DenseNets Reloaded: Paradigm Shift Beyond ResNets and ViTs – Appendix –

Donghyun Kim[1⋆], Byeongho Heo[2], and Dongyoon Han[2⋆]

[1]NAVER Cloud AI, [2]NAVER AI Lab

In this Appendix, we provide additional experiments and details to complement the main paper. The contents are as follows:

- §A presents our experimental setups for ImageNet and downstream tasks training and evaluation setups;
- §B presents detailed layer configuration of RDNet-T;
- §C evaluates further ImageNet top-accuracy versus latency trade-offs on various testbeds, encompassing both PyTorch and TensorRT A100 inference, as well as CPU inference outcomes;
- §D conducts an ablation study by aligning the depth, parameters, and FLOPs with ResNet and ConvNeXt to reaffirm the strengths of dense connections;
- §E reports further COCO [23] object detection and instance segmentation results with Mask-RCNN [14] and Cascade Mask-RCNN [5];
- §F benchmarks the transferability of ImageNet-1K pre-trained models. We utilize fine-grained classification datasets and long-tailed classification datasets, including CIFAR-10 [21], CIFAR-100 [21], Flowers-102 [29], Stanford-Cars [20], iNaturalist-2018 [17], and iNaturalist-2019 [18];
- §G evaluates the effectiveness of dense connections on generative models;
- §H benchmarks robustness of the ImageNet-1K [34] pre-trained models on the out-of-distribution datasets, including ImageNet-V2 [33], ObjectNet [4], ImageNet-A [16], ImageNet-Sketch [41], and ImageNet-R [15];
- §I gives more details of our pilot study described in §5.1 with overall results, and scaled-up experiments on ImageNet-1K.

⋆ Equal contribution. Correspondence to Dongyoon Han.

Table A: ImageNet-1K training settings. Most training setups are consistently used except for the multiple stochastic depth rates (e.g., 0.15/0.35/0.4/0.45) that regularize the corresponding models (e.g., RDNet-T/S/B/L), respectively.

	RDNet-T/S/B/L (Pre-)Training	RDNet-L Fine-Tuning
image size	224	384
weight init	kaiming normal	pre-trained
optimizer	AdamW	AdamW
base learning rate	1e-3	2e-5
weight decay	0.05	1e-8
optimizer momentum (β_1, β_2)	0.9, 0.999	0.9, 0.999
batch size	512	512
training epochs	300	30
learning rate schedule	cosine decay	cosine decay
warmup epochs	20	5
warmup schedule	linear	linear
layer-wise lr decay [3, 7]	None	0.7
randaugment [8]	(9, 0.5)	(9, 0.5)
mixup [48]	0.8	0.0
cutmix [47]	1.0	0.0
random erasing [49]	0.25	0.25
label smoothing [36]	0.1	0.1
stochastic depth [19]	0.15/0.35/0.4/0.5	0.6
layer scale [39]	1e-6	pre-trained
head init scale [39]	None	1e-3
gradient clip	None	None
center crop percent	0.9	1.0
exp. mov. avg. (EMA) [30]	None	None

A Experimental Settings

A.1 ImageNet Training

Table A presents the training settings for RDNet on ImageNet-1K. Each variant of RDNet adheres to these settings, except for the stochastic depth rate [19], which is tailored to each model variant. For fine-tuning, we group three consecutive feature mixer blocks for layer-wise learning rate decay [3, 7] akin to the approach taken in ConvNeXt. We use the timm package [44] for model training.

A.2 Downstream Tasks

We adhere to the hyper-parameter sweep protocol outlined in [27] but sweep much lightly. For UperNet [45] training on ADE20K, we explore the following hyperparameters: learning rate {8e-5, 1e-3}, weight decay {0.01, 0.03, 0.05}. For Mask-RCNN [14] training on COCO, we explore hyperparameters such as learning rate {1e-3, 2e-3, 3e-3}, weight decay {0.05, 0.1}, stochastic depth {0.1, 0.2}. For Cascade Mask-RCNN [5] training on COCO, we explore hyperparameters such as learning rate {8e-5, 1e-4}, weight decay {0.05, 0.1}, stochastic depth {0.4, 0.5, 0.6}, and layer-wise learning rate decay {0.7, 0.8}. We note that our search space is similar to or less extensive than the known search space in ConvNeXt [27] (*e.g.*, 6 (ours) vs. 12 (ConvNeXt) for ADE20K (UperNet) and 8/24 (ours) vs. 48 (ConvNeXt) for COCO (Cascade Mask-RCNN) experiments).

A.3 Benchmark Setup

We measure latency and memory on the V100 GPU utilizing PyTorch 1.13.1 and CUDA 11.6. In all measurements, we employ the channels-last memory format [28]. Memory is measured in the training phase with a batch size of 16.

B Model Configurations of RDNet

From a practitioner's perspective, it can be challenging to ascertain the number of channels of features of RDNet. Therefore, we provide a detailed model configuration in Table B. The values of resolution and channel are based on the output feature. We ensured that the output dimensions of the transition layers are multiples of 8 for efficiency.

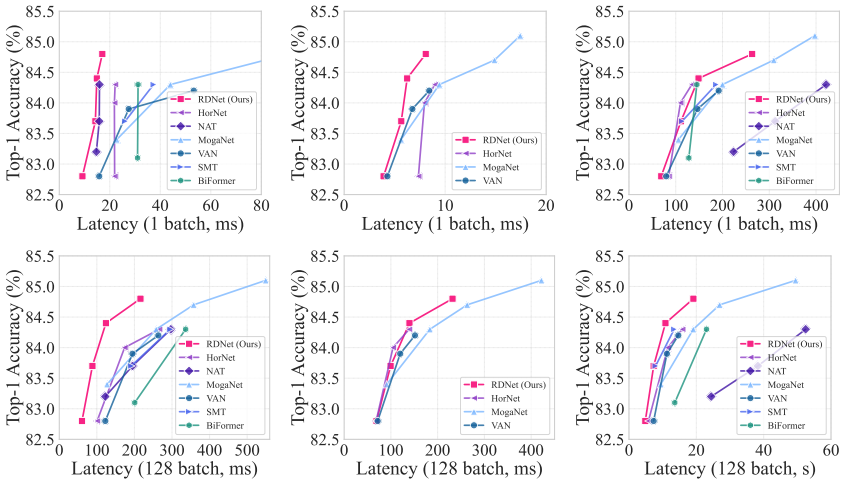
C More ImageNet Accuracy vs. Latency Trade-offs

To assess the practicality of our models, we measure speeds across diverse testbeds. We precisely measure inference speeds using the PyTorch framework on NVIDIA A100 GPU, Intel Xeon Gold 5120 CPU, and TensorRT Inference Engine on

Table B: Model configurations of RDNet. The table on the left presents the layer-wise configuration of RDNet-T. The table on the right details the configuration of the RDNet model family. As described in Fig. 1, each stage of the RDNet comprises L_N mixing blocks.

Stage	GR	Layer	Resolution	#Channels
		Patchification	56×56	64
S1	64	Feature mixer	56×56	128
		Feature mixer	56×56	192
		Feature mixer	56×56	256
		Transition S2	28×28	128
S2	104	Feature mixer	28×28	232
		Feature mixer	28×28	336
		Feature mixer	28×28	440
		Transition S2	14×14	216
S3	128	Feature mixer	14×14	344
		Feature mixer	14×14	472
		Feature mixer	14×14	600
		Transition S1	14×14	296
		Feature mixer	14×14	424
		Feature mixer	14×14	552
		Feature mixer	14×14	680
		Transition S1	14×14	336
		Feature mixer	14×14	464
		Feature mixer	14×14	592
		Feature mixer	14×14	720
		Transition S1	14×14	360
		Feature mixer	14×14	488
		Feature mixer	14×14	616
		Feature mixer	14×14	744
		Transition S2	7×7	368
S4	224	Feature mixer	7×7	592
		Feature mixer	7×7	816
		Feature mixer	7×7	1040
		Classifier	1×1	1000

Stage	Layer Settings	RDNet			
		Tiny	Small	Base	Large
S1	Growth Rate	64	64	96	128
	# Mixing Block	1	1	1	1
Transition S/2					
S2	Growth Rate	104	128	128	192
	# Mixing Block	1	1	1	1
Transition S/2					
S3	Growth Rate	128	128	168	256
	# Mixing Block	4	7	7	8
Transition S/2					
S4	Growth Rate	192	240	336	360
	# Mixing Block	1	2	2	2
Classifier		GAP, Linear			
Parameters (M)		24	50	87	186
FLOPs (G)		5.0	8.7	15.4	34.7



(a) PyTorch (A100 GPU) **(b)** TensorRT (A100 GPU) **(c)** CPU (Xeon Gold 5120)

Fig. A: Further trade-offs in ImageNet-1K performance. We provide comparative visualizations with diverse environments (including A100, TesorRT, and CPU) between *state-of-the-art models*, which were known for top-performing models. It turns out that RDNet is highly competitive in practice in terms of model speed.

NVIDIA A100 GPU. Our testing environment incorporates PyTorch version 1.13.1, CUDA version 11.6, and TensorRT version 8.5.3 for these experiments. Fig. A shows that our models consistently show superior accuracy vs. latency trade-offs across all evaluation setups. Note that (b) in Fig. A includes fewer models because those that are challenging to convert to inference engines due to factors like the use of CUDA custom kernels are not evaluated. We report all the numbers in Table C.

D More Comparison with ResNets and ConvNeXts

We presented the performance of RDNets matched with ResNets and ConvNeXts in terms of the number of parameters and FLOPs in Table 8a. Here, we further report some RDNets' performances, which are more closely aligned with ResNets and ConvNeXts than our previous models by 1) having identical depth; 2) having the same number of building blocks in each stage; 3) aligning the parameters and FLOPs as closely as possible with those of ResNet and ConvNext. Therefore, the comparison is more like an apples-to-apples comparison. As shown in Table D, despite RDNet not having the optimal width and depth for dense connections, RDNet demonstrates superior performance compared to the competing models.

Table C: ImageNet-1K comparison with the latest models. Fig. A visualized this table. We thoroughly compare our models against the latest architectures in practical latencies. bn denotes latency, measured with a batch size of n. Certain models were excluded from evaluation because their CUDA custom kernels were not compiled.

Model	Date	Param (M)	FLOPs (G)	Top-1 (%)	PyTorch (A100, ms) b1	b8	b32	b128	TensorRT (A100, ms) b1	b8	b32	b128	PyTorch (Xeon 5120, s) b1	b8	b32	b128
RDNet-T	Ours	24	5.0	82.8	9.2	9.2	17.8	60.5	3.9	7.5	19.7	69.3	0.07	0.25	1.09	4.85
HorNet-T$_{7\times7}$ [32]	NeurIPS'2022	22	4.0	82.8	21.9	21.9	27.8	100.7	7.4	10.6	22.8	68.4	0.09	0.21	0.89	5.79
VAN-B2 [11]	CVMJ'2023	27	5.0	82.8	15.8	16.4	32.9	122.5	4.1	8.6	21.2	71.8	0.08	0.38	1.61	7.33
BiFormer-S [50]	CVPR'2023	26	4.5	83.8	31.0	35.3	54.2	200.9	-	-	-	-	0.13	0.39	2.17	13.60
NAT-T [12]	CVPR'2023	28	4.3	83.2	14.7	14.7	32.9	122.4	-	-	-	-	0.22	1.43	5.73	24.42
SMT-S [24]	ICCV'2023	21	4.7	83.7	26.1	26.4	53.0	191.4	-	-	-	-	0.11	0.32	1.16	7.91
MogaNet-S [22]	ICLR'2024	25	5.0	83.4	22.7	22.7	34.9	127.2	5.6	10.3	27.6	91.0	0.11	0.42	1.97	9.46
RDNet-S	Ours	50	8.7	83.7	14.3	14.4	26.4	88.3	5.7	10.8	29.3	99.4	0.11	0.38	1.73	7.34
HorNet-S$_{7\times7}$ [32]	NeurIPS'2022	50	8.8	84.0	21.8	22.2	46.9	173.9	8.0	14.1	33.6	104.5	0.11	0.38	2.43	11.51
VAN-B3 [11]	CVMJ'2023	45	9.0	83.9	27.5	32.8	50.4	194.8	6.8	13.7	34.5	119.9	0.15	0.59	2.50	11.27
BiFormer-B [50]	CVPR'2023	57	9.8	84.3	31.2	31.1	89.3	336.3	-	-	-	-	0.15	0.78	4.90	23.03
NAT-S [12]	CVPR'2023	51	7.8	83.7	15.8	20.6	51.9	194.9	-	-	-	-	0.31	2.10	8.78	38.27
SMT-B [24]	ICCV'2023	32	7.7	84.3	37.4	39.1	81.2	295.6	-	-	-	-	0.19	0.60	2.24	13.41
MogaNet-B [22]	ICLR'2024	44	9.9	84.3	44.0	47.4	70.6	258.0	9.4	19.0	53.4	183.4	0.20	0.77	4.08	19.10
RDNet-B	Ours	87	15.4	84.4	14.9	15.1	36.0	124.2	6.2	13.6	39.5	139.8	0.15	0.58	2.52	10.84
HorNet-B$_{7\times7}$ [32]	NeurIPS'2022	87	15.6	84.3	22.1	22.3	68.9	266.1	9.0	16.5	41.8	139.1	0.13	0.57	3.43	15.86
VAN-B4 [11]	CVMJ'2023	60	12.2	84.2	53.3	53.6	80.7	263.6	8.4	17.0	45.1	151.6	0.19	0.70	3.26	14.65
NAT-B [12]	CVPR'2023	90	13.7	84.3	15.9	22.5	88.6	296.9	-	-	-	-	0.42	3.03	12.33	52.47
MogaNet-L [22]	ICLR'2024	83	15.9	84.7	81.6	90.3	98.8	357.7	14.9	28.7	77.4	263.2	0.31	1.08	51.56	26.93
RDNet-L	Ours	186	34.7	84.8	17.0	17.3	59.8	216.1	8.1	20.4	62.9	231.8	0.26	1.15	4.72	19.13
MogaNet-XL [22]	ICLR'2024	181	34.5	85.1	82.3	93.3	146.3	549.5	17.4	38.6	115.4	421.9	0.40	2.57	10.27	49.49

Table D: Apples-to-apples comparison with ResNet and ConvNeXt. We align the parameters, FLOPs, and depth of ResNet and ConvNext as closely as possible to compare their top-1 accuracy (%).

Model	Params	FLOPs	Depth	Top-1
ResNet-50	25.6M	4.1G	[3,4,6,3]	78.8
RDNet	25.5M	4.1G	[3,4,6,3]	**82.1**
ResNet-152	60.2M	11.5G	[3,8,36,3]	80.8
RDNet	59.7M	11.5G	[3,8,36,3]	**83.7**
ConvNeXt-T	28.6M	4.5G	[3,3,9,3]	82.1
RDNet	27.1M	4.5G	[3,3,9,3]	**82.4**
ConvNeXt-B	88.6M	15.4G	[3,3,27,3]	83.8
RDNet	88.3M	15.3G	[3,3,27,3]	**84.1**

E More Object Detection/Instance Segmentation results

An extension to the Mask-RCNN [14] with 3x schedule [13] results in Table 6 in the main paper, we additionally train Mask-RCNN using a 1x schedule [13] to perform a more comprehensive and fair comparison with a broader range of models. As shown in Table E, pre-trained RDNet models demonstrate competitive performance. Furthermore, we employ the Cascade Mask-RCNN head [5] to evaluate our pre-trained models. As demonstrated in Table F, RDNet exhibits competitive performance. Note that our models do not experience exhaustive

Table E: COCO object detection and segmentation results - Mask-RCNN 1x schedule. FLOPs (G) are calculated with image size (1280, 800).

Backbone	Param	FLOPs	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
PVT-S [42]	44M	245G	40.4	62.9	43.8	37.8	60.1	40.3
Swin-T [26]	48M	264G	43.6	66.2	47.7	39.6	62.9	42.2
PVTv2-B2 [43]	45M	309G	45.3	67.1	49.6	41.2	64.2	44.4
ConvNeXt-T [27]	48M	262G	43.5	65.6	48.0	39.7	62.5	42.7
CSwin-T [9]	42M	279G	46.7	68.6	51.3	42.2	65.6	45.4
FocalNet-S (SRF) [46]	49M	267G	45.9	68.3	50.1	41.3	65.0	44.3
RDNet-T	43M	278G	46.1	68.0	50.8	41.4	65.1	44.2
PVT-M [42]	64M	302G	42.0	64.4	45.6	39.0	61.6	42.1
Swin-S [26]	69M	354G	46.5	68.7	51.3	42.1	65.8	45.2
PVTv2-B3 [43]	65M	397G	47.0	68.1	51.7	42.5	65.7	45.7
ConvNeXt-S [27]	70M	348G	46.8	69.0	51.5	42.1	65.8	45.2
CSwin-S [9]	54M	342G	47.9	70.1	52.6	43.2	67.1	46.2
FocalNet-S (SRF) [46]	71M	356G	48.0	69.9	52.7	42.7	66.7	45.7
RDNet-S	70M	354G	48.2	69.9	53.0	43.0	66.9	46.3
Swin-B [26]	107M	496G	46.9	69.2	51.6	42.3	66.0	45.5
PVTv2-B5 [43]	102M	557G	47.4	68.6	51.9	42.5	65.7	46.0
ConvNeXt-B [27]	108M	486G	47.5	69.9	51.9	42.5	66.8	45.7
CSwin-B [9]	97M	526G	48.7	70.4	53.9	43.9	67.8	47.3
FocalNet-B (SRF) [46]	109M	496G	48.8	70.7	53.5	43.3	67.5	46.5
RDNet-B	107M	493G	48.8	70.4	53.5	43.4	67.5	46.6

Table F: COCO object detection and segmentation results - Cascade Mask-RCNN 3x schedule. FLOPs (G) are calculated with image size (1280, 800). The result of Swin-T is from the official repository [1].

Backbone	Param	FLOPs	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
Swin-T [26]	86M	745G	50.4	69.2	54.7	43.7	66.6	47.3
PVTv2-B2 [43]	83M	788G	51.1	69.8	55.3	-	-	-
FocalNet-T [46]	86M	746G	51.5	70.1	55.8	-	-	-
ConvNeXt-T [27]	86M	741G	50.4	69.1	54.8	43.7	66.5	47.3
NAT-T [12]	85M	737G	51.4	70.0	55.9	44.5	67.6	47.9
RDNet-T	81M	757G	51.6	70.5	56.0	44.6	67.9	48.3
Swin-S [26]	107M	838G	51.9	70.7	56.3	45.0	68.2	48.8
ConvNeXt-S [27]	108M	827G	51.9	70.8	56.5	45.0	68.4	49.1
NAT-S [12]	108M	809G	52.0	70.4	56.3	44.9	68.1	48.6
RDNet-S	108M	832G	52.3	70.8	56.6	45.4	68.5	49.3
Swin-B [26]	145M	982G	51.9	70.5	56.4	45.0	68.1	48.9
ConvNeXt-B [27]	146M	964G	52.7	71.3	57.2	45.6	68.9	49.5
NAT-B [12]	147M	931G	52.5	71.1	57.1	45.2	68.6	49.0
RDNet-B	144M	971G	52.9	71.5	57.2	46.0	69.1	50.0

fine-tuning of training hyperparameters for maximum precisions compared with ConvNeXt's, which indicates additional potential for achieving higher accuracy.

F Transferability Evaluation

We benchmark the transferability of ImageNet-1K pre-trained models. We utilize fine-grained classification datasets - CIFAR-10 [21], CIFAR-100 [21], Flowers-

Table G: Top-1 accuracy (%), parameter count (M), and FLOPs (G) on various classification tasks with ImageNet-1k pre-trained models.

Model	Param	FLOPs	CIFAR10	CIFAR100	Flowers	Cars	iNat18	iNat19
Grafit ResNet-50 [40]	26	4.1	-	-	98.2	92.5	69.8	75.9
DeiT-III-S [38]	22	4.6	98.9	90.6	96.4	89.9	67.1	72.7
RDNet-T	24	5.0	98.9	90.4	98.6	93.9	77.0	81.2
ResNet-152 [6]	60	11.6	-	-	-	-	69.1	-
RDNet-S	50	8.7	99.0	91.1	98.4	94.2	79.1	82.4
ViT-B/16 [10]	86	35.1	98.1	87.1	89.5	-	-	-
ViT-B/16 [35]	86	35.1	-	87.8	96.0	-	-	-
DeiT-B [37]	87	17.5	99.1	90.8	98.4	92.1	73.2	77.7
DeiT-III-B [38]	87	17.5	99.3	92.5	98.6	93.4	73.6	78.0
RDNet-B	87	15.4	99.3	91.4	98.6	94.1	80.5	83.5
RDNet-L	186	34.7	99.3	91.5	98.6	94.2	81.5	83.7
ViT-L/16 [10]	303	122.9	97.9	86.4	89.7	-	-	-
ViT-L/16 [35]	303	122.9	-	86.2	91.4	-	-	-
DeiT-III-L [38]	304	61.6	99.3	93.4	98.9	94.5	75.6	79.3

102 [29], and Stanford-Cars [20]. Furthermore, to verify the ability of long-tailed classification, we utilize iNaturalist-2018 [17] and iNaturalist-2019 [18]. We follow the training recipe of DeiT [37]. As demonstrated in Table G, RDNet exhibits strong transferability. Notably, RDNet delivers impressive performance on long-tailed classification tasks. RDNet-T surpasses DeiT-III-L on iNaturalist-2018 and iNaturalist-2019.

G Dense Connections for Generative Models

We test a generative model, WGAN [2], by replacing ResNet-GAN in the generator with our concatenation-based model. We maintained identical discriminator architecture, training setups, and similar model sizes. We focus on a proof of concept on CIFAR-10 to showcase whether our models could outperform with faster convergence. Table H results confirmed that our model works across multiple runs. We plan to extend our research to larger models in future work.

Table H: WGAN experiments. We demonstrate the capability of dense connections in RDNet for generation tasks. We utilize the WGAN architecture, redesigning only the generator to showcase whether the generation ability could improve.

Model	Param	IS↑	FID↓
ResBlock	5.4M	7.27±0.26	27.79±1.06
Ours	5.6M	**7.52±0.10**	**25.37±0.21**

Table I: Robustness evaluation. We compare the models evaluating the out-of-distribution (OOD) metrics ImageNet-V2/A/Sketch/ObjNet/R. We further average the OOD scores to show the averaged distribution shifts denoted by **Avg Shift**. Interestingly, even when RDNet demonstrates lower accuracy on ImageNet-1K compared to other networks, it consistently attains high OOD scores compared with other datasets.

Model	Param	FLOPs	IN	Avg Shift	V2	Obj	A	Sketch	R
Swin-T [26]	28	4.5	81.3	38.9	69.7	33.1	21.1	29.3	41.5
ConvNeXt-T [27]	29	4.5	82.1	42.7	72.5	35.6	24.2	33.8	47.2
HorNet-T [32]	22	4.0	82.8	43.4	72.3	37.5	26.6	34.1	46.6
SLaK-T [25]	30	5.0	82.5	43.3	72.0	36.6	30.0	32.4	45.3
NAT-T [12]	28	4.3	83.2	44.0	72.2	37.8	33.0	31.9	44.9
RDNet-T	24	5.0	82.8	44.7	72.9	36.9	27.7	37.0	49.0
Swin-S [26]	50	8.7	83.0	43.8	72.0	36.8	32.5	32.3	45.2
ConvNeXt-S [27]	50	8.7	83.1	45.7	72.5	38.0	31.3	37.1	49.6
HorNet-S [32]	50	8.8	84.0	47.3	73.6	39.9	36.2	36.9	49.7
SLaK-S [25]	55	9.8	83.8	48.2	73.6	39.6	39.3	37.5	50.9
NAT-S [12]	51	7.8	83.7	46.4	73.2	39.9	37.4	34.3	47.3
RDNet-S	50	8.7	83.7	47.8	73.8	39.3	33.5	39.8	52.8
Swin-B [26]	88	15.4	83.5	44.9	72.4	37.6	35.4	32.7	46.5
ConvNeXt-B [27]	89	15.4	83.8	47.9	73.7	39.9	36.7	38.2	51.2
HorNet-B [32]	87	15.6	84.3	48.8	73.9	41.0	39.9	38.1	51.2
SLaK-B [25]	95	17.1	84.0	48.9	74.0	39.7	41.6	38.5	50.8
NAT-B [12]	90	13.7	84.3	48.5	74.1	40.7	41.4	36.6	49.7
RDNet-B	87	15.4	84.4	49.0	74.2	39.7	38.1	40.1	52.7
ConvNeXt-L [27]	198	34.4	84.3	49.9	74.2	40.6	41.3	40.1	53.5
RDNet-L	186	34.7	84.8	52.2	75.0	42.1	42.9	44.5	56.5

H Robustness Evaluation

We further evaluate the robustness of our models using the ImageNet out-of-distribution (OOD) benchmarks - ImageNet-V2 [33], ImageNet-A [16], ImageNet-Sketch [41], ImageNet-R [15], and ObjectNet [4]. Table I shows our RDNet demonstrates superior robustness in comparison to the other models. We specifically select models (HorNet, SLaK, and NAT) having comparable ImageNet-1K accuracies to demonstrate the superior out-of-distribution (OOD) performance of our models compared with those. Nobaly, even when RDNet demonstrates lower accuracy on ImageNet-1K than competing models, RDNet achieves high OOD scores across various benchmarks.

I More Details and Extended Pilot Study

I.1 More Details of Pilot Study

We present individual RandNet experimental results performed under controlled setups. We conduct 200 experiments for each configuration of budget, block type, and skip connection type. Fig. B and Table J, concatenation outperforms addition across parameter spaces $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$. For the parameter spaces $\mathcal{D}$ and

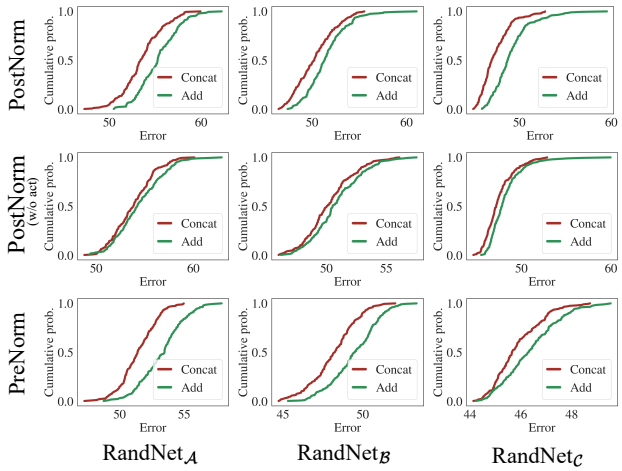


Fig. B: Cumulative prabability vs. error of trained models in Table J is visualized here following Radosavovic *et al.* [31]. Each row demonstrates three block types, and each column exhibits parameter spaces $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$.

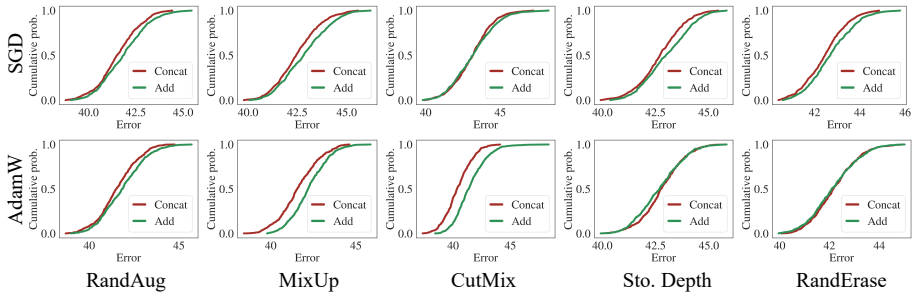


Fig. C: Cumulative prabability vs. error of trained models in Table K is visualized here following Radosavovic *et al.* [31]. The figure corresponds to the manuscript's parameter spaces $\mathcal{D}$ and $\mathcal{E}$. Each row demonstrates optimizers, and each column exhibits augmentations.

$\mathcal{E}$ using data augmentations, we use the following hyper-parameters for experiments. For RandAugment [8], we limit the magnitude values of {3, 5, 7}; for MixUp [48] and CutMix [47], we employ alpha values of {0.1, 0.3, 0.5} respectively; Stochastic Depth [19] ratio are set to {0.05, 0.1}; Random Erasing [49] is fixed to 0.25.

Due to the diverse setups in each data argumentation, we conduct 600 experiments for each element. Additionally, even when switching the optimizer to AdamW, we train 600 random networks for every augmentation as well. Fig. C and Table K suggest that that data augmentation reduces the performance gap between additive and concatenative shortcuts, particularly noting that the element (*i.e.*, trained using stochastic depth with AdamW) benefit more from an additive shortcut. Nevertheless, we continue to observe that concatenation-based

Table J: Concatenation vs. addition. The table corresponds to the manuscript's parameter spaces $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$. We sample 200 random networks within each parameter space, ensuring similar computational costs of FLOPs, the number of parameters (Param), and memory consumption (Mem), and individually train them on Tiny-ImageNet. All results are averaged along with the standard deviation (± std). Experimental results with higher accuracy are shaded in gray.

Model	Skip type	Block type	FLOPs (G)	Param (M)	Mem (GB)	Top-1 (%)
RandNet$_{\mathcal{A}}$	Add	PostNorm	2.24±0.14	2.20±0.13	0.66±0.03	45.6±2.3
RandNet$_{\mathcal{A}}$	Concat	PostNorm	2.24±0.14	2.25±0.14	0.71±0.05	**46.9±2.2**
RandNet$_{\mathcal{A}}$	Add	PostNorm (w/o act)	2.24±0.13	2.20±0.13	0.66±0.02	45.1±2.1
RandNet$_{\mathcal{A}}$	Concat	PostNorm (w/o act)	2.23±0.14	2.24±0.14	0.76±0.06	**46.8±2.2**
RandNet$_{\mathcal{A}}$	Add	PreNorm	2.24±0.13	2.2±0.13	0.66±0.02	46.8±1.67
RandNet$_{\mathcal{A}}$	Concat	PreNorm	2.25±0.14	2.25±0.14	0.82±0.08	**48.7±1.6**
RandNet$_{\mathcal{B}}$	Add	PostNorm	4.51±0.27	4.42±0.26	0.78±0.05	49.9±2.2
RandNet$_{\mathcal{B}}$	Concat	PostNorm	4.53±0.28	4.52±0.28	0.88±0.09	**50.7±2.2**
RandNet$_{\mathcal{B}}$	Add	PostNorm (w/o act)	4.52±0.27	4.43±0.26	0.78±0.05	49.4±2.3
RandNet$_{\mathcal{B}}$	Concat	PostNorm (w/o act)	4.51±0.28	4.50±0.28	0.84±0.08	**50.7±2.1**
RandNet$_{\mathcal{B}}$	Add	PreNorm	4.51±0.26	4.42±0.26	0.78±0.05	51.1±1.6
RandNet$_{\mathcal{B}}$	Concat	PreNorm	4.50±0.30	4.48±0.29	0.97±0.12	**52.2±1.5**
RandNet$_{\mathcal{C}}$	Add	PostNorm	9.58±0.23	9.37±0.23	1.00±0.09	52.8±2.2
RandNet$_{\mathcal{C}}$	Concat	PostNorm	9.55±0.27	9.46±0.27	1.05±0.11	**53.8±2.1**
RandNet$_{\mathcal{C}}$	Add	PostNorm (w/o act)	9.60±0.24	9.39±0.23	1.02±0.09	52.5±2.2
RandNet$_{\mathcal{C}}$	Concat	PostNorm (w/o act)	9.56±0.27	9.46±0.26	1.11±0.13	**53.9±2.1**
RandNet$_{\mathcal{C}}$	Add	PreNorm	9.58±0.22	9.37±0.21	1.02±0.10	54.4±1.6
RandNet$_{\mathcal{C}}$	Concat	PreNorm	9.56±0.26	9.46±0.25	1.24±0.17	**55.1±1.3**

Table K: Concatenation vs. addition with data augmentations. The table corresponds to the manuscript's parameter spaces $\mathcal{D}$ and $\mathcal{E}$. We utilize the PreNorm block for augmentation experiments. We sample 600 random networks due to diverse degrees of data augmentations and report identically in Table J. Experimental results with higher accuracy are shaded in gray.

Skip type	Augmentation	AdamW	FLOPs (G)	Param (M)	Mem (GB)	Top-1 (%)
Add	RandAug		9.60±0.23	9.40±0.23	1.02±0.09	57.8±1.3
Concat	RandAug		9.54±0.26	9.44±0.26	1.25±0.17	**58.7±1.2**
Add	MixUp		9.59±0.23	9.39±0.22	1.02±0.09	57.3±1.4
Concat	MixUp		9.55±0.26	9.45±0.26	1.24±0.16	**57.9±1.2**
Add	CutMix		9.60±0.23	9.39±0.23	1.03±0.09	57.0±1.7
Concat	CutMix		9.55±0.27	9.45±0.26	1.23±0.16	**57.5±1.5**
Add	Sto. Depth		9.59±0.23	9.39±0.23	1.02±0.09	57.3±1.4
Concat	Sto. Depth		9.55±0.27	9.45±0.26	1.24±0.17	**57.8±1.2**
Add	RandErase		9.59±0.23	9.38±0.23	1.02±0.09	57.8±1.3
Concat	RandErase		9.56±0.26	9.45±0.26	1.24±0.16	**58.2±1.2**
Add	RandAug	✓	9.60±0.23	9.40±0.23	1.02±0.09	58.5±1.4
Concat	RandAug	✓	9.54±0.26	9.44±0.26	1.25±0.17	**59.2±1.3**
Add	MixUp	✓	9.60±0.23	9.39±0.22	1.02±0.09	58.2±1.3
Concat	MixUp	✓	9.55±0.25	9.44±0.25	1.24±0.16	**58.9±1.3**
Add	CutMix	✓	9.59±0.23	9.38±0.23	1.02±0.09	58.9±1.6
Concat	CutMix	✓	9.54±0.26	9.44±0.26	1.25±0.17	**60.2±1.4**
Add	Sto. Depth	✓	9.60±0.23	9.39±0.23	1.02±0.09	**57.5±1.3**
Concat	Sto. Depth	✓	9.55±0.26	9.44±0.26	1.25±0.16	57.5±1.1
Add	RandErase	✓	9.58±0.23	9.37±0.23	1.02±0.08	**58.1±1.1**
Concat	RandErase	✓	9.56±0.26	9.46±0.26	1.24±0.18	58.1±1.0

models dominate over advantage over additive shortcuts, even in the AdamW training configurations.

I.2 Scaled-up Pilot Study

To further extend our pilot study, we conduct ImageNet-1K experiments across the parameter spaces $\mathcal{C}$, $\mathcal{D}$, and $\mathcal{E}$, as described in §5.1. For the parameter space $\mathcal{C}$, we train the model for 60 epochs. For $\mathcal{D}$ and $\mathcal{E}$, we extend the training epochs to 90 to assess the impact of augmentation. In the parameter space $\mathcal{C}$, we carry out 150 experiments for each type of skip connection. In $\mathcal{D}$ and $\mathcal{E}$, we conduct 60 experiments for each combination of augmentation and skip connection type. As shown in Table L, even on the ImageNet-1K dataset, the trained models using concatenation statistically surpass those using addition.

Table L: Scaled-up pilot study on ImageNet-1K - concatenation vs. addition experiments. The table corresponds to our extended pilot study on the same parameter spaces $\mathcal{C}$, $\mathcal{D}$ and $\mathcal{E}$. Each row in parameter space $\mathcal{C}$ contains statistics derived from 150 experiments, while each row in parameter spaces $\mathcal{D}$ and $\mathcal{E}$ encompasses statistics from 300 experiments. We utilize the PreNorm block only for scaled-up experiments. Experimental results with higher accuracy are shaded in gray.

Model	Skip type	FLOPs (G)	Param (M)	Mem (GB)	Top-1 (%)
RandNet$_{\mathcal{C}}$	Add	9.51±0.28	9.41±0.28	1.02±0.09	54.9±2.2
RandNet$_{\mathcal{C}}$	Concat	9.45±0.22	9.47±0.20	1.24±0.17	**55.1±2.0**
RandNet$_{\mathcal{D}}$	Add	9.60±0.24	9.46±0.20	1.02±0.09	54.7±2.6
RandNet$_{\mathcal{D}}$	Concat	9.55±0.26	9.45±0.26	1.25±0.16	**55.1±2.7**
RandNet$_{\mathcal{E}}$	Add	9.60±0.27	9.38±0.26	1.02±0.09	55.8±2.8
RandNet$_{\mathcal{E}}$	Concat	9.55±0.27	9.44±0.26	1.25±0.16	**56.7±2.7**

References

1. Github repository: Swin transformer for object detection, `https://github.com/SwinTransformer/Swin-Transformer-Object-Detection`
2. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: International Conference on Machine Learning (ICML) (2017)
3. Bao, H., Dong, L., Wei, F.: Beit: Bert pre-training of image transformers. In: International Conference on Learning Representations (ICLR) (2022)
4. Barbu, A., Mayo, D., Alverio, J., Luo, W., Wang, C., Gutfreund, D., Tenenbaum, J., Katz, B.: Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Conference on Neural Information Processing Systems (NeurIPS). vol. 32 (2019)
5. Cai, Z., Vasconcelos, N.: Cascade r-cnn: High-quality object detection and instance segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (2019)
6. Chu, P., Bian, X., Liu, S., Ling, H.: Feature space augmentation for long-tailed data. arXiv preprint arXiv:2008.03673 (2020)
7. Clark, K., Luong, M.T., Le, Q.V., Manning, C.D.: Electra: Pre-training text encoders as discriminators rather than generators. In: International Conference on Learning Representations (ICLR) (2020)
8. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V.: Randaugment: Practical automated data augmentation with a reduced search space. In: IEEE Transactions on Computer Vision and Pattern Recognition Workshop (CVPRW). pp. 702–703 (2020)
9. Dong, X., Bao, J., Chen, D., Zhang, W., Yu, N., Yuan, L., Chen, D., Guo, B.: Cswin transformer: A general vision transformer backbone with cross-shaped windows. In: IEEE Transactions on Computer Vision and Pattern Recognition (CVPR) (2022)
10. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (ICLR) (2020)
11. Guo, M.H., Lu, C.Z., Liu, Z.N., Cheng, M.M., Hu, S.M.: Visual attention network. Computational Visual Media (CVMJ) (2023)
12. Hassani, A., Walton, S., Li, J., Li, S., Shi, H.: Neighborhood attention transformer. In: IEEE Transactions on Computer Vision and Pattern Recognition (CVPR). pp. 6185–6194 (2023)
13. He, K., Girshick, R., Dollár, P.: Rethinking imagenet pre-training. In: International Conference on Computer Vision (ICCV). pp. 4918–4927 (2019)
14. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: International Conference on Computer Vision (ICCV) (2017)
15. Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., Song, D., Steinhardt, J., Gilmer, J.: The many faces of robustness: A critical analysis of out-of-distribution generalization. International Conference on Computer Vision (ICCV) (2021)
16. Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., Song, D.: Natural adversarial examples. IEEE Transactions on Computer Vision and Pattern Recognition (CVPR) (2021)
17. Horn, G.V., Mac Aodha, O., Song, Y., Shepard, A., Adam, H., Perona, P., Belongie, S.J.: The inaturalist challenge 2018 dataset. arXiv preprint arXiv:1707.06642 (2018)

18. Horn, G.V., Mac Aodha, O., Song, Y., Shepard, A., Adam, H., Perona, P., Belongie, S.J.: The iNaturalist species classification and detection dataset. arXiv preprint arXiv:1707.06642 (2019)
19. Huang, G., Sun, Y., Liu, Z., Sedra, D., Weinberger, K.Q.: Deep networks with stochastic depth. In: European Conference on Computer Vision (ECCV) (2016)
20. Krause, J., Stark, M., Deng, J., Fei-Fei, L.: 3d object representations for fine-grained categorization. In: IEEE Workshop on 3D Representation and Recognition (2013)
21. Krizhevsky, A.: Learning multiple layers of features from tiny images. Tech. rep., CIFAR (2009)
22. Li, S., Wang, Z., Liu, Z., Tan, C., Lin, H., Wu, D., Chen, Z., Zheng, J., Li, S.Z.: Moganet: Multi-order gated aggregation network. In: International Conference on Learning Representations (ICLR) (2024), https://openreview.net/forum?id=XhYWgjqCrV
23. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European Conference on Computer Vision (ECCV). pp. 740–755. Springer (2014)
24. Lin, W., Wu, Z., Chen, J., Huang, J., Jin, L.: Scale-aware modulation meet transformer. In: International Conference on Computer Vision (ICCV). pp. 5992–6003 (10 2023)
25. Liu, S., Chen, T., Chen, X., Chen, X., Xiao, Q., Wu, B., Pechenizkiy, M., Mocanu, D.C., Wang, Z.: More convnets in the 2020s: Scaling up kernels beyond 51x51 using sparsity. In: International Conference on Learning Representations (ICLR) (2023)
26. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: International Conference on Computer Vision (ICCV) (2021)
27. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. In: IEEE Transactions on Computer Vision and Pattern Recognition (CVPR). pp. 11976–11986 (2022)
28. Mingfei Ma, Vitaly Fedyunin, W.W.: Accelerating pytorch vision models with channels last on cpu (2022), https://pytorch.org/blog/accelerating-pytorch-vision-models-with-channels-last-on-cpu/
29. Nilsback, M.E., Zisserman, A.: Automated flower classification over a large number of classes. In: Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing (2008)
30. Polyak, B., Juditsky, A.B.: Acceleration of stochastic approximation by averaging. Siam Journal on Control and Optimization **30**, 838–855 (1992)
31. Radosavovic, I., Kosaraju, R.P., Girshick, R.B., He, K., Dollár, P.: Designing network design spaces. In: IEEE Transactions on Computer Vision and Pattern Recognition (CVPR). pp. 10425–10433 (2020)
32. Rao, Y., Zhao, W., Tang, Y., Zhou, J., Lim, S.N., Lu, J.: Hornet: Efficient high-order spatial interactions with recursive gated convolutions. In: Conference on Neural Information Processing Systems (NeurIPS) (2022)
33. Recht, B., Roelofs, R., Schmidt, L., Shankar, V.: Do imagenet classifiers generalize to imagenet? In: International Conference on Machine Learning (ICML) (2019)
34. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: Imagenet large scale visual recognition challenge. International Journal of Computer Vision (IJCV) **115**(3), 211–252 (2015)

35. Steiner, A., Kolesnikov, A., Zhai, X., Wightman, R., Uszkoreit, J., Beyer, L.: How to train your vit? data, augmentation, and regularization in vision transformers. arXiv preprint arXiv:2106.10270 (2021)
36. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. IEEE Transactions on Computer Vision and Pattern Recognition (CVPR) pp. 2818–2826 (2016)
37. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jegou, H.: Training data-efficient image transformers & distillation through attention. In: International Conference on Machine Learning (ICML). pp. 10347–10357 (2021)
38. Touvron, H., Cord, M., J'egou, H.: Deit iii: Revenge of the vit. In: European Conference on Computer Vision (ECCV) (2022)
39. Touvron, H., Cord, M., Sablayrolles, A., Synnaeve, G., J'egou, H.: Going deeper with image transformers. International Conference on Computer Vision (ICCV) pp. 32–42 (2021)
40. Touvron, H., Sablayrolles, A., Douze, M., Cord, M., Jégou, H.: Grafit: Learning fine-grained image representations with coarse labels. International Conference on Computer Vision (ICCV) (2021)
41. Wang, H., Ge, S., Lipton, Z., Xing, E.P.: Learning robust global representations by penalizing local predictive power. In: Conference on Neural Information Processing Systems (NeurIPS). pp. 10506–10518 (2019)
42. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In: International Conference on Computer Vision (ICCV). pp. 548–558 (2021)
43. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pvtv2: Improved baselines with pyramid vision transformer. Computational Visual Media (CVMJ) (2022)
44. Wightman, R.: Github repository: Pytorch image models, `https://github.com/huggingface/pytorch-image-models`
45. Xiao, T., Liu, Y., Zhou, B., Jiang, Y., Sun, J.: Unified perceptual parsing for scene understanding. In: European Conference on Computer Vision (ECCV). Springer (2018)
46. Yang, J., Li, C., Dai, X., Gao, J.: Focal modulation networks. In: Conference on Neural Information Processing Systems (NeurIPS) (2022)
47. Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., Yoo, Y.: Cutmix: Regularization strategy to train strong classifiers with localizable features. In: International Conference on Computer Vision (ICCV). pp. 6023–6032 (2019)
48. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. In: International Conference on Learning Representations (ICLR) (2018)
49. Zhong, Z., Zheng, L., Kang, G., Li, S., Yang, Y.: Random erasing data augmentation. In: AAAI Conference on Artificial Intelligence (AAAI). pp. 13001–13008 (2020)
50. Zhu, L., Wang, X., Ke, Z., Zhang, W., Lau, R.: Biformer: Vision transformer with bi-level routing attention. IEEE Transactions on Computer Vision and Pattern Recognition (CVPR) (2023)