Nested Hierarchical Transformer: Towards Accurate, Data-Efficient and Interpretable Visual Understanding

Zizhao Zhang[1] **Han Zhang**[2] **Long Zhao**[2] **Ting Chen**[2] **Sercan Ö. Arık**[1] **Tomas Pfister**[1]

[1]Google Cloud AI [2]Google Research

Abstract

Hierarchical structures are popular in recent vision transformers, however, they require sophisticated designs and massive datasets to work well. In this paper, we explore the idea of nesting basic local transformers on non-overlapping image blocks and aggregating them in a hierarchical way. We find that the block aggregation function plays a critical role in enabling cross-block non-local information communication. This observation leads us to design a simplified architecture that requires minor code changes upon the original vision transformer. The benefits of the proposed judiciously-selected design are threefold: (1) NesT converges faster and requires much less training data to achieve good generalization on both ImageNet and small datasets like CIFAR; (2) when extending our key ideas to image generation, NesT leads to a strong decoder that is 8$\times$ faster than previous transformer-based generators; and (3) we show that decoupling the feature learning and abstraction processes via this nested hierarchy in our design enables constructing a novel method (named GradCAT) for visually interpreting the learned model. Source code is available https://github.com/google-research/nested-transformer.

Introduction

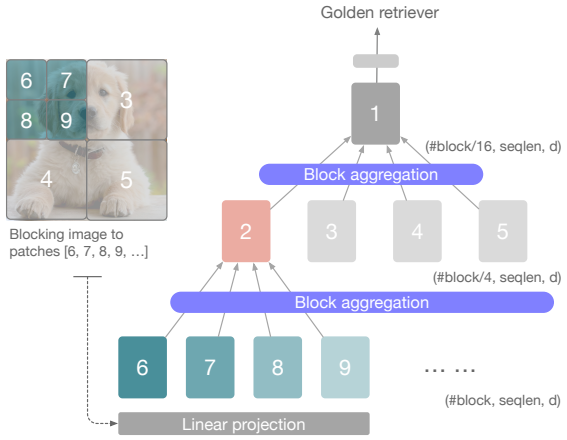
Vision Transformer (ViT) (Dosovitskiy et al. 2021) model and its variants have received significant interests recently due to their superior performance on many core visual applications (Cordonnier, Loukas, and Jaggi 2020; Liu et al. 2021). ViT first splits an input image into patches, and then patches are treated in the same way as tokens in NLP applications. Following, several self-attention layers are used to conduct global information communication to extract features for classification. Recent work (Dosovitskiy et al. 2021; Cordonnier, Loukas, and Jaggi 2020) shows that ViT models can achieve better accuracy than state-of-the-art convnets (Tan and Le 2019; He et al. 2016) when trained on datasets with tens or hundreds of millions of labeled samples. However, when trained on smaller datasets, ViT usually underperforms its counterparts based on convolutional layers. Addressing this data inefficiency is important to make ViT applicable to other application scenarios, e.g. semi-supervised learning (Sohn et al. 2020) and generative modeling (Goodfellow et al. 2014; Zhang et al. 2019).

Lack of inductive bias such as locality and translation equivariance, is one explanation for the data inefficiency of ViT models. Cordonnier, Loukas, and Jaggi (2020) discovered that transformer models learn locality behaviors in a deformable convolution manner (Dai et al. 2017): bottom layers attend locally to the surrounding pixels and top layers favor long-range dependency. On the other hand, global self-attention between pixel pairs in high-resolution images is computationally expensive. Reducing the self-attention range is one way to make the model training more computationally efficient (Beltagy, Peters, and Cohan 2020). These type of insights align with the recent structures with local self-attention and hierarchical transformer (Han et al. 2021; Vaswani et al. 2021; Liu et al. 2021). Instead of holistic global self-attention, these perform attention on local image patches. To promote information communication across patches, they propose specialized designs such as the "haloing operation" (Vaswani et al. 2021) and "shifted window" (Liu et al. 2021). These are based on modifying the self-attention mechanism and often yields in complex architectures. Our design goal on the other hand keeping the attention as is, and introducing the design of the aggregation function, to improve the accuracy and data efficiency, while bringing interpretability benefits.

The proposed NesT model stacks canonical transformer blocks to process non-overlapping image blocks individually. Cross-block self-attention is achieved by nesting these transformers hierarchically and connecting them with a proposed aggregation function. Fig. 1 illustrates the overall architecture and the simple pseudo code to generate it. Our contributions can be summarized as:

1. We demonstrate integrating hierarchically nested transformers with the proposed block aggregation function can outperform previous sophisticated (local) self-attention variants, leading to a substantially-simplified architecture and improved data efficiency. This provides a novel perspective for achieving effective cross-block communication.

2. NesT achieves impressive ImageNet classification accuracy with a significantly simplified architectural design. E.g., training a NesT with 38M/68M parameters obtains $83.3\%/83.8\%$ ImageNet accuracy.The favorable data efficiency of NesT is embodied by its fast convergence, such as achieving 75.9%/82.3% training with 30/100 epochs. Moreover, NesT achieves matched accuracy on small datasets compared with popular convolutional



Pseudo code: NesT

```
# embed and block image to (#block,seqlen,d)
x = Block(PatchEmbed(input_image))

for i in range(num_hierarchy):
  # apply transformer layers T_i within each block
  # with positional encodings (PE)
  y = Stack([T_i(x[0] + PE_i[0]), ...])
  if i < num_hierarchy - 1:
    # aggregate blocks and reduce #block by 4
    x = Aggregate(y, i)

h = GlobalAvgPool(x) # (1,seqlen,d) to (1,1,d)
logits = Linear(h[0,0]) # (num_classes,)

def Aggregate(x, i):
  z = UnBlock(x) # unblock seqs to (h,w,d)
  z = ConvNormMaxPool_i(x) # (h/2,w/2,d)
  return Block(z) # block to seqs
```

Figure 1: (Left) Illustration of NesT with nested transformer hierarchy; (right) the simple pseudo code to generate the architecture. Each node T_i processes an image block. The block aggregation is performed between hierarchies (num_hierarchy= 3 here) to achieve cross-block communication on the image (feature map) plane.

architectures. E.g., training a NesT with 6M parameters using a single GPU results in 96% accuracy on CIFAR10 .

3. We show that when extending this idea beyond classification to image generation, NesT can be repurposed into a strong decoder that achieves better performance than convolutional architectures meanwhile has comparable speed, demonstrated by 64×64 ImageNet generation, which is an important to be able to adopt transformers for efficient generative modeling.
4. Our proposed architectural design leads to decoupled feature learning and abstraction, which has significant interpretability benefits. To this end, we propose a novel method called GradCAT to interpret NesT reasoning process by traversing its tree-like structure. This providing a new type of visual interpretability that explains how aggregated local transformers selectively process local visual cues from semantic image patches.

Related Work

Vision transformer-based models (Cordonnier, Loukas, and Jaggi 2020; Dosovitskiy et al. 2021) and self-attention mechanisms (Vaswani et al. 2021; Ramachandran et al. 2019) have recently attracted significant interest in the research community, with explorations of more suitable architectural designs that can learn visual representation effectively, such as injecting convolutional layers (Li et al. 2021; Srinivas et al. 2021; Yuan et al. 2021) and building local or hierarchical structures (Zhang et al. 2021b; Wang et al. 2021b). Existing methods focus on designing a variety of self-attention modifications. Hierarchical ViT structures becomes popular both in vision (Liu et al. 2021; Vaswani et al. 2021) and NLP (Zhang, Wei, and Zhou 2019; Santra, Anusha, and Goyal 2020; Liu and Lapata 2019; Pappagari et al. 2019). However, many methods often add significant architectural complexity in order to optimize accuracy.

One challenge for vision transformer-based models is data efficiency. Although the original ViT (Dosovitskiy et al. 2021) can perform better than convolutional networks with hundreds of millions images for pre-training, such a data requirement is not always practical. Data-efficient ViT (DeiT) (Touvron et al. 2020, 2021) attempts to address this problem by introducing teacher distillation from a convolutional network. Although promising, this increases the supervised training complexity, and existing reported performance on data efficient benchmarks (Hassani et al. 2021; Chen et al. 2021) still significantly underperforms convolutional networks. Since ViT has shown to improve vision tasks beyond image classification, with prior work studying its applicability to generative modeling (Parmar et al. 2018; Child et al. 2019; Jiang, Chang, and Wang 2021; Hudson and Zitnick 2021), video understanding (Neimark et al. 2021; Akbari et al. 2021), segmentation and detection (Wang et al. 2021a; Liang et al. 2020; Kim et al. 2021), interpretability (Chefer, Gur, and Wolf 2021; Abnar and Zuidema 2020), a deeper understanding of the data efficiency and training difficulties from the architectural perspective is of significant impact.

Proposed Method

Main Architecture

According to Fig. 1, our overall design stacks canonical transformer layers to conduct local self-attention on every image block independently, and then nests them hierarchically. Coupling of processed information between spatially adjacent blocks is achieved through a proposed block aggregation between every two hierarchies. The overall hierarchical structure can be determined by two key hyper-parameters: patch size $S \times S$ and number of block hierarchies T_d. All blocks inside each hierarchy share one set of parameters.

Given an input of image with shape $H \times W \times 3$, each image patch with size $S \times S$ is linearly projected to an embedding in $\mathbb{R}^d$. Then, all embeddings are partitioned to blocks and flattened to generate input $X \in \mathbb{R}^{b \times T_n \times n \times d}$, where b is the batch size, T_n is the total number of blocks at bottom of the

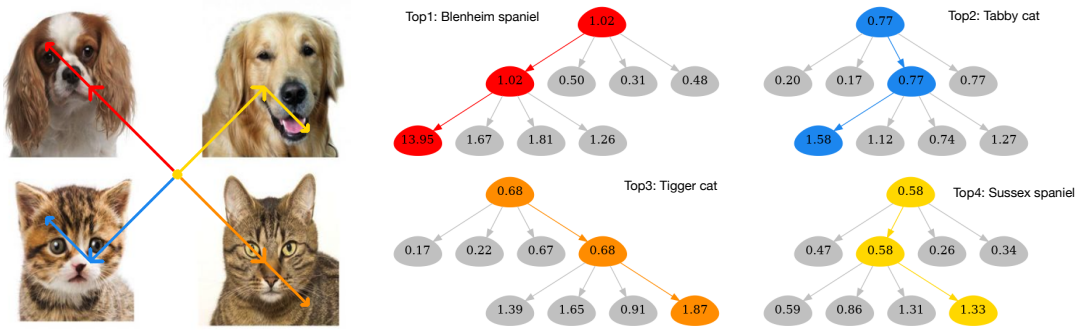


Figure 2: Example results of the proposed GradCAT. Given the left input image (containing four objects), the figure visualizes the top-4 class traversal results (4 colors) using an ImageNet-trained NesT (with three tree hierarchies). Each tree node denotes the averaged activation value ($\hat{h}_l$ defined in Algorithm 1). The traversals can correctly find the model decision path along the tree to locate an image patch belonging to the objects of given target classes.

NesT hierarchy, and n is the sequence length (the number of embeddings) at each block. Note that $T_n \times n = H \times W/S^2$.

Inside each block, we stack a number of canonical transformer layers, where each is composed of a multi-head self-attention (MSA) layer followed by a feed-forward fully-connected network (FFN) with skip-connection (He et al. 2016) and Layer normalization (LN) (Ba, Kiros, and Hinton 2016). Trainable positional embedding vectors (Touvron et al. 2020) are added to all sequence vectors in $\mathbb{R}^d$ to encode spatial information before feeding into the block function T:

$$\text{multiple } \times \begin{cases} y = x + \text{MSA}_{\text{NesT}}(x', x', x'), \; x' = \text{LN}(x) \\ x = y + \text{FFN}(\text{LN}(y)) \end{cases} \tag{1}$$

The FFN is composed of two layers: $\max(0, xW_1 + b)W_2 + b$. Given input $X \in \mathbb{R}^{b \times T_n \times n \times d}$, since all blocks at one NesT hierarchy share the same parameters, MSA_{NesT} basically MSA is applied (Vaswani et al. 2017) to all blocks in parallel:

$$\begin{aligned} \text{MSA}_{\text{NesT}}(Q, K, V) &= \text{Stack}(\text{block}_1, ..., \text{block}_{T_n}), \\ \text{where block}_i &= \text{MSA}(Q, K, V)W^O. \end{aligned} \tag{2}$$

block_i has shape $b \times n \times d$. Lastly, we build a nested hierarchy with block aggregation – every four spatially connected blocks are merged into one. The overall design makes NesT easy to implement, requiring minor code changes to the original ViT.

Block Aggregation

From a high-level view, NesT leads to hierarchical representations, which share similarity with several pyramid designs (Zhang et al. 2021b; Wang et al. 2021b). However, most of these works use global self-attention throughout the layers, interleaved with (spatial) down-sampling. In contrast, we show that NesT, which leverages local attention, can lead to significantly improved data efficiency. In local self-attention, non-local communication is important to maintain translational equivariance (Vaswani et al. 2021). To this end, Halonet (Vaswani et al. 2021) allows the query to attend to slightly larger regions than the assigned block. Swin Transformer (Liu et al. 2021) achieves this by shifting the block partition windows between consecutive self-attention layers to connect adjacent blocks; applying special masked self-attention to guarantee spatial continuity. However, both add complexity to the self-attention layers and such sophisticated architectures are not desired from implementation perspective.

On the other hand, every block in NesT processes information independently via standard transformer layers, and only communicate and mix global information during the block aggregation step via simple spatial operations (e.g. convolution and pooling). One key ingredient of block aggregation is to perform it in the image plane so that information can be exchanged between nearby blocks. This procedure is summarized in Fig. 1. The output $X_l \in \mathbb{R}^{b \times \#\text{block} \times n \times d}$ at hierarchy l is unblocked to the full image plane $A_l \in \mathbb{R}^{b \times H' \times W' \times d'}$. A number of spatial operations are applied to down-sample feature maps $A'_l \in \mathbb{R}^{b \times H'/2 \times W'/2 \times d}$. Finally, the feature maps are blocked back to $X_{l+1} \in \mathbb{R}^{b \times \#\text{block}/4 \times n \times d'}$ for hierarchy $l+1$. The sequence length n always remains the same and the total number of blocks is reduced by a factor of 4, until reduced to 1 at the top (i.e. $\#\text{block}/4^{(T_d - 1)} = 1$). Therefore, this process naturally creates hierarchically nested structure where the "receptive field" expands gradually. $d' \geq d$ depends on the specific model configuration.

Our block aggregation is specially instantiated as a 3×3 convolution followed by LN and a 3×3 max pooling. Figure A2 in Appendix explains the core design and the importance of applying it on the image plane (i.e. full image feature maps) versus the block plane (i.e. partial feature maps corresponding to 2×2 blocks that will be merged). The small information exchange through the small convolution and max. pooling kernels across block boundaries are particularly important. We conduct comprehensive ablation studies to demonstrate the importance of each of the design components.

Note that the resulting design shares some similarities with recent works that combine transformer and convolutional networks (Wu et al. 2021; Yuan et al. 2021; Bello 2021) as specialized hybrid structures. However, unlike these, our proposed method aims to solve cross-block communications in local self-attention, and the resulting architecture is simple as a stacking of basic transformer layers.

Transposed NesT for Image Generation

The data efficiency and straightforward implementation of NesT makes it desirable for more complex learning tasks. With transpose the key ideas from NesT to propose a decoder for generative modeling, and show that it has better performance than convolutional decoders with comparable speed. Remarkably, it is nearly a magnitude faster than the transformer-based decoder TransGAN (Jiang, Chang, and Wang 2021).

Creating such a generator is straightforward by transposing NesT (see Table A6 of Appendix for architecture details). The input of the model becomes a noise vector and the output is a full-sized image. To support the gradually increased number of blocks, the only modification to NesT is replacing the block aggregation with appropriate block de-aggregation, i.e. up-sampling feature maps (we use pixel shuffle (Shi et al. 2016)). The feature dimensions in all hierarchies are $(b, nd) \rightarrow (b, 1, n, d) \rightarrow (b, 4, n, d'), ..., \rightarrow (b, \#\text{blocks}, n, 3)$. The number of blocks increases by a factor of 4. Lastly, we can unblock the output sequence tensor to an image with shape $H \times W \times 3$. The remaining adversarial training techniques are based on (Goodfellow et al. 2014; Zhang et al. 2019) as explained in experiments. Analogous to our results for image classification, we show the importance of careful block de-aggregation design, in making the model significantly faster while achieving better generation quality.

GradCAT: Interpretability via Tree Traversal

Different from previous work, the nested hierarchy with the independent block process in NesT resembles a decision tree in which each block is encouraged to learn non-overlapping features and be selected by the block aggregation. This unique behavior motivates us to explore a new method to explain the model reasoning, which is an important topic with significant real world impact in convnets (Selvaraju et al. 2017; Sundararajan, Taly, and Yan 2017).

Algorithm 1: GradGAT

Define: A_l denotes the feature maps at hierarchy l. Y_c is the logit of predicted class c. $[\cdot]_{2\times2}$ indexes one of 2×2 partitions of input maps.
Input: $\{A_l | l = 2, ..., T_d\}, \alpha_{T_d} = A_{T_d}, P = []$
Output: The traversal path P from top to bottom
for $l = [T_d, ..., 2]$ **do**
  $h_l = \alpha_l \cdot (-\frac{\partial Y_c}{\alpha_l})$ *# obtain target activation maps*
  $\hat{h}_l = \text{AvgPool}_{2\times2}(h_l) \in \mathbb{R}^{2\times2}$
  $n_l^* = \arg\max \hat{h}_l, \quad P = P + [n_l^*]$ *# pick the maximum index*
  $\alpha_l = A_l[n_l^*]_{2\times2}$ *# obtain the partition for the index*
end for

We present a gradient-based class-aware tree-traversal (GradCAT) method (Algorithm 1). The main idea is to find the most valuable traversal from a child node to the root node that contributes to the classification logits the most. Intuitively, at the top hierarchy, each of four child nodes processes one of 2×2 non-overlapping partitions of feature maps A_{T_d}. We can use corresponding activation and class-specific gradient features to trace the high-value information flow recursively from the root to a leaf node. The negative gradient $-\frac{\partial Y_c}{A_l}$ provides the gradient ascent direction to maximize the class c logit, i.e., a higher positive value means higher importance. Fig. 2 illustrates a sample result.

Table 1: Test accuracy on CIFAR with input size 32×32. The compared convolutional architectures are optimized models for CIFAR. All transformer-based architectures are trained from random initialization with the same data augmentation. DeiT uses $S = 2$. Swin and our NesT uses $S = 1$. * means model tends to diverge.

Arch. base	Method	C10 (%)	C100 (%)
Convolutional	Pyramid-164-48	95.97	80.70
	WRN28-10	95.83	80.75
Transformer full-attention	DeiT-T	88.39	67.52
	DeiT-S	92.44	69.78
	DeiT-B	92.41	70.49
	PVT-T	90.51	69.62
	PVT-S	92.34	69.79
	PVT-B	$85.05^\star$	$43.78^\star$
	CCT-7/3×1	94.72	76.67
Transformer local-attention	Swin-T	94.46	78.07
	Swin-S	94.17	77.01
	Swin-B	94.55	78.45
	NesT-T	96.04	78.69
	NesT-S	96.97	81.70
	NesT-B	**97.20**	**82.56**

Experiments

We first show the benefit of NesT for data efficient learning and then demonstrate benefits for interpretability and generative modeling. Finally, we present ablation studies to analyze the major constituents of the methods.

Experimental setup. We follow previous work (Dosovitskiy et al. 2021) to generate three architectures that have comparable capacity (in number of parameters and FLOPS), noted as tiny (NesT-T), small (NesT-S), and base (NesT-B). Most recent ViT-based methods follow the training techniques of DeiT (Touvron et al. 2020). We follow the settings with minor modifications that we find useful for local self-attention (see Appendix for all architecture and training details). We do not explore the specific per-block configurations (e.g. number of heads and hidden dimensions), which we believe can be optimized through architecture search (Tan and Le 2019).

Comparisons to Previous Work

CIFAR. We compare NesT to recent methods on CIFAR datasets (Krizhevsky, Hinton et al. 2009) in Table 1, to investigate the data efficiency. It is known that transformer-based methods usually perform poorly on such tasks as they typically require large datasets to be trained on. The models that perform well on large-scale ImageNet do not necessary work perform on small-scale CIFAR, as the full self-attention based models require larger training datasets. DeiT (Touvron et al. 2020) performs poorly and does not improve given

Table 2: Comparison on the ImageNet dataset. All models are trained from random initialization. ViT-B/16 uses an image size 384 and others use 224.

Arch. base	Method	#Params	Top-1 acc. (%)
Convolutional	ResNet-50	25M	76.2
	RegNetY-4G	21M	80.0
	RegNetY-16G	84M	82.9
Transformer full-attention	ViT-B/16	86M	77.9
	DeiT-S	22M	79.8
	DeiT-B	86M	81.8
Transformer local-attention	Swin-T	29M	81.3
	Swin-S	50M	83.0
	Swin-B	88M	83.3
	NesT-T	17M	81.5
	NesT-S	38M	83.3
	NesT-B	68M	**83.8**

Table 3: Comparison on ImageNet benchmark with ImageNet-22K pre-training.

	ViT-B/16	Swin-B	Nest-B
ImageNet Acc. (%)	84.0	86.0	**86.2**

bigger model size. PVT (Wang et al. 2021b) has also a full self-attention based design, though with a pyramid structure. PVT-T seems to perform better than DeiT-T when model size is small, however, the performance largely drops and becomes unstable when scaling up, further suggesting that full self-attention at bottom layers is not desirable for data efficiency. Other transformer-based methods improve slowly with increasing model size, suggesting that bigger models are more challenging to train with less data. We attribute this to to their complex design (i.e. shifted windows with masked MSA) requiring larger training datasets, while NesT benefiting from a judiciously-designed block aggregation. We also include comparisons with convolutional architectures that are specifically optimized for small CIFAR images and show that NesT can give better accuracy without any small dataset specific architecture optimizations (while still being larger and slower, as they do not incorporate convolutional inductive biases). The learning capacity and performance of NesT get better with increased model size. Most variants of NesT in Fig. A1 of Appendix outperform compared methods with far better throughput. E.g., NesT$_3$-T ($S = 2$) leads to 94.5% CIFAR10 accuracy with 5384 images/s throughout, $10\times$ faster than the best compared result 94.6% accuracy. More details can be found in Appendix.

ImageNet. We test NesT on standard ImageNet 2012 benchmarks (Deng et al. 2009) with commonly used 300 epoch training on TPUs in Table 2. The input size is 224×224 and no extra pre-training data is used. DeiT does not use teacher distillation, so it can be viewed as ViT (Dosovitskiy et al. 2021) with better data augmentation and regularization. NesT matches the performance of prior work with a significantly more straightforward design (e.g. NesT-S matches the accuracy of Swin-B, 83.3%). The results of NesT suggest that correctly aggregating the local transformer can improve the performance of local self-attention.

ImageNet-22K. We scale up NesT to ImageNet-22K following the exact training schedules in (Liu et al. 2021; Dosovitskiy et al. 2021). The pre-training is 90 epoch on 224×224 ImageNet21K images and finetuning is 30 epoch on 384×384 ImageNet images. Table 3 compares the results. NesT again achieves competitive results, with a significantly more straightforward design.

Visual Interpretability

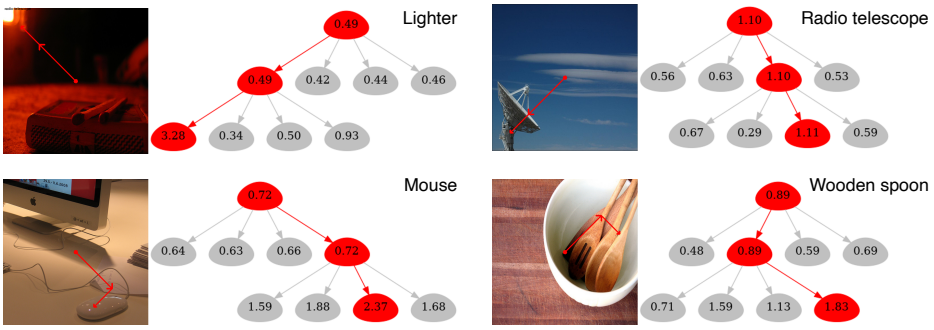
GradGAT results. Fig. 3 (left) shows the explanations obtained with the proposed GradGAT. For GradGAT, each tree node corresponds to a value that reflects the mean activation strength. Visualizing the tree traversal through image blocks, we can get insights about the decision making process of NesT. The traversal passes through the path with the highest values. As can be seen, the decision path can correctly locate the object corresponding to the model prediction. The Lighter example is particularly interesting because the ground truth class – lighter/matchstick – actually defines the bottom-right matchstick object, while the most salient visual features (with the highest node values) are actually from the upper-left red light, which conceptually shares visual cues with a lighter. Thus, although the visual cue is a mistake, the output prediction is correct. This example reveals the potential of using GradGAT to conduct model diagnosis at different tree hierarchies. Fig. A5 of Appendix shows more examples.

Class attention map (CAM) results. In contrast to ViT (Dosovitskiy et al. 2021) which uses class tokens, NesT uses global average pooling before softmax. This enables conveniently applying CAM-like (Zhou et al. 2016) methods to interpret how well learned representations measure object features, as the activation coefficients can be directly without approximate algorithms. Fig. 3(right) shows quantitative evaluation of weakly-supervised object localization, which is a common evaluation metric for CAM-based methods (Zhou et al. 2016), including GradCAM++ (Chattopadhay et al. 2018) with ResNet50 (He et al. 2016), DeiT with Rollout attention (Abnar and Zuidema 2020), and our NesT CAM (Zhou et al. 2016). We follow (Gildenblat 2021) in using an improved version of Rollout. NesT with standard CAM, outperforms others that are specifically designed for this task. Fig. 4 shows a qualitative comparison (details in the Appendix), exemplifying that NesT can generate clearer attention maps which converge better on objects.

Overall, decoupling local self-attention (transformer blocks) and global information selection (block aggregation), which is unique to our work, shows significant potential for making models easier to interpret.

Generative Modeling with Transposed NesT

We evaluate the generative ability of Transposed NesT on ImageNet (Russakovsky et al. 2015) where all images are resized to 64×64 resolution. We focus on the unconditional image generation setting to test the effectiveness of different decoders. We compare Transposed NesT to TransGAN (Jiang, Chang, and Wang 2021), that uses a full Transformer as the generator, as well as a convolutional baseline following the



Method	Top-1 loc. err (%)
DeiT RollOut[§]	67.6
ADL (Choe and Shim 2019)	55.1
ACoL (Zhang et al. 2018b)	54.2
R50 GradCAM++[§]	53.5
NesT CAM	**51.8**

Figure 3: Left: Output visualization of the proposed GradGAT. Tree nodes annotate the averaged responses to the predicted class. We use a NesT-S with three tree hierarchies. Right: CAM-based weakly supervised localization comparison on the ImageNet validation set. [§] indicates results obtained by us.

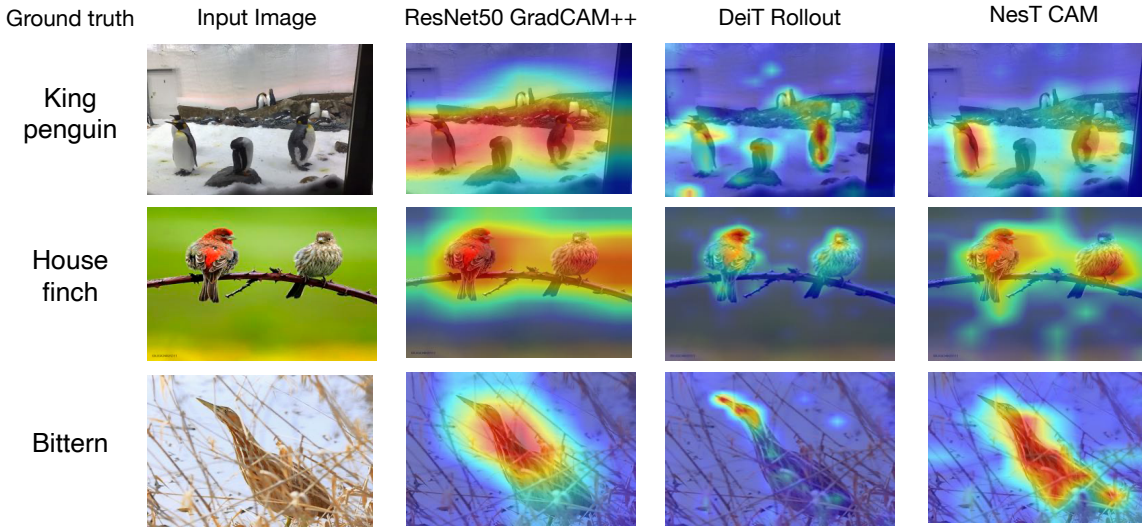


Figure 4: Visualization of CAM-based attention results. All models are trained on ImageNet. CAM (vanilla) with NesT achieves accurate attention patterns on object regions, yielding finer attention to objects than DeiT Rollout (Abnar and Zuidema 2020) and less noise than ResNet50 GradCAM++ (Chattopadhay et al. 2018).

widely-used architecture from (Zhang et al. 2019) (its computationally expensive self-attention module is removed). Fig. 5 shows the results. Transposed NesT obtains significantly faster convergence and achieves the best FID and Inception score (see Fig. A6 of Appendix for results). Most importantly, it achieves $8\times$ throughput over TransGAN, showing its potential for significantly improving the efficiency of transformer-based generative modeling. More details are explained in the Appendix.

It is noticeable from Fig. 5 (middle) that appropriate unsampling (or block de-aggregation) impacts the generation quality. Pixel shuffle (Shi et al. 2016) works the best and the margin is considered surprisingly large compared to other alternatives widely-used in convnets. This aligns with our main findings in classification, suggesting that judiciously injecting spatial operations is important for nested local transformers to perform well.

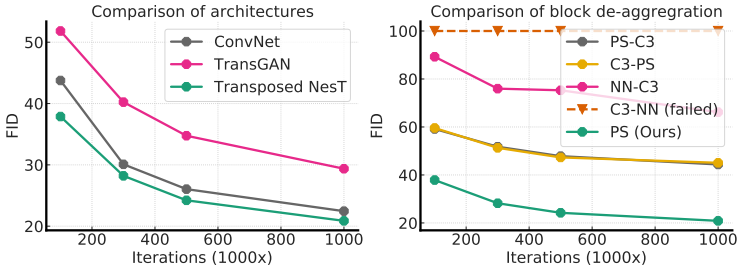
Ablation Studies

We summarize key ablations below (more in Appendix).

Fast convergence. NesT achieves fast convergence, as shown in Fig. 6 (top) for Imagenet training with $\{30, 60, 100, 300\}$ epochs. NesT-B merely loses 1.5% when reducing the training epoch from 300 to 100. The results suggest that NesT can learn effective visual features faster and more efficiently.

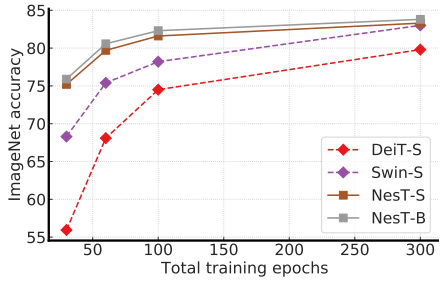
Less sensitivity to data augmentation. NesT uses several kinds of data augmentation following (Touvron et al. 2020). As shown in Fig. 6 (right) and Fig. A4, our method shows higher stability in data augmentation ablation studies compared to DeiT. Data augmentation is critical for global self-attention to generalize well, but reduced dependence on domain or task dependent data augmentation helps with generalization to other tasks.

Impact of block aggregation. Here we show that the design of block aggregation is critical for performance and data efficiency. We study this from four perspectives: (1) whether unblocking to the full image plane is necessary; (2) how to use convolution; (3) what kinds of pooling to use; and (4) whether to perform query down-sampling inside



Method	#Params (millions)	Throughput (images/s)
Convnet (Zhang et al. 2019)	77.8M	709.1
TransGAN (Jiang, Chang, and Wang 2021)	82.6M	67.7
Transposed NesT	74.4M	523.7

Figure 5: Left: FID comparison for 64×64 ImageNet generation at different training iterations. Middle: FID comparison of different popular un-sampling methods for block de-aggregation, including combinations of pixel shuffling (PS), Conv3x3 (C3), and nearest neighbor (NN). Right: The number of parameters and throughput of compared generators.



Augmentation Removed	ImageNet Accuracy (%)	
	DeiT-B	NesT-T
None	81.8	81.5
RandomErasing	4.3	81.4
RandAugment	79.6	81.2
CutMix&MixUp	75.8	79.8

Figure 6: Top: Training convergence. NesT achieves better performance than DeiT with the same total epoch of training (each point is a single run). Bottom: Data augmentation ablations. Results of DeiT-B (Touvron et al. 2020) are reported by its paper. NesT shows less reliance on data augmentation.

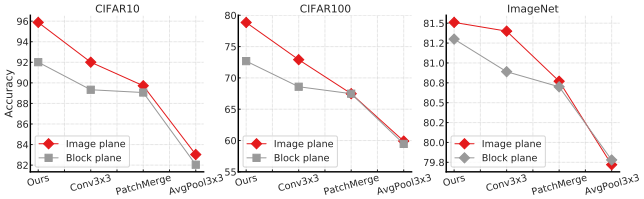


Figure 7: Demonstration of the impact of block aggregation on CIFAR and ImageNet. NesT-T is used. Conv3x3 has stride 2. AvgPool3x3 on ImageNet is followed by Conv1x1 to change hidden dimensions of self-attention. Four plausible block aggregation designs are shown in x-axis, and applied on the image plane and block plane both for comparison. Note that Ours in x-axis is Conv3x3 followed by LN and MaxPool3x3 (stride 2). More alternatives are validated in Fig. A3 of Appendix.

self-attention (Vaswani et al. 2021). Fig. 7 and Fig. A3 of Appendix compare the results of different plausible designs.

The results show that: (1) when performing these spatial operations, it is important to apply it on the holistic image plane versus the block plane although both can reasonably introduce spatial priors; (2) small kernel convolution is sufficient and has to be applied ahead of pooling; (3) max. pooling is far better than other options, such as stride-2 sub-sampling and average pooling; (4) sub-sampling the query sequence length (similar to performing sub-sampling on the block plane as illustrated in Fig. A2), as used by Halonet (Vaswani et al. 2021), performs poorly on data efficient benchmarks. We also experiment PatchMerge from Swin Transformer (Liu et al. 2021) on both CIFAR and ImageNet. Our block aggregation closes the accuracy gap on ImageNet, suggesting that a conceptually negligible difference in aggregating nested transformers can lead to significant differences in model performance.

Conclusion

We have shown that aggregating nested transformers can match the accuracy of previous more complex methods with significantly improved data efficiency and convergence speed. In addition, we have shown that this idea can be extended to image generation, where it provides significant speed gains. Finally, we have shown that the decoupled feature learning and feature information extraction in this nested hierarchy design allows for better feature interpretability through a new gradient-based class-aware tree traversal method. In future work we plan to generalize this idea to non-image domains.

References

Abnar, S.; and Zuidema, W. 2020. Quantifying attention flow in transformers. *arXiv preprint arXiv:2005.00928*. 2, 5, 6, 12

Akbari, H.; Yuan, L.; Qian, R.; Chuang, W.-H.; Chang, S.-F.; Cui, Y.; and Gong, B. 2021. VATT: Transformers for Multimodal Self-Supervised Learning from Raw Video, Audio and Text. *arXiv preprint arXiv:2104.11178*. 2

Ba, J. L.; Kiros, J. R.; and Hinton, G. E. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*. 3

Bello, I. 2021. Lambdanetworks: Modeling long-range interactions without attention. *arXiv preprint arXiv:2102.08602*. 3

Beltagy, I.; Peters, M. E.; and Cohan, A. 2020. Long-

former: The long-document transformer. *arXiv preprint arXiv:2004.05150*. 1

Chattopadhay, A.; Sarkar, A.; Howlader, P.; and Balasubramanian, V. N. 2018. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *WACV*. 5, 6, 12

Chefer, H.; Gur, S.; and Wolf, L. 2021. Generic Attention-model Explainability for Interpreting Bi-Modal and Encoder-Decoder Transformers. *arXiv preprint arXiv:2103.15679*. 2

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020a. A simple framework for contrastive learning of visual representations. In *ICML*. 10

Chen, T.; Kornblith, S.; Swersky, K.; Norouzi, M.; and Hinton, G. 2020b. Big self-supervised models are strong semi-supervised learners. *NeurIPS*. 10

Chen, Z.; Xie, L.; Niu, J.; Liu, X.; Wei, L.; and Tian, Q. 2021. Visformer: The Vision-friendly Transformer. *arXiv preprint arXiv:2104.12533*. 2

Child, R.; Gray, S.; Radford, A.; and Sutskever, I. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*. 2

Choe, J.; and Shim, H. 2019. Attention-based dropout layer for weakly supervised object localization. In *CVPR*. 6

Cordonnier, J.-B.; Loukas, A.; and Jaggi, M. 2020. On the relationship between self-attention and convolutional layers. *ICLR*. 1, 2

Cubuk, E. D.; Zoph, B.; Shlens, J.; and Le, Q. V. 2020. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPR Workshops*. 10

Dai, J.; Qi, H.; Xiong, Y.; Li, Y.; Zhang, G.; Hu, H.; and Wei, Y. 2017. Deformable convolutional networks. In *ICCV*. 1

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*. 5

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*. 1, 2, 4, 5, 10, 12

Gildenblat, J. 2021. Exploring Explainability for Vision Transformers. 5, 12

Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial networks. *NeurIPS*. 1, 4

Han, D.; Kim, J.; and Kim, J. 2017. Deep pyramidal residual networks. In *CVPR*. 12

Han, K.; Xiao, A.; Wu, E.; Guo, J.; Xu, C.; and Wang, Y. 2021. Transformer in transformer. *arXiv preprint arXiv:2103.00112*. 1

Hassani, A.; Walton, S.; Shah, N.; Abuduweili, A.; Li, J.; and Shi, H. 2021. Escaping the Big Data Paradigm with Compact Transformers. *arXiv preprint arXiv:2104.05704*. 2, 10, 12

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*. 1, 3, 5, 10, 12

Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; and Hochreiter, S. 2017. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*. 11

Hoffer, E.; Ben-Nun, T.; Hubara, I.; Giladi, N.; Hoefler, T.; and Soudry, D. 2020. Augment Your Batch: Improving Generalization Through Instance Repetition. In *CVPR*. 10

Huang, G.; Sun, Y.; Liu, Z.; Sedra, D.; and Weinberger, K. Q. 2016. Deep networks with stochastic depth. In *ECCV*. 10

Hudson, D. A.; and Zitnick, C. L. 2021. Generative Adversarial Transformers. *arXiv preprint arXiv:2103.01209*. 2

Jiang, Y.; Chang, S.; and Wang, Z. 2021. Transgan: Two transformers can make one strong gan. *arXiv preprint arXiv:2102.07074*. 2, 4, 5, 7

Karras, T.; Laine, S.; Aittala, M.; Hellsten, J.; Lehtinen, J.; and Aila, T. 2020. Analyzing and improving the image quality of stylegan. In *CVPR*. 11

Kim, B.; Lee, J.; Kang, J.; Kim, E.-S.; and Kim, H. J. 2021. HOTR: End-to-End Human-Object Interaction Detection with Transformers. *CVPR*. 2

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. In *ICLR*. 11

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images. 4

Li, Y.; Zhang, K.; Cao, J.; Timofte, R.; and Van Gool, L. 2021. LocalViT: Bringing Locality to Vision Transformers. *arXiv preprint arXiv:2104.05707*. 2

Liang, J.; Homayounfar, N.; Ma, W.-C.; Xiong, Y.; Hu, R.; and Urtasun, R. 2020. Polytransform: Deep polygon transformer for instance segmentation. In *CVPR*. 2

Liu, Y.; and Lapata, M. 2019. Hierarchical transformers for multi-document summarization. *arXiv preprint arXiv:1905.13164*. 2

Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; and Guo, B. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*. 1, 2, 3, 5, 7, 10, 12, 13

Loshchilov, I.; and Hutter, F. 2018. Fixing weight decay regularization in adam. 10

Mescheder, L.; Geiger, A.; and Nowozin, S. 2018. Which training methods for GANs do actually converge? In *ICML*. 11

Neimark, D.; Bar, O.; Zohar, M.; and Asselmann, D. 2021. Video transformer network. *arXiv preprint arXiv:2102.00719*. 2

Pappagari, R.; Zelasko, P.; Villalba, J.; Carmiel, Y.; and Dehak, N. 2019. Hierarchical transformers for long document classification. In *Automatic Speech Recognition and Understanding Workshop (ASRU)*, 838–844. 2

Parmar, N.; Vaswani, A.; Uszkoreit, J.; Kaiser, L.; Shazeer, N.; Ku, A.; and Tran, D. 2018. Image transformer. In *ICML*. 2

Radosavovic, I.; Kosaraju, R. P.; Girshick, R.; He, K.; and Dollár, P. 2020. Designing network design spaces. In *CVPR*. 12

Ramachandran, P.; Parmar, N.; Vaswani, A.; Bello, I.; Levskaya, A.; and Shlens, J. 2019. Stand-alone self-attention in vision models. *NeurIPS*. 2

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. ImageNet large scale visual recognition challenge. *IJCV*. 5

Santra, B.; Anusha, P.; and Goyal, P. 2020. Hierarchical transformer for task oriented dialog systems. *arXiv preprint arXiv:2011.08067*. 2

Selvaraju, R. R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; and Batra, D. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*. 4, 12

Shi, W.; Caballero, J.; Huszár, F.; Totz, J.; Aitken, A. P.; Bishop, R.; Rueckert, D.; and Wang, Z. 2016. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*. 4, 6, 11

Sohn, K.; Berthelot, D.; Li, C.-L.; Zhang, Z.; Carlini, N.; Cubuk, E. D.; Kurakin, A.; Zhang, H.; and Raffel, C. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *NeurIPS*. 1

Srinivas, A.; Lin, T.-Y.; Parmar, N.; Shlens, J.; Abbeel, P.; and Vaswani, A. 2021. Bottleneck transformers for visual recognition. *arXiv preprint arXiv:2101.11605*. 2

Sundararajan, M.; Taly, A.; and Yan, Q. 2017. Axiomatic attribution for deep networks. In *ICML*. 4

Tan, M.; and Le, Q. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*. 1, 4

Touvron, H.; Cord, M.; Douze, M.; Massa, F.; Sablayrolles, A.; and Jégou, H. 2020. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*. 2, 3, 4, 6, 7, 10, 12

Touvron, H.; Cord, M.; Sablayrolles, A.; Synnaeve, G.; and Jégou, H. 2021. Going deeper with image transformers. *arXiv preprint arXiv:2103.17239*. 2

Vaswani, A.; Ramachandran, P.; Srinivas, A.; Parmar, N.; Hechtman, B.; and Shlens, J. 2021. Scaling Local Self-Attention For Parameter Efficient Visual Backbones. *CVPR*. 1, 2, 3, 7, 13

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. *NeurIPS*. 3

Wang, H.; Zhu, Y.; Adam, H.; Yuille, A.; and Chen, L.-C. 2021a. MaX-DeepLab: End-to-End Panoptic Segmentation with Mask Transformers. *CVPR*. 2

Wang, W.; Xie, E.; Li, X.; Fan, D.-P.; Song, K.; Liang, D.; Lu, T.; Luo, P.; and Shao, L. 2021b. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. *arXiv preprint arXiv:2102.12122*. 2, 3, 5, 12

Wu, H.; Xiao, B.; Codella, N.; Liu, M.; Dai, X.; Yuan, L.; and Zhang, L. 2021. Cvt: Introducing convolutions to vision transformers. *arXiv preprint arXiv:2103.15808*. 3

Yuan, L.; Chen, Y.; Wang, T.; Yu, W.; Shi, Y.; Jiang, Z.; Tay, F. E.; Feng, J.; and Yan, S. 2021. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986*. 2, 3

Yun, S.; Han, D.; Oh, S. J.; Chun, S.; Choe, J.; and Yoo, Y. 2019. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*. 10

Zagoruyko, S.; and Komodakis, N. 2016. Wide residual networks. *BMVC*. 12

Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2018a. mixup: Beyond empirical risk minimization. *ICLR*. 10

Zhang, H.; Goodfellow, I.; Metaxas, D.; and Odena, A. 2019. Self-attention generative adversarial networks. In *ICML*. 1, 4, 6, 7

Zhang, H.; Koh, J. Y.; Baldridge, J.; Lee, H.; and Yang, Y. 2021a. Cross-Modal Contrastive Learning for Text-to-Image Generation. In *CVPR*. 11

Zhang, H.; Zhang, Z.; Odena, A.; and Lee, H. 2020. Consistency regularization for generative adversarial networks. *ICLR*. 11

Zhang, P.; Dai, X.; Yang, J.; Xiao, B.; Yuan, L.; Zhang, L.; and Gao, J. 2021b. Multi-Scale Vision Longformer: A New Vision Transformer for High-Resolution Image Encoding. *arXiv preprint arXiv:2103.15358*. 2, 3

Zhang, X.; Wei, F.; and Zhou, M. 2019. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. *arXiv preprint arXiv:1905.06566*. 2

Zhang, X.; Wei, Y.; Feng, J.; Yang, Y.; and Huang, T. S. 2018b. Adversarial complementary learning for weakly supervised object localization. In *CVPR*. 6

Zhong, Z.; Zheng, L.; Kang, G.; Li, S.; and Yang, Y. 2020. Random erasing data augmentation. In *AAAI*. 10

Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; and Torralba, A. 2016. Learning deep features for discriminative localization. In *CVPR*. 5, 12

Appendix

We provide more experimental results below to complete the experimental sections of the main paper.

NesT Architecture and Training Details

Architecture configuration. The focus on this paper is how to aggregating nested transformers and its extended usage. We do not focus on the specific per-block hyper-parameters (e.g. number of heads and number of MSA layers). We mainly follow previous work to obtain right architectures that has similar capacity (e.g. number of parameters and throughput).

Recall that the overall hierarchy can be determined by two key hyper-parameters: patch size $S \times S$ and hierarchy depth T_d. Just like how ResNet (He et al. 2016) adapts to small and large input sizes, NesT also has different configuration for small input size and large input size. We follow (Touvron et al. 2020; Liu et al. 2021) to configure the number of head, hidden dimensions for the tiny, small and base versions. For 32×32 image size, we follow (Touvron et al. 2020). Specifically, we setup the same number of repeated MSA_{NesT} per block hierarchy. In each hierarchy, the number of hidden dimensions and the number of heads are the same as well. For 224×224 image size, we follow (Liu et al. 2021). Therefore, different hierarchy has a gradually increased number of head, hidden dimensions, and number of repeated MSA_{NesT} layers. Table A1 specifies details.

NesT Hierarchy Variants

We study flexible variants to understand how the hierarchical structure of NesT impacts accuracy. When increasing the hierarchy depth by one, every four blocks are splitted to process four image partitions (see Figure 1 of the main paper). A deeper hierarchy structure makes each block focus on a narrower range of pixel information (i.e., shorter sequence length).

We test combinations with $S = \{1, 2\}$ and depth=$\{2, 3, 4, 5\}$ on NesT-{T, S, B}. Figure A1 compares different variants on two CIFAR datasets. Shallower NesT has clear accuracy drop (although the total number of self-attention layers are the same). It is because, when depth$= 1$, the model degenerates to a global self-attention method, such as ViT (Dosovitskiy et al. 2021). Depth$= 5$ has marginal decrease, because the sequence length of all blocks is only $n = 2 \times 2$. Note that we denote NesT_4-B, $S = 1$ as the base NesT with hierarchy depth 4 and word patch size 1×1. We use the configuration $\text{NesT}_4, S = 1$ as the default for the most CIFAR experiments (subscription is omitted sometimes).

Data Augmentation. We apply the commonly used data augmentation and regularization techniques as (Touvron et al. 2020), which include a mixture of data augmentation (MixUp (Zhang et al. 2018a), CutMix (Yun et al. 2019), RandAugment (Cubuk et al. 2020), RandomErasing (Zhong et al. 2020)) and regularization like Stochastic Depth (Huang et al. 2016). Repeated augmentation (Hoffer et al. 2020) in DeiT is not used. In addition, for ImageNet models, we also add color jittering similar to (Chen et al. 2020a,b) which seems to reduce dependency on local texture cues and slightly improves generalization ($\sim 0.3\%$ on ImageNet).

Training details. We use a base learning rate 2.5e-6 per device. We use the AdamW optimizer (Loshchilov and Hutter 2018) and set the weight decay 0.05. The warm-up epoch is 20. The initial learning rate is linearly scaled by a factor of $total\_batch\_size/256$. For ImageNet training, the total batch size can be 1024 or 2048 when using distributed training on the TPU hardware. We use $[0.2, 0.3, 0.5]$ stochastic death rates for NesT-T, NesT-S, and NesT-B models, respectively. All transformer layer weights and block aggregation weights are initialized using truncated normal distribution.

We use a 0.1 stochastic depth drop rate for all CIFAR models and the warmup is 5 epoch. The CIFAR results of compared transformer based methods, besides CCT-7/3$\times$1 (Hassani et al. 2021), in Table 1 of the main paper are trained by us. We train these models using their suggested hyper-parameters and we find it works nearly optimal on CIFAR by searching from a set of learning rate and weight decay combinations.

Impact of Block Aggregation

Figure A3 shows detailed studies of different block aggregation functions to complete results in Figure 7 of the main paper. Although many of them has tiny difference, it is interesting that the impact to performance is non-trivial. In addition, we find perform query down-sampling inside self attention makes transformers more difficult to train because the skip connection also needs proper down-sampling.

Ablation Studies

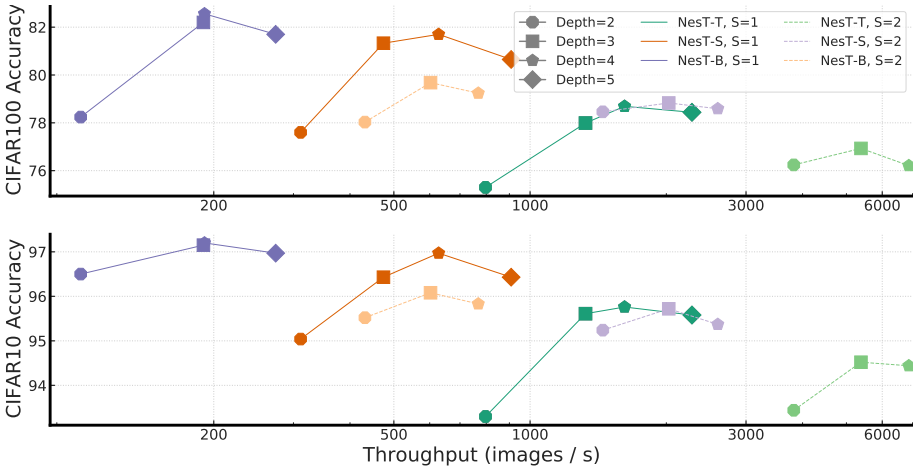
Data augmentation sensitivity. NesT uses several kinds of data augmentation types following DeiT (Touvron et al. 2020). As shown in Figure A4, our method shows the high stability in augmentation ablation studies compared with DeiT. We speculate the underline reason is that learning effective vision cues is much easier in local attention than in global attention, so Swin Transformer also shows comparable stability. More comparison on ImageNet will be left as future work.

Weak teacher distillation. We also explore the teacher distillation proposed by (Touvron et al. 2020), which suggests the inductive bias introduced by convnet teachers are helpful for ViT data efficiency. Table A4 provides detailed distillation study on CIFAR datasets. With such a weak teacher distillation, NesT-B is able to achieve 84.9% CIFAR100 accuracy with 300 epochs and even 86.1% with 1000 epoch training using 2 GPUs.

Convnet teacher distillation (Touvron et al. 2020) is effective to further improve our method as well as DeiT. As explained in (Touvron et al. 2020), the inductive biases of convnets have positive implicit effects to the transformer training. Because data augmentation can improve teacher performance, we question the inductive biases brought by data augmentation is useful or not. Based on our experiments, it seems data augmentation negatively affect the effectiveness of teacher distillation. If the teacher and target model are both trained with strong augmentation, the performance decreases either for a small teacher or a big teacher. In other words, our study suggests that training a high accuracy teacher using strong augmentation negatively impact the distillation

Table A1: Architecture details of NesT. In each block, the structure is defined using the protocol $[d, h] \times a, b$, where $[d, h]$ refers to [hidden dimensions, number of heads]; a refers to the number of repeated transformer layers V in Equation 1 of the main paper; b refers to the total number of blocks in that hierarchy. Tiny, Small, and Base models have different setup and they are specified below. Note that once the hierarchy is fixed. The sequence length of all blocks are consistent. Configurations of each block for CIFAR and ImageNet are different.

Input size	Seq. length	NesT Hierarchy (Froward direction is 5 to 1)				
		1	2	3	4	5
32×32 $S = 1$	$d = [192, 384, 768]$ and $h = [3, 6, 12]$ for model T, S, and B					
	8×8	$[d, h] \times 4, 1$	$[d, h] \times 4, 4$	$[d, h] \times 4, 16$	-	-
	4×4	$[d, h] \times 3, 1$	$[d, h] \times 3, 4$	$[d, h] \times 3, 16$	$[d, h] \times 3, 64$	-
	2×2	$[d, h] \times 2, 1$	$[d, h] \times 2, 4$	$[d, h] \times 2, 16$	$[d, h] \times 2, 64$	$[d, h] \times 2, 256$
224×224 $S = 4$	$d = [96, 96, 128]$, $h = [3, 6, 12]$, and $k = [8, 20, 20]$ for model T, S, and B					
	14×14	$[d, h] \times 2, 1$	$[2d, 2h] \times 2, 4$	$[4d, 4h] \times k, 16$	-	-



Depth	Node seq. length n	
	$S = 1$	$S = 2$
2	256	64
3	64	16
4	16	4
5	4	-

Figure A1: Comparison of NesT hierarchy variants with different depth, word size $S \times S$, and model size. The right table specifies the resulting sequence length given hierarchy depth and S combinations.

effectiveness. Future verification on ImageNet will be left for future work.

Number of heads. We realize different architecture design uses different number of heads for MSA. We attempt to understand the effectiveness of the different configurations. We experiment number of head from 1 to 96 given a fixed $d = 768$ hidden dimension using NesT_4-B. Table A5 shows CIFAR10 results on NesT. It is interesting find the number of heads affects less to the final performance.

Generative Modeling

NesT can become a decoder with minor changes. For fair comparison in terms of model capacity, we configure NesT following the architecture design of TransGAN for image generation. Table A6 specifies the architectural details. The block aggregation layer is swapped to a block de-aggregation layer to achieve the gradually increased image size. Pixel shuffle (PS) (Shi et al. 2016) is leveraged to increase the image size at block de-aggregation by a factor of two while the hidden dimension is reduced to a quarter of the input.

We adopt the same discriminator architecture as (Karras et al. 2020) where R1 gradient penalty (Mescheder, Geiger, and Nowozin 2018) is applied during training. Adam (Kingma and Ba 2014) is utilized for optimization with $\beta_1 = 0$ and $\beta_2 = 0.99$. The learning rate is 0.0001 for both the generator and discriminator with mini-batches of size 256. We use Fréchet Inception Distance (FID) (Heusel et al. 2017) for assessing image quality, which has been shown to be consistent with human judgments of realism (Heusel et al. 2017; Zhang et al. 2020, 2021a). Lower FID values indicate closer distances between synthetic and real data distributions. Figure A6 shows the inception score of different compared methods on 64×64 image generation.

Interpretabilty

GradGAT results. GradGAT always traverses from the root node to one of the leaf node. Since each node of the bottom layer only corresponds to a small non-overlapping patch of

Table A2: Test accuracy on CIFAR with input size 32×32. Compared convnets are optimized models for CIFAR. All transformer based models are trained from random initialization with the same data augmentation. The number of parameters (millions), GPU memory (MB), and inference throughput (images/s) on single GPU are compared. We minimize the word size $S \times S$ for each transformer based method. DeiT uses $S = 2$. Swin and our NesT uses $S = 1$. * means models tends to diverge.

Arch. base	Method	#Params	GPU	Throughput	C10 (%)	C100 (%)
Convnet	Pyramid-164-48 (Han, Kim, and Kim 2017)	1.7M	126M	3715.9	95.97	80.70
	WRN28-10 (Zagoruyko and Komodakis 2016)	36.5M	202M	1510.8	95.83	80.75
Transformer full-attention	DeiT-T (Touvron et al. 2020)	5.3M	158M	1905.3	88.39	67.52
	DeiT-S (Touvron et al. 2020)	21.3M	356M	734.7	92.44	69.78
	DeiT-B (Touvron et al. 2020)	85.1M	873M	233.7	92.41	70.49
	PVT-T (Wang et al. 2021b)	12.8M	266M	1478.1	90.51	69.62
	PVT-S (Wang et al. 2021b)	24.1M	477M	707.2	92.34	69.79
	PVT-B (Wang et al. 2021b)	60.9M	990M	315.1	$85.05^\star$	$43.78^\star$
	CCT-7/3×1 (Hassani et al. 2021)	3.7M	94M	3040.2	94.72	76.67
Transformer local-attention	Swin-T (Liu et al. 2021)	27.5M	183M	2399.2	94.46	78.07
	Swin-S (Liu et al. 2021)	48.8M	311M	1372.5	94.17	77.01
	Swin-B (Liu et al. 2021)	86.7M	497M	868.3	94.55	78.45
	NesT-T	6.2M	187M	1616.9	96.04	78.69
	NesT-S	23.4M	411M	627.9	96.97	81.70
	NesT-B	90.1M	984M	189.8	**97.20**	**82.56**

Table A3: Comparison on the ImageNet benchmark. The number of parameters (millions), GFLOPS, and inference throughput (images/s) evaluated on a single GPU are also compared. All models are trained from random initialization without extra pre-training.

Arch. base	Method	Size	#Params	GFLOPS	Throughput	Top-1 acc. (%)
Convnet	ResNet-50 (He et al. 2016)	224	25M	3.9G	1226.1	76.2
	RegNetY-4G (Radosavovic et al. 2020)	224	21M	4.0G	1156.7	80.0
	RegNetY-16G (Radosavovic et al. 2020)	224	84M	16.0G	334.7	82.9
Transformer full-attention	ViT-B/16 (Dosovitskiy et al. 2021)	384	86M	55.4G	85.9	77.9
	DeiT-S (Touvron et al. 2020)	224	22M	4.6G	940.4	79.8
	DeiT-B (Touvron et al. 2020)	224	86M	17.5G	292.3	81.8
Transformer local-attention	Swin-T (Liu et al. 2021)	224	29M	4.5G	755.2	81.3
	Swin-S (Liu et al. 2021)	224	50M	8.7G	436.9	83.0
	Swin-B (Liu et al. 2021)	224	88M	15.4G	278.1	83.3
	NesT-T	224	17M	5.8G	633.9	81.5
	NesT-S	224	38M	10.4G	374.5	83.3
	NesT-B	224	68M	17.9G	235.8	**83.8**

the whole image, visualizing GradCAT is less meaningful when the targeting object is large and centered. We find it is true for the majority of ImageNet images although we find our results fairly promising for most ImageNet images that have small objects. Exploring more comprehensive studies on image datasets with non-centered objects is be left for future work.

The proposed GradCAT is partially inspired by how Grad-CAM (Selvaraju et al. 2017) in convnets uses gradient information to improve visual attention. Nevertheless, the actual detailed design and serving purposes are distinct.

Class attention map results. Figure 4 of the main paper compares the qualitative results of CAM, including Grad-CAM++ (Chattopadhay et al. 2018) with ResNet50 (He et al. 2016), DeiT with Rollout attention (Abnar and Zuidema 2020), and our NesT CAM (Zhou et al. 2016). We follow (Gildenblat 2021) to use an improved version of Rollout, which is better than the original version. When converting CAM generated by different methods to bounding boxes, the best threshold of each method varies. We search the best threshold [0, 1] using 0.05 as the interval to find the best number for each method on the ImageNet 50k validation set. It is promising to find that NesT CAM can outperform methods for this task and our baselines. We only use the single forward to obtain bounding boxes.

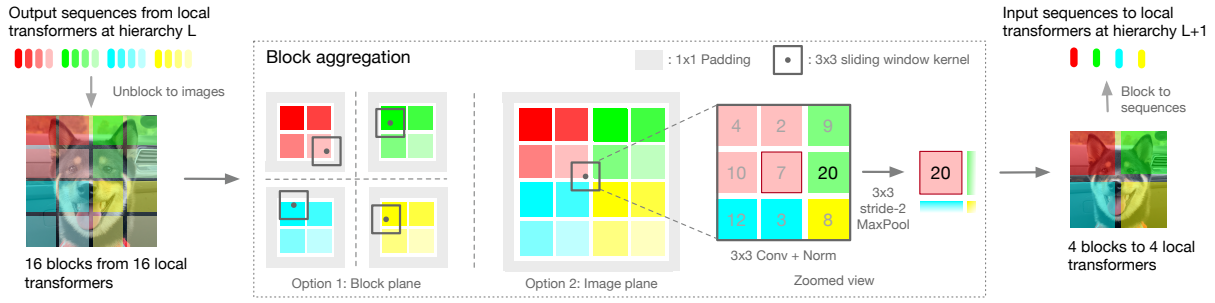


Figure A2: Illustration of block aggregation and a comparison when applying to the block plane versus on the image plane. Although both perform convolution and pooling spatially, performing block aggregation on the image plane allows information communication among blocks (different color palettes) that belong to different merged blocks at the upper hierarchy.

Table A4: Teacher distillation studies on CIFAR datasets. Left: The top two rows of the table are teacher supervised accuracy on CIFAR100. The bottom two rows show accuracy using these trained teachers with standard or strong augmentation. Right: Teacher (PN-164-48 with standard augmentation) distillation effects on DeiT and the proposed NesT. DeiT and NesT are always trained with strong augmentation.

Teacher	Target model	Standard	Strong
-	PN-164-48	80.7	81.5
-	PN-164-270	83.4	84.9
PN-164-48	NesT-B	84.5	83.7
PN-164-270	NesT-B	**84.9**	83.8

Distillation	✗		✓	
Dataset	C10	C100	C10	C100
DeiT-B	92.4	70.5	95.5	81.5
NesT-B	**97.2**	82.6	97.1	**84.5**

Notation of spatial operations
C3: 3 × 3 Conv
LN: LayerNorm
MX3: 3 × 3 MaxPool
S2: 2 × 2 sub-sampling
AVG3: 3 × 3 AvgPool
D4: 4 × 1 1D Conv
PM: Patch merge

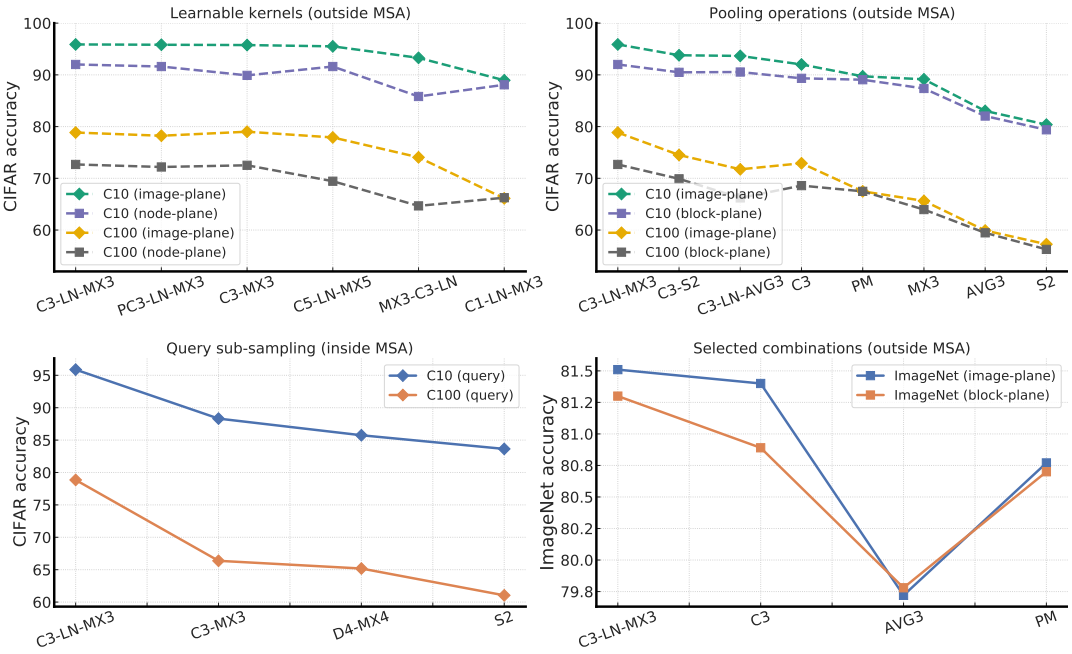


Figure A3: Study the impact of block aggregation on CIFAR and ImageNet. NesT-T is used. We study from different perspectives as explained in the text of the main paper. We verify ImageNet with NesT-T in the bottom-right figure using a subset of representative block aggregation options found on CIFAR datasets. Patch merge (Liu et al. 2021) and 2x2 sub-sampling (Vaswani et al. 2021) are used by previous methods. Since NesT for ImageNet has different hidden dimensions at different hierarchies, AVG3 on ImageNet is followed by a 1x1 convolution to map hidden dimensions. The chosen combinations are specified in x-axis. The leftmost x-axis point (C3-LN-MX3) of each figure is ours.

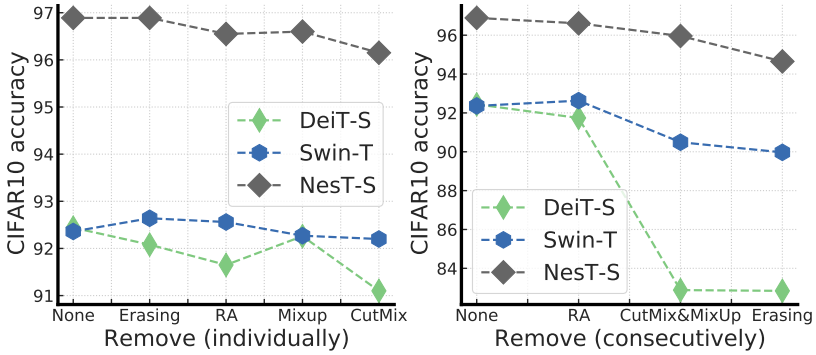


Figure A4: Data augmentation ablation studies on CIFAR10, either removing augmentation individually (middle) or removing (from left to right of x-axis) consecutively (right). None means all are used.

Table A5: Study the impact of number of heads in MSA on the CIFAR10 dataset with NesT_4-B. When #head=96, the hidden dimension used for computing Attention is only 8. However, it can still lead to similar accuracy.

#head in MSA	1	2	3	6	12	24	48	96
Hidden dimension d	768	384	256	128	64	32	16	8
Accuracy	97.1	96.85	96.92	97.07	**97.21**	97.01	97.03	97.08

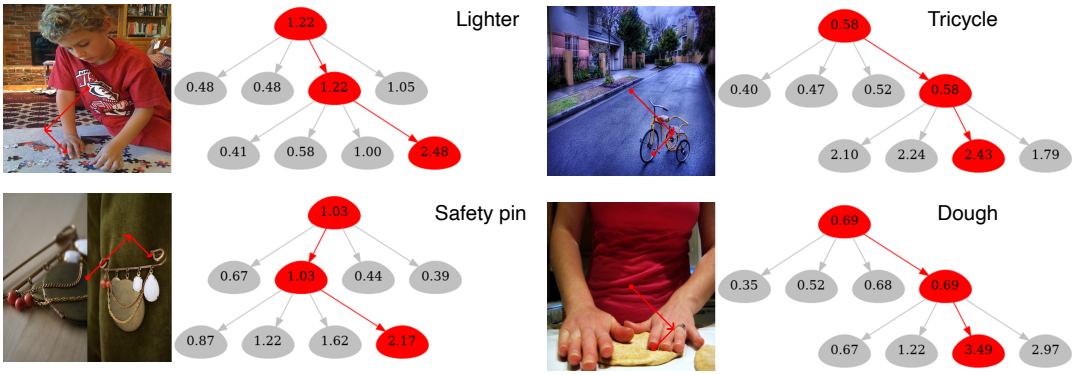


Figure A5: More output visualization of the proposed GradGAT.

Table A6: Architecture details of NesT as image generator. $d = 1024$ and $h = 4$. The input is a reshaped noise vector. At the last hierarchy, there are 64 image blocks. Since the sequence length is 8×8, it is easy to see that the output image size is 64×64. At hierarchy 1, the hidden dimension is $1024/64 = 16$. Then a LayerNorm followed by Conv1x1 maps the hidden dimension to the output with shape $64 \times 64 \times 3$.

Input size	Seq. length	NesT Hierarchy (Froward direction is 4 to 1)			
		1	2	3	4
$1 \times n \times d$	8×8	$[d/64, h] \times 2, 64$	$[d/16, h] \times 3, 16$	$[d/4, h] \times 3, 4$	$[d, h] \times 5, 1$

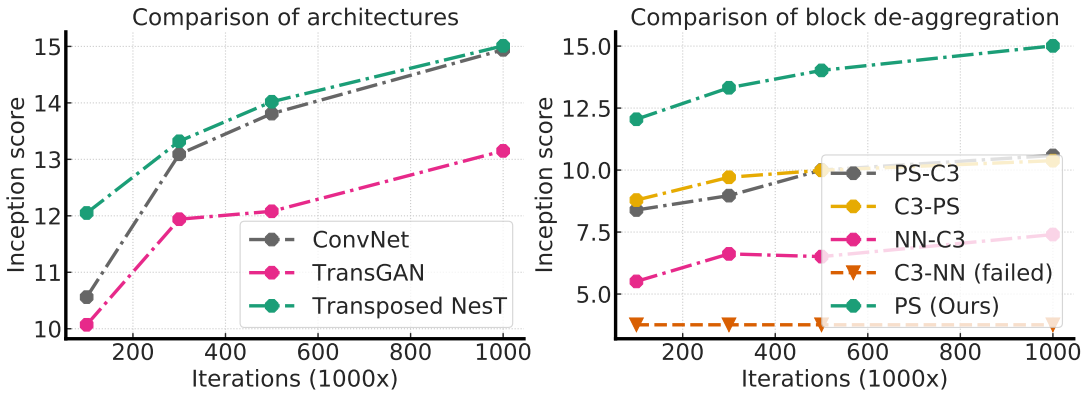


Figure A6: Left: Inception score of 64×64 ImageNet generation of different architectures. Right: Inception score with different un-pooling options. The models used to report the results are the same models in Figure 5 of the main paper.