# Appendix for *Bootstrapping SparseFormers from Vision Foundation Models*

## A.1. More Ablations on Bootstrapping Settings

In Section 3, We have proposed to *truncate the leading, tune the middle, and freeze the ending pre-trained transformer blocks* to further reduce the compute and preserve the output embedding space of the foundation transformer to bootstrap from. Here, we investigate the effect of this bootstrapping paradigm in Table 1.

| model | #truncate blocks | #tunable blocks | IN-1K top-1 acc. | FLOPs | #Params |
|---|---|---|---|---|---|
| SF-B$_{\text{AugReg}}$, default | 4 | 4 of 8 | 82.5 | 3.8G | 86M |
| SF-B$_{\text{AugReg}}$, all frozen | 4 | 0 of 8 | 81.8 | 3.8G | 86M |
| SF-B$_{\text{AugReg}}$, all tunable | 4 | 8 of 8 | 82.4 | 3.8G | 86M |
| SF-B$_{\text{AugReg}}$, w/o truncation | 0 | 4 of 12 | 82.7 | 5.2G | 92M |
| SF-L$_{\text{AugReg}}$, default | 8 | 8 of 16 | 84.5 | 11.4G | 213M |
| SF-L$_{\text{AugReg}}$, all frozen | 8 | 0 of 16 | 84.0 | 11.4G | 213M |
| SF-L$_{\text{AugReg}}$, all tunable | 8 | 16 of 16 | 84.4 | 11.4G | 213M |
| SF-L$_{\text{AugReg}}$, w/o truncation | 0 | 8 of 24 | 84.3 | 16.4G | 314M |
| SF-L$_{\text{AugReg}}$, w/o truncation | 0 | 16 of 24 | 84.7 | 16.4G | 314M |

Table 1. Ablation on truncating, tuning, and freezing settings.

As shown in the table, bootstrapping SparseFormers without tuning pre-trained transformer blocks ("all frozen") leads to inferior results compared to ones that do tune. This is expected since frozen pre-trained transformer blocks can not adapt to the output of the focusing transformer during the bootstrapping procedure. However, going to the opposite extreme of making all pre-trained blocks tunable ("all tunable") can also be lagging. This may be because the frozen classifier relies on the structure of the well-preserved output embedding space in our bootstrapping setting. We believe that this is also true for vision language models. Besides that, we observe that bootstrapping without truncating leading blocks can be very unstable, and lead to different effects on SF-B$_{\text{AugReg}}$ and SF-L$_{\text{AugReg}}$ but with much more FLOPs and parameters. Therefore, we choose our truncating the leading, tuning the middle, and freezing the ending paradigm as our bootstrapping design due to the reduced computation and minimal tunable parameters.

## A.2. Experiment Settings in Details

We here describe more experiment details in the bootstrapping procedure. The learning rate for tuning pre-trained transformer blocks is set to $0.1\times$ that of the focusing transformer to make the training more stable after the warmup. The focusing transformer in our designed SparseFormer variant performs the feature sampling first, then self attention between tokens, the feed-forward network, and then the RoI adjustment for each iteration, in contrast to the original SparseFormer which the self attention is performed first and the feature sampling then. We use this reversed order to prioritize the self-attention interaction between different tokens with sampled features. We use two-layered MLP to produce RoI adjusting deltas in the focusing transformer.

Different from the original SparseFormers without positional information into latent tokens, we inject RoI-based position encoding into tokens after every feature sampling operation in the focusing transformer to align with typical vision transformers. Our adopted positional encoding is also sinusoidal but in a continuous form:

$$\text{PE}_v = [\sin(\pi f_0 v), \cos(\pi f_0 v), \sin(\pi f_1 v), ...] \in \mathbb{R}^{d/4},$$

where $f_i$ is the frequency term that evenly lies in the exponential space from 1 to $f_{\text{max}} = 128$ (there are $d/8$ frequency terms), $v \in [v_{\text{left}}, v_{\text{top}}, v_{\text{right}}, v_{\text{bottom}}]$ where each component is the normalized coordinate of a token RoI that lies in $[0, 1]$. The final positional encoding is these four positional encoding parts concatenated:

$$\text{PE} = [\text{PE}_{\text{left}} | \text{PE}_{\text{top}} | \text{PE}_{\text{right}} | \text{PE}_{\text{bottom}}] \in \mathbb{R}^d.$$

## A.2. More Visualizations

We visualize the detailed RoI adjustments in each iteration in the focusing transformer of our bootstrapped SparseFormer SF-B$_{\text{AugReg}}$ in Figure 1 and 2. In addition to that, we perform visualizations on assorted bootstrapped SparseFormers (SF-B$_{\text{AugReg}}$, SF-L$_{\text{AugReg}}$, SF-B$_{\text{CLIP}}$, and SF-L$_{\text{CLIP}}$) on IN-1K val samples in Figure 3 and 4. Bootstrapped SparseFormers exhibit better sparsity and localization on foregrounds than the original SparseFormer.

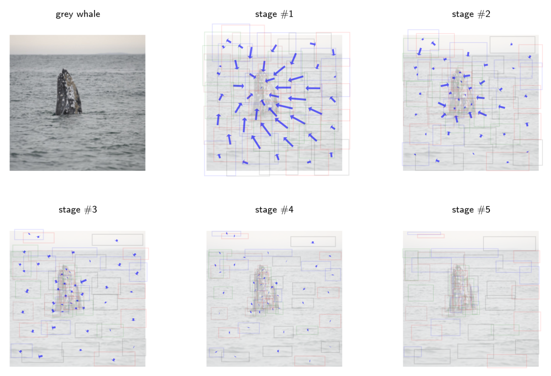

Figure 1. RoI adjustments in each iteration in SF-B$_{\text{AugReg}}$.

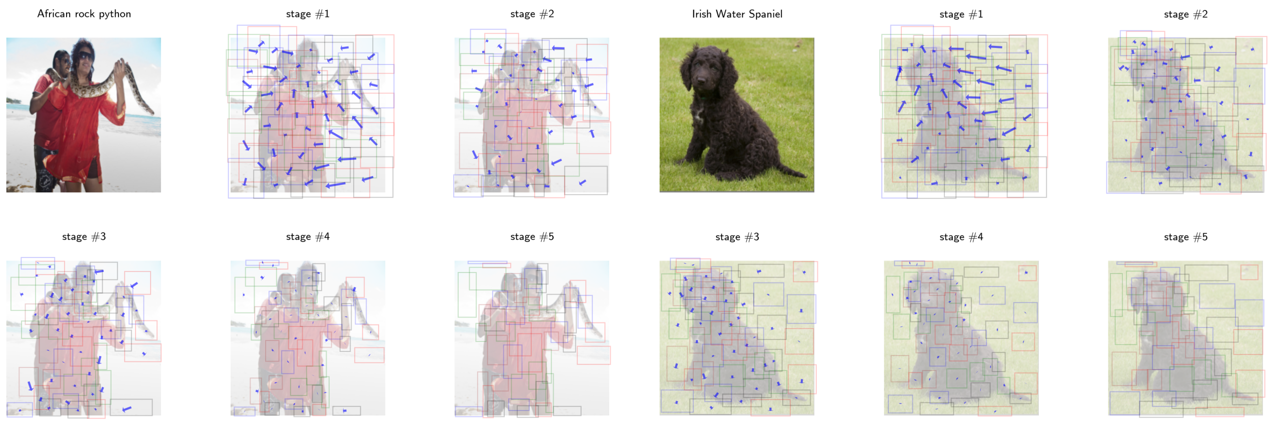

Figure 2. RoI adjustments (cont'd).

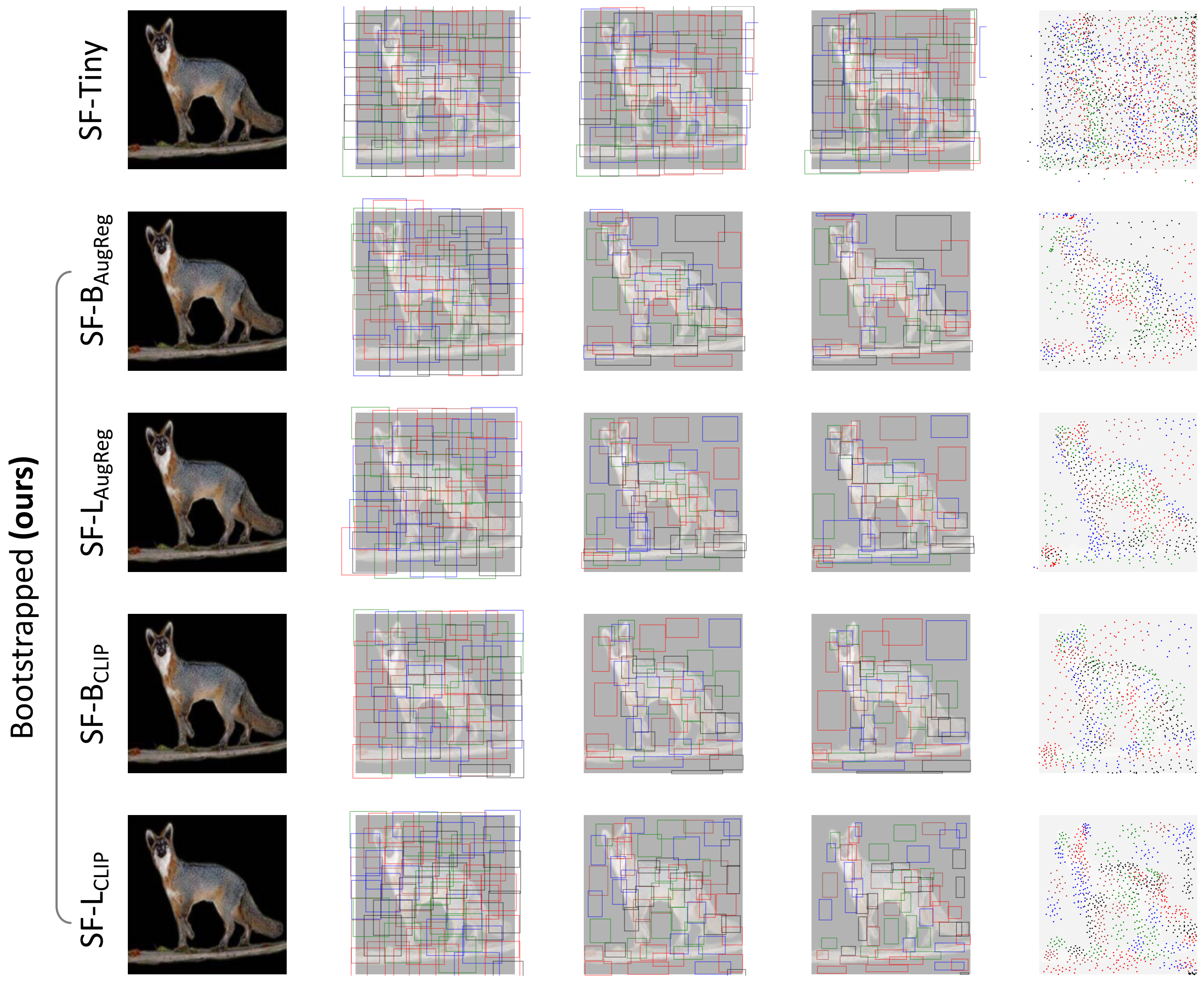

Figure 3. Visualizations on the original SparseFormer and our bootstrapped SparseFormers. For each image, there are an input image, token RoIs in the {first, third, last} stage, and sampling points in the last stage in the focusing transformer from left to right.

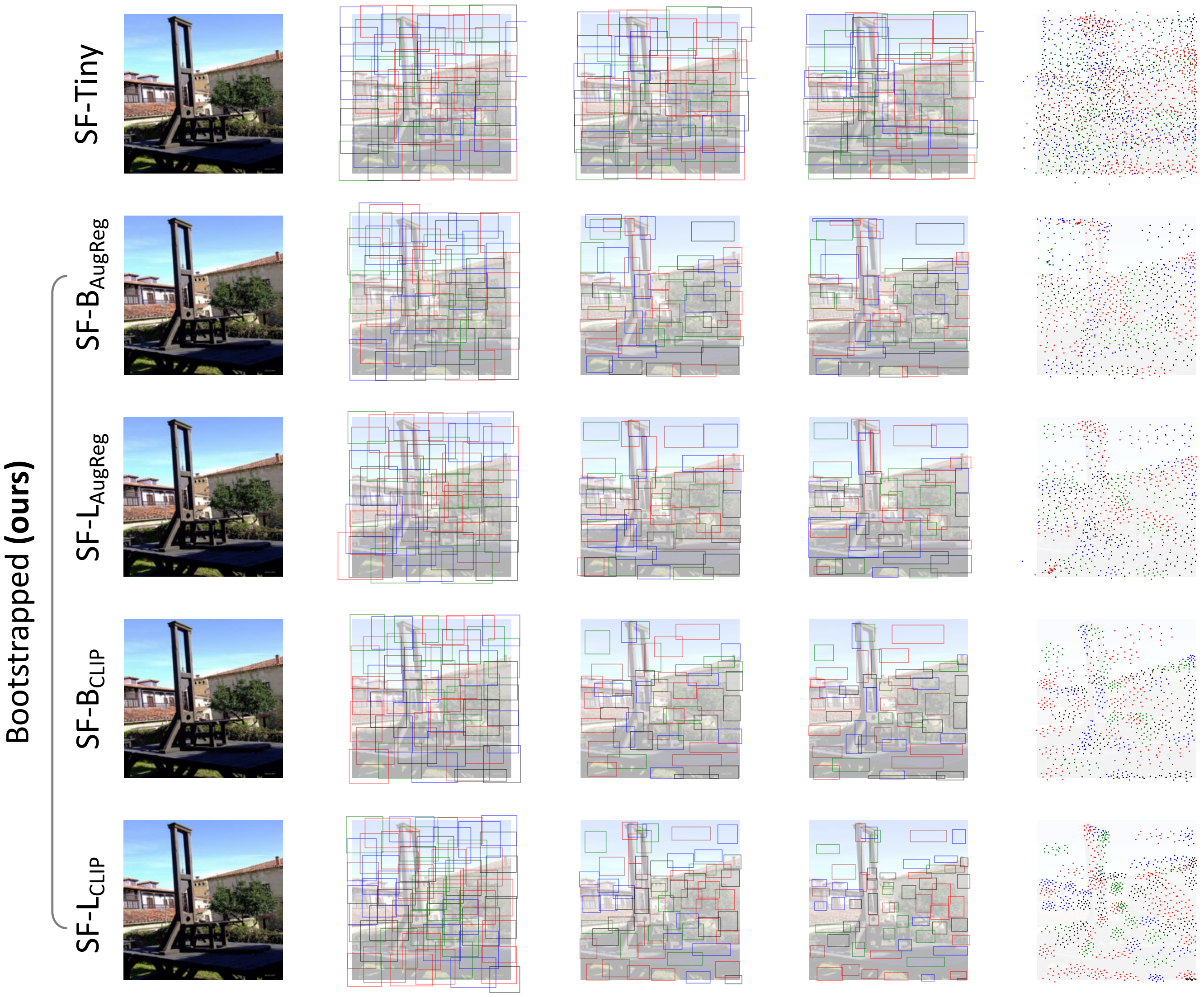
SF-Tiny
$\text{SF-B}_{\text{AugReg}}$
$\text{SF-L}_{\text{AugReg}}$
$\text{SF-B}_{\text{CLIP}}$
$\text{SF-L}_{\text{CLIP}}$
Bootstrapped **(ours)**

Figure 4. Visualizations (cont'd).