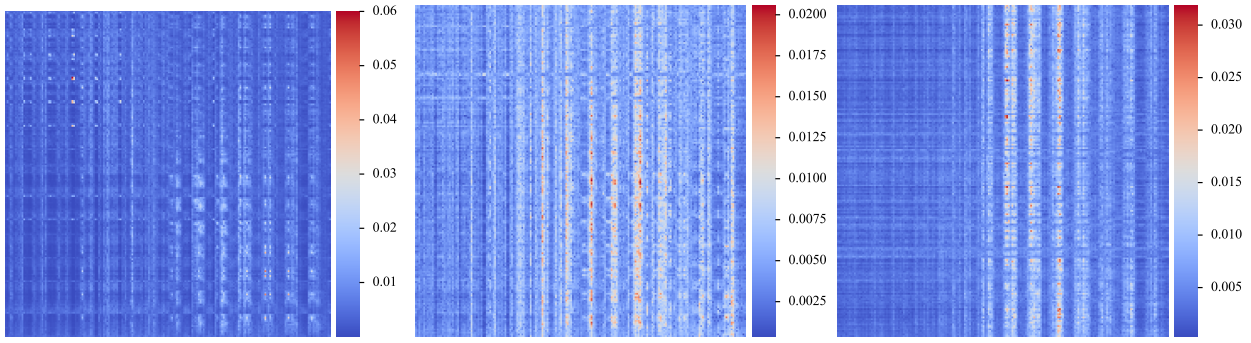
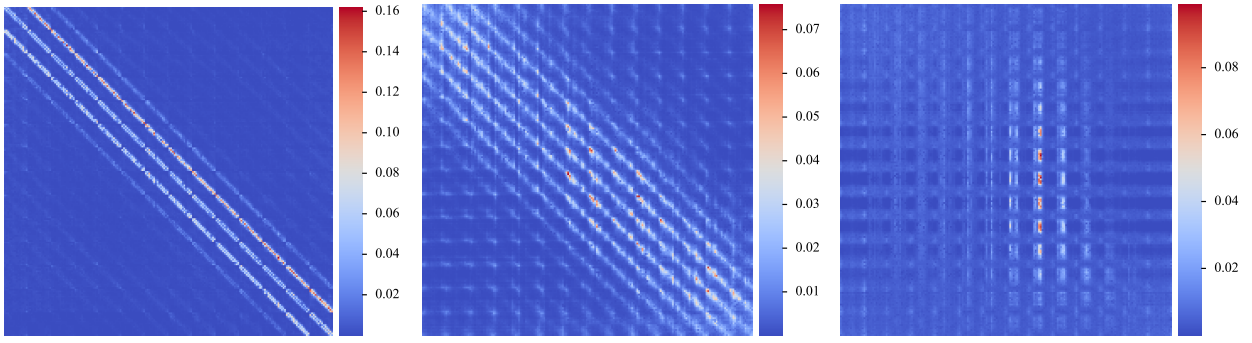
Layer	2	4	6
ViT-S			
Joint ViT-S			

Table 5. Visualization of intermediate features for input Fig. 2a. Please zoom for better view.

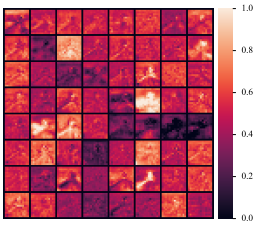
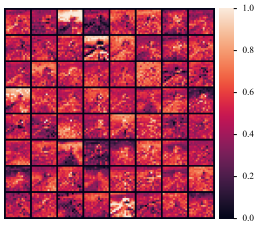
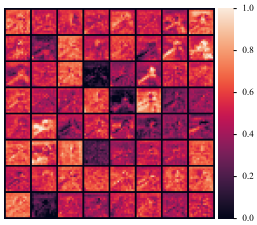
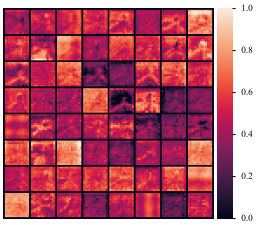
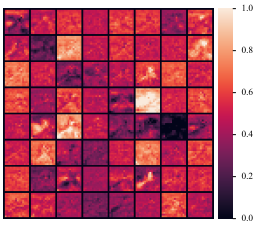
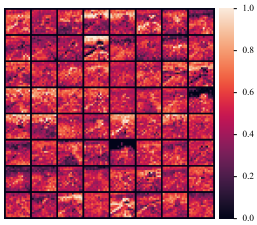
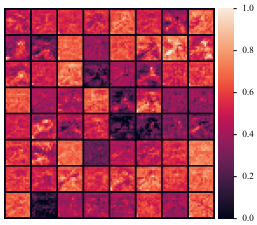
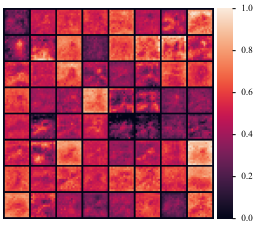
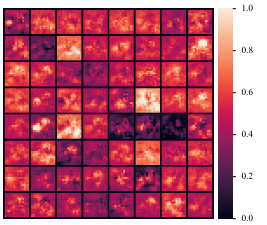
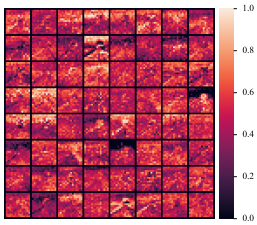
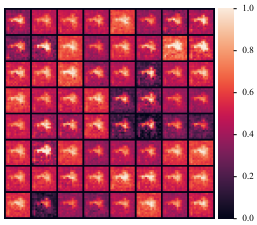
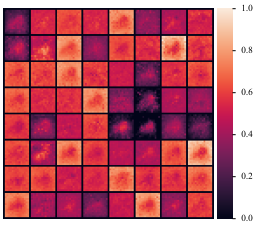
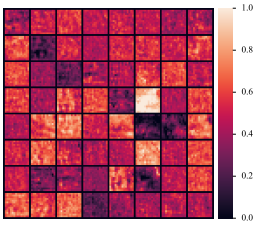
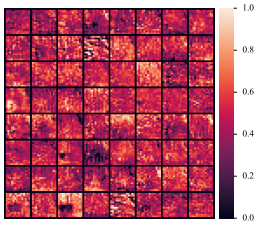
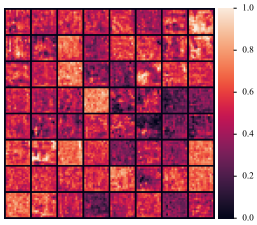
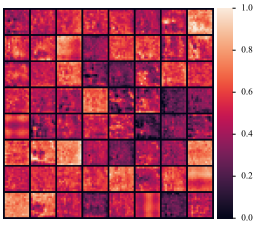
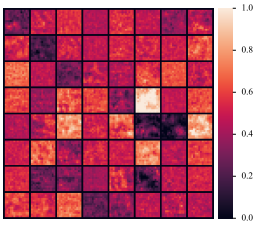
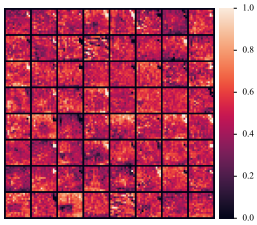
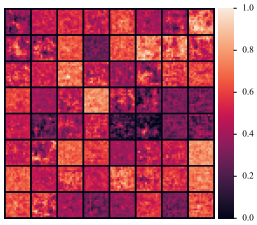
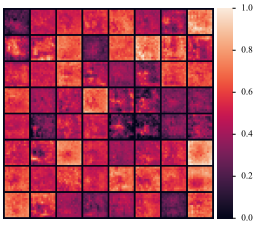
Layer	Agent-S	ViT-S	Agent-S-Joint	ViT-S-Joint
2				
4				
6				

Table 6. Visualization of intermediate features for input Fig. 2b. Please zoom for better view.

Layer	Agent-S	ViT-S	Agent-S-Joint	ViT-S-Joint
2				
4				
6