UniConvNet: Expanding Effective Receptive Field while Maintaining Asymptotically Gaussian Distribution for ConvNets of Any Scale

Yuhao Wang
Xi'an Jiaotong University
yuhaowang.ai@gmail.com

Wei Xi *
Xi'an Jiaotong University
xiwei@xjtu.edu.cn

Abstract

Convolutional neural networks (ConvNets) with large effective receptive field (ERF), still in their early stages, have demonstrated promising effectiveness while constrained by high parameters and FLOPs costs and disrupted asymptotically Gaussian distribution (AGD) of ERF. This paper proposes an alternative paradigm: rather than merely employing extremely large ERF, it is more effective and efficient to expand the ERF while maintaining AGD of ERF by proper combination of smaller kernels, such as 7×7, 9×9, 11×11. *This paper introduces a Three-layer Receptive Field Aggregator and designs a Layer Operator as the fundamental operator from the perspective of receptive field. The ERF can be expanded to the level of existing large-kernel ConvNets through the stack of proposed modules while maintaining AGD of ERF. Using these designs, we propose a universal model for ConvNet of any scale, termed UniConvNet. Extensive experiments on ImageNet-1K, COCO2017, and ADE20K demonstrate that UniConvNet outperforms state-of-the-art CNNs and ViTs across various vision recognition tasks for both lightweight and large-scale models with comparable throughput. Surprisingly, UniConvNet-T achieves* 84.2% *ImageNet top-1 accuracy with* $30M$ *parameters and* $5.1G$ *FLOPs. UniConvNet-XL also shows competitive scalability to big data and large models, acquiring* 88.4% *top-1 accuracy on ImageNet. Code and models are publicly available at https://github.com/ai-paperwithcode/UniConvNet.*

1. Introduction

With the impressive triumph of transformers [19, 38], constructing long-range dependencies has become a crucial principle in designing convolutional neural networks (ConvNets). Some prior works [17, 18, 37] have made attempts to capture relationships across large receptive fields, surpassing traditional convolutional neural networks [23, 45, 49, 59, 82] and achieving significant improvements in various vision recognition tasks, such as image classification, object detection, instance segmentation, and semantic segmentation. Current ConvNets achieve long-range dependencies by scaling up the convolutional kernel with reparameterization [15, 17], parameter sharing [5] or sparsity [37] techniques. Some recent works leverage the key properties of large kernels [18] or encode their interactions [32] to inform ConvNet architecture design. They benefit from large ERF while constrained by high parameters and FLOPs costs.

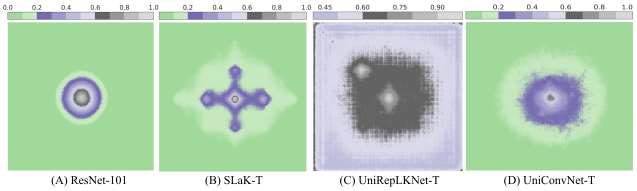


Figure 1. Effective Receptive Field (ERF) of ResNet-101, SLaK-T, UniRepLKNet-T, UniConvNet-T. The more stepped colour area from the center indicates better asymptotically Gaussian distribution (AGD) of ERF. The wider area indicates a larger ERF. Large-kernel ConvNets, such as SLaK-T and UniRepLKNet-T, disrupt the AGD of ERF.

A typical paradigm [21, 23, 70] for ConvNets is to use a stack of many small spatial convolutions (*e.g.* 3×3) to enlarge the receptive fields in ConvNets. Why small-kernel ConvNets constrained by small ERF still get effective performance? The conventional ConvNets, such as ResNet-101 [23], have a small ERF but benefit from multi-scale impact (gradient), which follows AGD, through the stack of 3×3 convolution modules, as shown in Fig. 1 (A). This suggests that smaller-scale pixels, around the position of the output pixel, of the input should have more impact on the output pixel. Large-kernel ConvNets, such as SLaK-T [37] and UniRepLKNet-T [18], achieve a large ERF but disrupt the AGD, which either obtain a discriminative impact at weird position or get similar impacts of different scales, as depicted in Fig. 1 (B) and Fig. 1 (C).

Is there a proper way to combine smaller kernels to ex-

*Corresponding author

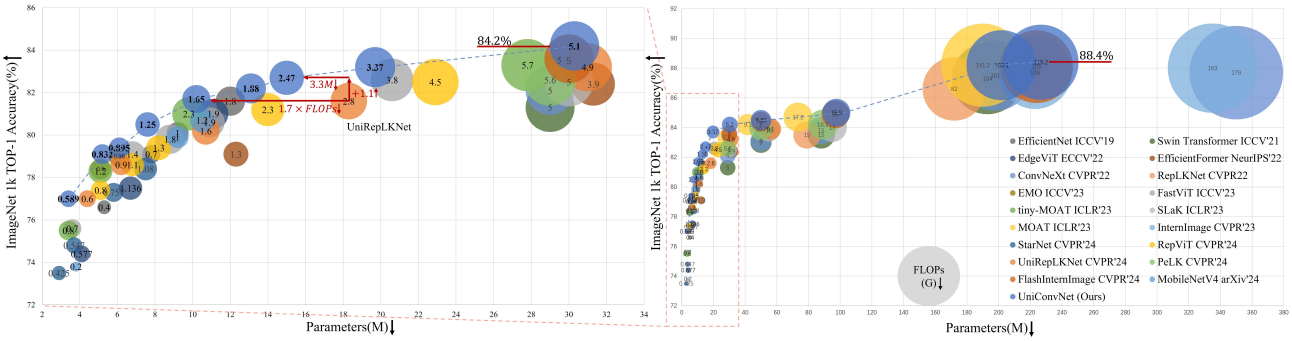


Figure 2. **Comparison of parameters and accuracy between UniConvNet (ours) and others.** The area of the circle, the number in the circle, represents the FLOPs of the relevant model. UniConvNet achieves the best accuracy-parameter and accuracy-FLOPs trade-off.

pand ERF while maintaining the AGD of ERF? This paper proposes an alternative paradigm: rather than merely employing extremely large ERF, it is more effective and efficient to expand the ERF while maintaining the AGD of ERF by proper combination of smaller kernels.

To answer this question, we introduce a Receptive Field Aggregator (RFA) for ConvNets, designed to obtain AGD at a shadow module by directly assign impact for different scales. The input images are separated into multiple heads according to the layer of RFA. Parameters and FLOPs costs are reduced by recursively feeding multi-head inputs into layer operators, creating a pyramidal increment among channels. For heads with different patterns among channels, in each layer, we propose a spatial encoder from the perspective of receptive field, called the Layer Operator (LO). The LO consists of two components: the Amplifier (Amp) and the Discriminator (Dis). The Amp expands the scale of the receptive field and amplifies the impacts of pixels on the receptive field by element-wise multiplication. The salient pixels in the receptive field will have a more distinguished impact. The Dis provides small-scale impacts from new pixels to the large receptive field produced by the Amp. The final receptive field becomes a large two-layer receptive field. Sequentially, each LO expands and amplifies the receptive field of the previous LO by Amp and provides a discriminative receptive field for adding impacts of small-scale pixels. The final receptive field of three-layer RFA results in a four-layer receptive field following an AGD. The ERF can be expanded and the AGD of ERF can be maintained through the stack of many RFA modules, as illustrated in Fig. 1 (D).

With these designs, the proposed UniConvNet efficiently reduces parameters and FLOPs while getting a multi-scale impact on ERF. Consequently, it outperforms state-of-the-art CNNs and ViTs in various vision recognition tasks, from lightweight to large-scale models, as illustrated in Fig. 2. Notably, UniConvNet-T achieves 84.2% TOP-1 accuracy, surpassing models with similar parameters and FLOPs by at least 0.6 points, representing a significant improvement over existing ConvNets [5, 37, 40, 72]. UniConvNet-XL breaks through ConvNet bottleneck, achieving 88.4% TOP-1 accuracy with a superior parameters and FLOPs trade-off compared to contemporary CNNs [17, 32, 40, 50, 69, 72] and ViTs [13, 38, 39, 74]. UniConvNet is also powerful on downstream tasks. UniConvNet-L obtains 55.7% on COCO [34] and 55.1% on ADE20K [85].

We believe the high performance of UniConvNet is mainly because of the large ERF[42] while maintaining AGD as compared in Fig. 1. The ERF scale is comparable with ConvNets using extremly large kernels. The AGD of ERF are more similar to the intuition that the closer pixels should have more impacts. We hope our findings can help to understand the intrinsic mechanism of ConvNets.

2. Model Architecture

2.1. Rceptive Field Aggregator

In order to expand the effective receptive field (ERF) [42] while maintianning the asymptotically Gaussian distribution (AGD) of ERF, we introduce a Rceptive Field Aggregator (RFA) as illustrated in Fig. 3 (left), which directly assign discriminative impact on receptive field of different scales at a shadow module. Specifically, input images are initially divided into $N+1$ parts along the channel dimension based on the layer N of RFA resulting in $N+1$ heads: A_1, H_1, ..., H_n, ..., H_N. The input heads can be categorized into two parts: A_1 and H_1, ..., H_n, ..., H_N. Concretely, the input head $A_1 \in \mathbb{R}^{B\times\frac{C}{N+1}\times H\times W}$, where B, $\frac{C}{N+1}$, H, and W represent the batch size, channel dimension, height, and width, respectively, is initially fed into the Layer Operator (LO) 1, resulting in a new head A_2. The channel dimension of the output head A_2 increases from $\frac{C}{N+1}$ to $\frac{2}{N+1}C$. Sub-

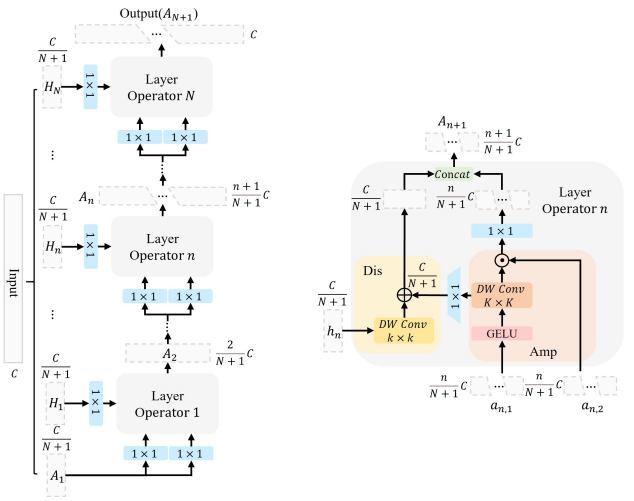


Figure 3. **Left:** Receptive Field Aggregator. **Right:** Layer Operator.

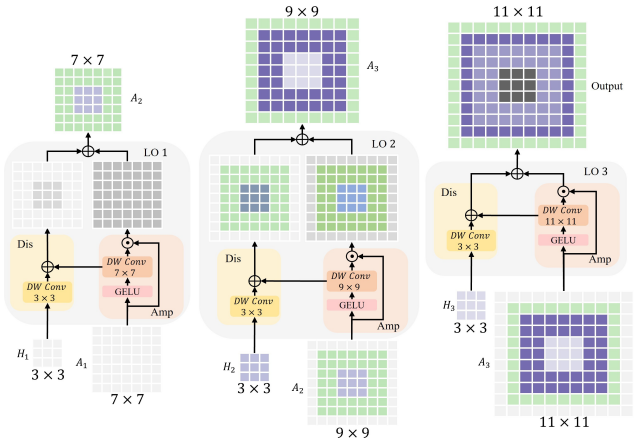


Figure 4. **Receptive Field Flow of the Three-layer Receptive Field Aggregator.**

sequently, the output head A_2 is recursively fed into the LO n, illustrated in Fig. 3 (right), according to the layer number n ($n \in [2, N]$), with its channel dimension increasing from $\frac{n}{N+1}C$ to $\frac{n+1}{N+1}C$. The remaining N input heads, H_1, ..., H_n, ..., H_N are sequentially fed into the LO n according to the layer number n ($n \in [1, N]$) to interact with the corresponding input head A_n. Each head is initially projected using 1×1 convolution before feeding in the LO to enhance feature diversity. In the RFA, the output channels of heads A_n follow a pyramidal increment, reducing parameters and FLOPs compared to the standard direct-in, direct-out construction. Increasing the layer N allows for higher input image resolution, potentially providing a more effective alternative to training on low-resolution images followed by fine-tuning at high resolutions.

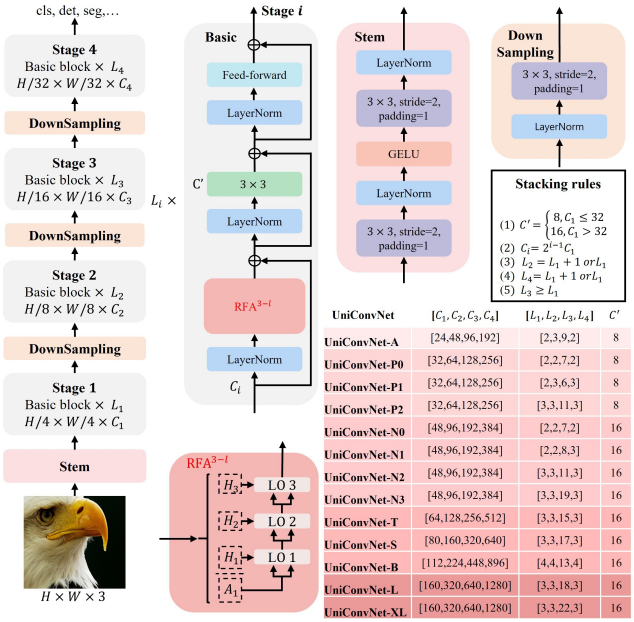
2.2. Layer Operator

To effectively expand receptive field and assign discriminative impact on receptive field, we introduce the Layer Operator (LO). This operator is designed from the perspective of receptive field and serves as the core operator in the RFA, illustrated in Fig. 3 (right). This technique can construct a two-layer AGD of receptive field. Specifically, for layer number n, the three distinct inputs for LO n are $a_{n,1}$, $a_{n,2}$, and h_n, projected by three individual 1×1 convolutions as illustrated in Fig. 3 (left). The LO is generated by interacting two components, the Discriminator (Dis) and the Amplifier (Amp), as illustrated in Fig. 3 (right). For the Amp, we conduct an element-wise multiplication between $a_{n,2}$ and output features derived from a GELU activation and a depth-wise large-kernel $K \times K$ convolution applied to $a_{n,1}$. For each output pixel in the Amp module, the corresponding pixels within the relevant $K \times K$ receptive field are multiplied by the pixel value at the same spatial location in $a_{n,2}$. This operation expands the receptive field and amplifies the impact of pixels on the receptive field. Additionally, for the Dis, we incorporate features from depth-wise $K \times K$ and $k \times k$ convolutions. This introduces impact from small-scale new pixels for the large $K \times K$ receptive field, establishing a two-layer discriminative AGD. The 1×1 convolutions facilitate information interaction among channels and change the channel dimension for the compatibility of features. The outputs of the Amp and the Dis are concatenated, resulting in the final output A_{n+1} with a two-layer AGD of receptive field and increased channels for the subsequent layer.

2.3. Three-layer RFA for UniConvNet

The layer number N varies with the resolution of the input image. In this work, for the input image size of 224×224, we construct a three-layer RFA using a layer $N = 3$. The progressive large-scale kernel size K in the LO is calculated as $K = 2n + 5 (n \in [1, N])$. The small-scale kernel size is $k = 3$. Thus, the RFA achieves a four-layer AGD for receptive field within a shadow module using progressive large-kernel convolutions 7×7, 9×9 and 11×11, which is an optimal configuration for 224×224 images. We do an ablation study in Sec. 4.1.

The smallest kernel size of 7×7 provides a significantly larger receptive field than 3×3 and 5×5 convolutions. The largest convolution kernel size of 11×11 can maintain the 14×14 features with padding size 5 in stage 3, the main stage of the feature extraction in our models, depicted in Fig. 5, of our model to have at most quarter pixels of the feature at the corner for avoiding the frequent overlapping of the center pixels during the convolution progress. The efficiency and effectiveness are justified and analyzed in Sec. 4.2.



UniConvNet	$[C_1, C_2, C_3, C_4]$	$[L_1, L_2, L_3, L_4]$	C'
UniConvNet-A	[24,48,96,192]	[2,3,9,2]	8
UniConvNet-P0	[32,64,128,256]	[2,2,7,2]	8
UniConvNet-P1	[32,64,128,256]	[2,3,6,3]	8
UniConvNet-P2	[32,64,128,256]	[3,3,11,3]	8
UniConvNet-N0	[48,96,192,384]	[2,2,7,2]	16
UniConvNet-N1	[48,96,192,384]	[2,2,8,3]	16
UniConvNet-N2	[48,96,192,384]	[3,3,11,3]	16
UniConvNet-N3	[48,96,192,384]	[3,3,19,3]	16
UniConvNet-T	[64,128,256,512]	[3,3,15,3]	16
UniConvNet-S	[80,160,320,640]	[3,3,17,3]	16
UniConvNet-B	[112,224,448,896]	[4,4,13,4]	16
UniConvNet-L	[160,320,640,1280]	[3,3,18,3]	16
UniConvNet-XL	[160,320,640,1280]	[3,3,22,3]	16

Figure 5. **Overall architecture of UniConvNet.**

2.4. Receptive Field Flow of Three-layer RFA

We sketch the receptive field flow of the proposed three-layer RFA in Fig. 4. In LO 1, the receptive field scale is expanded and the impacts of pixels on 7×7 receptive field is amplified by the Amp module. The Dis module provide the impacts of small-scale new pixels, closer to the position of output pixel, for large receptive field generated by 7×7 convolution to build a discriminative receptive field. The output combines the two receptive fields to build a two-layer AGD by directly assigning impacts of different scales on the receptive field. Sequentially, each LO expands and amplifies the receptive field of the previous LO by Amp and provides a discriminative receptive field for adding impacts of small-scale new pixels. The final receptive field of three-layer RFA results in a four-layer receptive field following an AGD. The ERF can be expanded and the AGD of ERF can be maintained through the stack of many RFA modules, as illustrated in Fig. 1 (D).

2.5. UniConvNet Model

The Three-layer RFA can be integrated into any model as a plug-and-play module, effectively replacing each convolution in ConvNet architectures. For better performance, we directly integrate the Three-layer RFA module into the state-of-the-art CNN-based model, InternImage [69]. Specifically, the convolution used in the 3×3 convolution residual component is the DCNV3 in the InternImage [69] and we remove the softmax normalization in DCNV3 as DCNV4 did because the optimizations for memory on DCNV4 in the Efficient Deformable ConvNets [72] is incompatible with the other CNN-based models. We develop a new CNN-based backbone model called UniConvNet using various stacking strategies for its variants. The overall architecture and stacking rules are illustrated in Fig. 5. We present integral blocks of the model and provide details on the embedded blocks in Appendix C.

Model	Type	Scale	#Params	FLOPs	Acc(%)	Publication
EdgeViT-XXS [47]	T	256^2	4.1M	0.577G	74.4	ECCV'22
FastViT-T8 [64]	T	256^2	3.6M	0.7G	75.6	ICCV'23
PVTv2-B0 [68]	T	224^2	3.7M	0.572G	70.5	CVM'22
tiny-MOAT-0 [74]	T	224^2	3.4M	0.8G	75.5	ICLR'23
UniRepLKNet-A [18]	C	224^2	4.4M	0.6G	77.0	CVPR'24
StarNet-S2 [43]	C	224^2	3.7M	0.547G	74.8	CVPR'24
UniConvNet-A(ours)	**C**	**224^2**	**3.4M**	**0.589G**	**77.0**	**-**
EdgeNeXt-S [44]	T	256^2	5.6M	0.965G	78.8	ECCVW'22
MobileViTv1-S [45]	T	256^2	5.6M	2.01G	78.4	ICLR'22
tiny-MOAT-1 [74]	T	224^2	5.1M	1.2G	78.3	ICLR'23
EMO-5M [81]	T	224^2	5.1M	0.903G	78.4	ICCV'23
StarNet-S3 [43]	C	224^2	5.8M	0.757G	77.3	CVPR'24
RepViT-M0.9 [66]	C	224^2	5.1M	0.8G	77.4	CVPR'24
DCNV4 [72]	C	224^2	5.3M	0.805G	78.5	CVPR'24
UniConvNet-P0(ours)	**C**	**224^2**	**5.2M**	**0.832G**	**79.1**	**-**
EdgeViT-XS [47]	T	256^2	6.7M	1.136G	77.5	ECCV'22
FastViT-T12 [64]	T	256^2	6.8M	1.4G	79.1	ICCV'23
SMT-M [36]	T	224^2	6.5M	1.3G	78.4	ICCV'23
EMO-6M [81]	T	224^2	6.1M	0.961G	79.0	ICCV'23
StarNet-S4 [43]	C	224^2	7.5M	1.075G	78.4	CVPR'24
RepViT-M1.0 [66]	C	224^2	6.8M	1.1G	78.6	CVPR'24
UniRepLKNet-F [18]	C	224^2	6.2M	0.9G	78.6	CVPR'24
UniConvNet-P1(ours)	**C**	**224^2**	**6.1M**	**0.895G**	**79.6**	**-**
FastViT-S12 [64]	T	256^2	8.8M	1.8G	79.8	ICCV'23
EfficientNet-B2 [59]	C	288^2	9.2M	1.0G	80.1	ICCV'19
RepViT-M1.1 [66]	C	224^2	8.2M	1.3G	79.4	CVPR'24
UniConvNet-P2(ours)	**C**	**224^2**	**7.6M**	**1.25G**	**80.5**	**-**
Shunted-T [52]	T	256^2	11.5M	2.1G	79.8	CVPR'22
EdgeViT-S [47]	T	256^2	11.1M	1.9G	81.0	ECCV'22
FastViT-SA12 [64]	T	256^2	10.9M	1.9G	80.6	ICCV'23
EfficientFormer-L1 [33]	T	224^2	12.3M	1.3G	79.1	NeurIPS'22
MpViT-XS [31]	T	224^2	10.5M	2.9G	80.9	CVPR'22
EfficientNet-B3 [59]	C	300^2	12.0M	1.8G	81.6	ICCV'19
UniConvNet-N0(ours)	**C**	**224^2**	**10.2M**	**1.65G**	**81.6**	**-**
PVTv1-Tiny [67]	T	224^2	13.2M	1.9G	75.1	ICCV'21
RepViT-M1.5 [66]	C	224^2	14.0M	2.3G	81.2	CVPR'24
UniConvNet-N1(ours)	**C**	**224^2**	**13.1M**	**1.88G**	**82.2**	**-**
FastViT-SA24 [64]	T	256^2	20.6M	3.8G	82.6	ICCV'23
Dilate-T [29]	T	224^2	17.0M	3.2G	82.1	TMM'23
RepViT-M2.3 [66]	C	224^2	22.9M	4.5G	82.5	CVPR'24
UniRepLKNet-N [18]	C	224^2	18.3M	2.8G	81.6	CVPR'24
UniConvNet-N2(ours)	**C**	**224^2**	**15.0M**	**2.47G**	**82.7**	**-**
XCiT-S12/16 [1]	T	224^2	26.0M	4.8G	82.0	NeurIPS'21
ViTAE-S [73]	T	224^2	23.6M	5.6G	82.0	NeurIPS'21
CoAtNet-0 [13]	T	224^2	25.0M	4.0G	81.6	NeurIPS'21
PVTv2-B2-Li [68]	T	224^2	22.6M	3.9G	82.1	CVM'22
HorNet-T [50]	C	224^2	23.0M	4.0G	83.0	NeurIPS'22
UniConvNet-N3(ours)	**C**	**224^2**	**19.7M**	**3.37G**	**83.2**	**-**

Table 1. **Image classification performance on the ImageNet validation set for lightweight variants.** "Type" refers to model type, where "C" and "T" denote pure CNN and the model using the Transformer for its architecture, respectively. "Scale" is the input image scale. "Acc" is the TOP-1 accuracy.

3. Experiments

We develop different UniConvNet variants to match the complexities of various contemporary models, including state-of-the-art lightweight networks [43, 64, 66, 81] and large-scale networks [18, 38, 40, 69, 72, 74]. We evaluate the performance of UniConvNet variants and compare them to leading CNNs and ViTs across representative vision tasks, including image classification, object detection, and instance and semantic segmentation.

Model	Type	Scale	#Params	FLOPs	Acc(%)	Publication
CoAtNet-1 [13]	T	224^2	42.0M	8.0G	83.3	NeurIPS'21
EfficientFormer-L3 [33]	T	224^2	31.3M	3.9G	82.4	NeurIPS'22
Swin-T [38]	T	224^2	29.0M	5.0G	81.3	ICCV'21
Focal-T [75]	T	224^2	29.0M	5.0G	82.2	NeurIPS'21
CrossViT-18 [4]	T	224^2	28.2M	6.1G	82.3	ICCV'21
UniRepLKNet-T [18]	C	224^2	31.0M	4.9G	83.2	CVPR'24
SLaK-T [37]	C	224^2	30.0M	5.0G	82.5	ICLR'23
InternImage-T [69]	C	224^2	30.0M	5.0G	83.5	CVPR'23
FlashInternImage-T [72]	C	224^2	30.0M	-	83.6	CVPR'24
PeLK-T [5]	C	224^2	29.0M	5.6G	82.6	CVPR'24
ConvNeXt-T [40]	C	224^2	29.0M	5.0G	82.1	CVPR'22
MogaNet-S [32]	C	224^2	25.0M	5.0G	83.4	ICLR'24
WTConvNeXt-T [20]	-	224^2	30.0M	4.5G	82.5	ECCV'24
UniConvNet-T(ours)	**C**	**224^2**	**30.3M**	**5.1G**	**84.2**	**-**
SwinV2-S/8 [39]	T	256^2	50.0M	12.0G	83.7	CVPR'22
CoAtNet-2 [13]	T	224^2	75.0M	16.0G	84.1	NeurIPS'21
PVTv2-B4 [68]	T	224^2	63.0M	10.0G	83.6	CVM'22
Swin-S [38]	T	224^2	50.0M	9.0G	83.0	ICCV'21
RepLKNet-31B [17]	C	224^2	79.0M	15.0G	83.5	CVPR'22
PeLK-S [5]	C	224^2	50.0M	10.7G	83.9	CVPR'24
SLaK-S [37]	C	224^2	55.0M	10.0G	83.8	ICLR'23
ConvNeXt-S [40]	C	224^2	50.0M	9.0G	83.1	CVPR'22
HorNet-S [50]	C	224^2	50.0M	9.0G	84.0	NeurIPS'22
InternImage-S [69]	C	224^2	50.0M	8.0G	84.2	CVPR'23
FlashInternImage-S [72]	C	224^2	50.0M	-	84.4	CVPR'24
WTConvNeXt-S [20]	-	224^2	54.0M	8.8G	83.6	ECCV'24
UniConvNet-S(ours)	**C**	**224^2**	**50.0M**	**8.48G**	**84.5**	**-**
SwinV2-B/8 [39]	T	256^2	88.0M	20.0G	84.2	CVPR'22
Swin-B [38]	T	224^2	88.0M	15.0G	83.5	ICCV'21
PVTv2-B5 [68]	T	224^2	82.0M	12.0G	83.8	CVM'22
ConvNeXt-B [40]	C	224^2	88.0M	15.0G	83.8	CVPR'22
InternImage-B [69]	C	224^2	97.0M	16.0G	84.9	CVPR'23
FlashInternImage-B [72]	C	224^2	97.0M	-	84.9	CVPR'24
SLaK-B [37]	C	224^2	95.0M	17.0G	84.0	ICLR'23
PeLK-B [5]	C	224^2	89.0M	18.3G	84.2	CVPR'24
HorNet-B [50]	C	224^2	88.0M	16.0G	84.3	NeurIPS'22
WTConvNeXt-B [20]	-	224^2	93.0M	15.5G	84.1	ECCV'24
UniConvNet-B(ours)	**C**	**224^2**	**97.6M**	**15.9G**	**85.0**	**-**
Swin-B [38]	T	384^2	88.0M	47.1G	84.5	ICCV'21
ConvNeXt-B [40]	C	384^2	89.0M	45.0G	85.1	CVPR'22
UniConvNet-T(ours)	**C**	**384^2**	**30.3M**	**15.0G**	**85.4**	**-**
ConvNeXt-L [40]	C	384^2	198.0M	101.0G	85.5	CVPR'22
PeLK-B [5]	C	384^2	89.0M	54.0G	85.6	CVPR'24
UniConvNet-S(ours)	**C**	**384^2**	**50.0M**	**24.9G**	**85.7**	**-**
PeLK-B-101 [5]	C	384^2	90.0M	68.3G	85.8	CVPR'24
UniConvNet-B(ours)	**C**	**384^2**	**97.6M**	**46.6G**	**85.9**	**-**
SwinV2-L/24†[39]	T	384^2	197.0M	115.0G	87.6	CVPR'22
Swin-L†[38]	T	384^2	197.0M	104.0G	87.3	ICCV'21
MOAT-3†[74]	T	384^2	190.0M	141.2G	88.2	ICLR'23
HorNet-L†[50]	C	384^2	202.0M	102.0G	87.7	NeurIPS'22
ConvNeXt-L†[40]	C	384^2	198.0M	101.0G	87.5	CVPR'22
MogaNet-XL [32]	C	384^2	181.0M	102.0G	87.8	ICLR'24
RepLKNet-31L†[17]	C	384^2	172.0M	96.0G	86.6	CVPR'22
UniConvNet-L†(ours)	**C**	**384^2**	**201.8M**	**100.1G**	**88.2**	**-**
CoAtNet-4†[13]	T	384^2	275.0M	190.0G	87.9	NeurIPS'21
ConvNeXt-XL†[40]	C	384^2	350.0M	179.0G	87.7	CVPR'22
InternImage-XL†[69]	C	384^2	335.0M	163.0G	88.0	CVPR'23
FlashInternImage-L†[72]	C	384^2	224.0M	-	88.1	CVPR'24
InternImage-L†[69]	C	384^2	223.0M	108.0G	87.7	CVPR'23
UniConvNet-XL†(ours)	**C**	**384^2**	**226.7M**	**115.2G**	**88.4**	**-**

Table 2. **Image classification performance on the ImageNet validation set for scaled-up variants.** "Type" refers to model type, where "C" and "T" denote pure CNN and the model using the Transformer for its architecture, respectively. "Scale" is the input image scale. "Acc" is the TOP-1 accuracy. "†" indicates the model is pre-trained on ImageNet-22K[14].

3.1. Image Classification

For a fair comparison, in line with common practices [40, 72], UniConvNet-A/P0/P1/P2/N0/N1/N2/N3/T/S/B are trained on ImageNet-1K for 300 epochs, while UniConvNet-L/XL is first trained on ImageNet-22K for 90 epochs and then fine-tuned on ImageNet-1K for 20 epochs. Detailed ImageNet-1K/22K training settings, ImageNet-1K fine-tune settings and training recipes for different variants are presented in Appendix D.1, D.2 and D.3, respectively.

heavy RetinaNet

Backbone	Type	#Params	FLOPs	mAP	mAP_{50}	mAP_{75}	mAP_S	mAP_M	mAP_L
EdgeViT-XXS [47]	T	13.1M	-	38.7	59.0	41.0	22.4	42.0	51.6
EMO-5M [81]	T	14.4M	-	38.9	59.8	41.0	23.8	42.2	51.7
UniConvNet-A(ours)	**C**	**12.6M**	**16.3G**	**40.0**	**60.4**	**42.9**	**23.3**	**44.2**	**53.2**
Mobile-Former-151M [8]	T	14.4M	-	34.2	53.4	36.0	19.9	36.8	45.3
Mobile-Former-214M [8]	T	15.2M	-	35.8	55.4	38.0	21.8	38.5	46.8
Mobile-Former-294M [8]	T	16.1M	-	36.6	56.6	38.6	21.9	39.5	47.9
EdgeViT-XS [47]	T	16.3M	-	40.6	61.3	43.3	25.2	43.9	54.6
UniConvNet-P0(ours)	**C**	**14.4M**	**16.8G**	**41.1**	**61.4**	**43.9**	**24.2**	**45.2**	**55.2**
Mobile-Former-508M [8]	T	17.9M	-	38.0	58.3	40.3	22.9	41.2	49.7
UniConvNet-P2(ours)	**C**	**16.9M**	**16.8G**	**42.2**	**62.5**	**45.2**	**25.1**	**45.9**	**56.2**
PVTv1-Tiny [67]	T	23.0M	-	36.7	56.9	38.9	22.6	38.8	50.0
ResNet-18 [23]	C	21.3M	-	31.8	49.6	33.6	16.3	34.3	43.2
UniConvNet-N0(ours)	**C**	**20.7M**	**18.8G**	**42.8**	**63.2**	**45.7**	**25.2**	**47.0**	**56.8**
PVTv2-B1 [68]	T	23.8M	-	41.2	61.9	43.9	25.4	44.5	54.3
UniConvNet-N1(ours)	**C**	**23.9M**	**19.4G**	**44.6**	**65.7**	**47.9**	**27.9**	**49.0**	**59.2**
PVTv1-Small [67]	T	44.1M	-	40.4	61.3	43.0	25.0	42.9	55.7
MPViT-T [31]	T	28.0M	-	41.8	62.7	44.6	27.2	45.1	54.2
Twins-PCPVT-S [10]	T	34.4M	-	43.0	64.1	46.0	27.5	46.3	57.3
PVTv2-B2 [68]	T	35.1M	-	44.6	65.6	47.6	27.4	48.8	58.6
Shunted-S [52]	T	32.1M	-	45.4	65.9	49.2	28.7	49.3	60.0
UniConvNet-N2(ours)	**C**	**26.0M**	**20.7G**	**45.5**	**66.4**	**48.9**	**28.4**	**50.2**	**60.4**

light SSDLite

Backbone	Type	#Params	FLOPs	mAP	mAP_{50}	mAP_{75}	mAP_S	mAP_M	mAP_L
MobileNetv3 [26]	C	5.0M	0.6G	22.0	-	-	-	-	-
MobileNetv2 [53]	C	4.3M	0.8G	22.1	-	-	-	-	-
MobileNetv1 [27]	C	5.1M	1.3G	22.2	-	-	-	-	-
MixNet [60]	T	4.5M	-	22.3	-	-	-	-	-
MNASNet [61]	T	4.9M	0.8G	23.0	-	-	3.8	21.7	42.0
MobileViTv1-Small [45]	T	5.7M	3.4G	27.7	-	-	-	-	-
EdgeNeXt-S [44]	T	6.2M	2.1G	27.9	-	-	-	-	-
UniConvNet-A(ours)	**C**	**4.4M**	**1.3G**	**29.5**	**46.7**	**30.2**	**5.3**	**31.8**	**53.6**
MobileViTv2-1.25 [46]	T	8.2M	4.7G	27.8	-	-	-	-	-
EMO-5M [81]	T	6.0M	1.8G	27.8	45.2	28.2	5.2	30.5	50.0
UniConvNet-P0(ours)	**C**	**6.4M**	**1.8G**	**30.5**	**48.1**	**31.1**	**6.2**	**33.6**	**55.3**
UniConvNet-P2(ours)	**C**	**8.6M**	**2.6G**	**32.0**	**49.8**	**33.0**	**7.6**	**35.8**	**56.0**
MobileViTv2-1.75 [46]	T	14.9M	9.0G	29.5	-	-	-	-	-
UniConvNet-N0(ours)	**C**	**12.2M**	**3.8G**	**33.7**	**52.1**	**34.8**	**9.4**	**37.6**	**58.5**

Table 3. **Object detection performance by heavy RetinaNet and light SSDLite on COCO val2017.** "Type" refers to model type, where "C" and "T" denote pure CNN and the model using the Transformer for its architecture, respectively. The FLOPs are measured with 320×320 inputs.

As is shown in Tab. 1 and Tab. 2, our proposed model variants demonstrate significant improvements over state-of-the-art models, effectively bridging the gap between lightweight and large-scale models. Existing models either exhibit inferior performance in lightweight scenarios or fail to achieve adequate accuracy when scaled up.

3.2. Downstream Tasks

Object Detection and Instance Segmentation on COCO We fine-tune the heavy RetinaNet [35] and light SSDLite [26] using our ImageNet-1K pre-trained UniConvNet on the MS-COCO 2017 with a $1\times$ schedule training recipe. We further fine-tune scaled-up UniConvNet using the representative object detection framework Mask R-CNN [24] and Cascade Mask R-CNN [3] on the MS-COCO 2017 datasets, employing either a $1\times$ (12epoch) training schedule or a $3\times$ (36 epoch) training schedule. Detailed fine-tuning settings are provided in Appendix D.4.

Semantic Segmentation on ADE20K We fine-tune DeepLabv3 [7] and PSPNet [83] using our ImageNet-1K pre-trained UniConvNet on the ADE20K dataset with a training recipe of 160k iterations. We also fine-tune the scaled-up UniConvNet using the representative semantic segmentation framework UperNet on the ADE20K dataset for 160k iterations. Detailed fine-tuning settings are provided in Appendix D.5.

Backbone	Type	#Params	FLOPs	Mask R-CNN 1× schedule						Mask R-CNN 3× + MS schedule					
				AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
UniConvNet-N2	**C**	**34.7M**	**220G**	**46.6**	**68.0**	**51.3**	**41.9**	**65.1**	**45.2**	**48.4**	**69.7**	**53.2**	**43.2**	**66.7**	**46.4**
PVTv2-B2 [68]	T	45.0M	309G	45.3	67.1	49.6	41.2	64.2	44.4	47.8	69.7	52.6	43.1	66.8	46.7
ViT-Adapter-S [9]	T	48.0M	403G	44.7	65.8	48.3	39.9	62.5	42.8	48.2	69.7	52.5	42.8	66.4	45.9
MogaNet-S [32]	C	45.0M	272G	46.7	-	-	42.2	-	-	-	-	-	-	-	-
UniConvNet-N3	**C**	**39.4M**	**239G**	**47.0**	**68.6**	**51.8**	**42.4**	**65.6**	**45.7**	**49.4**	**70.7**	**54.4**	**44.2**	**67.9**	**47.5**
Swin-T [38]	T	48.0M	267G	42.7	65.2	46.8	39.3	62.2	42.2	46.0	68.1	50.3	41.6	65.1	44.9
ConvNeXt-T [40]	C	48.0M	262G	44.2	66.6	48.3	40.1	63.3	42.8	46.2	67.9	50.8	41.7	65.0	44.9
InternImage-T [69]	C	49.0M	270G	47.2	69.0	52.1	42.5	66.1	45.8	49.1	70.4	54.1	43.7	67.3	47.3
MogaNet-B [32]	C	63.0M	373G	47.9	-	-	43.2	-	-	-	-	-	-	-	-
FlashInternImage-T [72]	C	49.0M	-	48.0	-	-	43.1	-	-	49.5	-	-	44.0	-	-
UniConvNet-T	**C**	**50.0M**	**265G**	**48.2**	**69.8**	**52.9**	**43.3**	**66.6**	**45.7**	**50.1**	**71.0**	**54.8**	**44.5**	**68.4**	**48.0**
Swin-S [38]	T	69.0M	354G	44.8	66.6	48.9	40.9	63.4	44.2	48.2	69.8	52.8	43.2	67.0	46.1
ConvNeXt-S [40]	C	70.0M	348G	45.4	67.9	50.0	41.8	65.2	45.1	47.9	70.0	52.7	42.9	66.9	46.2
InternImage-S [69]	C	69.0M	340G	47.8	69.8	52.8	43.3	67.1	46.7	49.7	71.1	54.5	44.5	68.5	47.8
FlashInternImage-S [72]	C	69.0M	-	49.2	-	-	44.0	-	-	50.5	-	-	44.9	-	-
UniConvNet-S	**C**	**70.0M**	**336G**	**48.8**	**70.4**	**53.4**	**43.8**	**67.4**	**47.3**	**50.8**	**71.6**	**55.6**	**45.2**	**69.3**	**48.9**
Swin-B [38]	T	107.0M	496G	46.9	-	-	42.3	-	-	48.6	70.0	53.4	43.3	67.1	46.7
PVTv2-B5 [68]	T	102.0M	557G	47.4	68.6	51.9	42.5	65.7	46.0	48.4	69.2	52.9	42.9	66.6	46.2
ConvNeXt-B [40]	C	108.0M	486G	47.0	69.4	51.7	42.7	66.3	46.0	48.5	70.1	53.3	43.5	67.1	46.7
InternImage-B [69]	C	115.0M	501G	48.8	70.9	54.0	44.0	67.8	47.4	50.3	71.4	55.3	44.8	68.7	48.0
FlashInternImage-B [72]	C	115.0M	-	50.1	-	-	44.5	-	-	50.6	-	-	45.4	-	-
UniConvNet-B	**C**	**118.0M**	**498G**	**50.0**	**71.7**	**55.3**	**45.0**	**69.0**	**48.5**	**51.2**	**72.2**	**56.1**	**45.6**	**69.6**	**49.2**

Backbone	Type	#Params	FLOPs	Cascade Mask R-CNN 1× schedule						Cascade Mask R-CNN 3× + MS schedule					
Swin-L†[38]	T	253.0M	1382G	51.8	71.0	56.2	44.9	68.4	48.9	53.9	72.4	58.8	46.7	70.1	50.8
RepLKNet-31L†[17]	C	229.0M	1321G	-	-	-	-	-	-	53.9	72.5	58.6	46.5	70.0	50.6
ConvNeXt-L†[40]	C	255.0M	1354G	53.5	72.8	58.3	46.4	70.2	50.2	54.8	73.8	59.8	47.6	71.3	51.7
ConvNeXt-XL†[40]	C	407.0M	1898G	53.6	72.9	58.5	46.5	70.3	50.5	55.2	74.2	59.9	47.7	71.6	52.2
HorNet-L†[50]	C	259.0M	1358G	-	-	-	-	-	-	56.0	-	-	48.6	-	-
InternImage-L†[69]	C	277.0M	1399G	54.9	74.0	59.8	47.7	71.4	52.1	56.1	74.8	60.7	48.5	72.4	53.0
InternImage-XL†[69]	C	387.0M	1782G	55.3	74.4	60.1	48.1	71.9	52.4	56.2	75.0	61.2	48.8	72.5	53.4
FlashInternImage-L†[72]	C	277.0M	-	55.6	-	-	48.2	-	-	56.7	-	-	48.9	-	-
UniConvNet-L†	**C**	**254.8M**	**1288G**	**55.7**	**74.4**	**60.4**	**48.3**	**71.9**	**52.9**	**56.6**	**75.6**	**61.8**	**48.9**	**73.0**	**53.4**

Table 4. **Object detection and instance segmentation performance by Mask R-CNN and Cascade Mask R-CNN on COCO val2017.** "Type" refers to model type, where "C" and "T" denote pure CNN and the model using the Transformer for its architecture, respectively. The FLOPs are measured with 1280×800 inputs.

DeepLabv3

Model	Type	#Params	FLOPs	mIoU
MobileViTv2-0.5 [46]	T	6.3M	26.1G	31.9
EMO-2M [81]	T	6.9M	3.5G	35.3
UniConvNet-A(ours)	**C**	**7.9M**	**4.2G**	**38.2**
MobileViTv2-0.75 [46]	T	9.6M	40.0G	34.7
EMO-5M [81]	T	10.3M	5.8G	37.8
UniConvNet-P0(ours)	**C**	**9.7M**	**5.3G**	**39.7**
MobileViTv2-1.0 [46]	T	13.4M	56.4G	37.0
UniConvNet-P2(ours)	**C**	**12.6M**	**7.8G**	**40.0**
MobileNetv2 [53]	C	18.7M	75.4G	34.1
UniConvNet-N0(ours)	**C**	**17.2M**	**11.0G**	**41.0**
UniConvNet-N1(ours)	**C**	**20.3M**	**12.3G**	**42.1**
MobileViTv2-2.0 [46]	T	64.0M	147.0G	40.9
ResNet-50 [23]	C	68.2M	270.3G	42.4
UniConvNet-N2(ours)	**C**	**22.5M**	**15.7G**	**42.9**

PSPNet

Model	Type	#Params	FLOPs	mIoU
EMO-2M [81]	T	5.5M	3.1G	34.5
MobileViTv2-0.75 [46]	T	6.2M	26.6G	35.2
UniConvNet-A(ours)	**C**	**6.5M**	**3.8G**	**37.9**
MobileViTv2-1.0 [46]	T	9.4M	40.3G	36.5
EMO-5M [81]	T	8.5M	5.3G	38.2
UniConvNet-P0(ours)	**C**	**8.1M**	**4.8G**	**39.2**
MobileNetv2 [53]	T	13.7M	53.1G	29.7
UniConvNet-P2(ours)	**C**	**10.9M**	**7.3G**	**39.6**
UniConvNet-N0(ours)	**C**	**15.0M**	**10.4G**	**40.1**
MobileViTv2-1.75 [46]	T	22.5M	95.9G	39.8
UniConvNet-N1(ours)	**C**	**18.2M**	**11.6G**	**42.1**
ResNet-50 [23]	T	49.1M	179.1G	41.1
UniConvNet-N2(ours)	**C**	**20.3M**	**15.1G**	**42.5**

Table 5. **Semantic segmentation performance by DeepLabv3 and PSPNet on ADE20K dataset.** "Type" refers to model type, where "C" and "T" denote pure CNN and the model using the Transformer for its architecture, respectively. The FLOPs are measured with 512×512 inputs.

Backbone	Type	Crop Size	#Params	FLOPs	mIoU(SS)	mIoU(MS)
Swin-T [38]	T	512^2	60.0	939	44.5	45.8
ConvNeXt-T [40]	C	512^2	60.0	939	46.0	46.7
UniConvNet-N2	**C**	**512^2**	**43.9**	**893**	**47.9**	**48.9**
SLaK-T [37]	C	512^2	65.0	936	47.6	-
InternImage-T [69]	C	512^2	59.0	944	47.9	48.1
UniConvNet-N3	**C**	**512^2**	**48.6**	**912**	**49.0**	**50.0**
Shunted-S [52]	T	512^2	52.0	940	48.9	49.9
PeLK-T [5]	C	512^2	62.0	970	48.1	-
UniRepLKNet-T [18]	C	512^2	61.0	946	48.6	49.1
FlashInternImage-T [72]	C	512^2	59.0	944	49.3	50.3
MogaNet-B [32]	C	512^2	74.0	1050	50.1	-
UniConvNet-T	**C**	**512^2**	**59.2**	**939**	**50.3**	**51.2**
Swin-S [38]	T	512^2	81.0	1038	47.6	49.5
ConvNeXt-S [40]	C	512^2	82.0	7027	48.7	49.6
SLaK-S [37]	C	512^2	91.0	1028	49.4	-
PeLK-S [5]	C	512^2	84.0	1077	49.7	-
InternImage-S [69]	C	512^2	80.0	1017	50.1	50.9
UniRepLKNet-S [18]	C	512^2	86.0	1036	50.5	51.0
FlashInternImage-S [69]	C	512^2	80.0	-	50.6	51.6
MogaNet-L [32]	C	512^2	113.0	1176	50.9	-
UniConvNet-S	**C**	**512^2**	**78.9**	**1015**	**52.2**	**52.8**
Swin-B [38]	T	512^2	121.0	1188	48.1	49.7
ConvNeXt-B [40]	C	512^2	122.0	1170	49.1	49.9
RepLKNet-31B [17]	C	512^2	112.0	1170	49.9	50.6
SLaK-B [37]	C	512^2	135.0	1172	50.2	-
PeLK-B-101 [5]	C	512^2	126.0	1339	50.6	-
InternImage-B [69]	C	512^2	128.0	1185	50.8	51.3
FlashInternImage-B [69]	C	512^2	128.0	-	52.0	52.6
UniConvNet-B	**C**	**512^2**	**126.5**	**1179**	**52.3**	**52.9**
Swin-L†[38]	T	640^2	234	2468	52.1	53.5
RepLKNet-31L†[17]	C	640^2	207	2404	52.4	52.7
ConvNeXt-L†[40]	C	640^2	235	2458	53.2	53.7
ConvNeXt-XL†[40]	C	640^2	391	3335	53.6	54.0
InternImage-L†[69]	C	640^2	256	2526	53.9	54.1
InternImage-XL†[69]	C	640^2	368	3142	55.0	55.3
UniConvNet-L†	**C**	**640^2**	**234**	**2310**	**55.1**	**55.4**

Table 6. **Semantic segmentation performance by UperNet on ADE20K validation set.** "Type" refers to model type, where "C" and "T" denote pure CNN and the model using the Transformer for its architecture, respectively. "SS" and "MS" denote single-scale and multi-scale testing, respectively. The FLOPs are measured with 512×2048 or 640×2560 inputs.

Overall Results As shown in Tab. 3, Tab. 4, Tab. 5 and Tab. 6, our proposed UniConvNet significantly enhances

Model	N	Kernel Size	#Params	FLOPs	Acc(%)
	3	5, 7, 9	3.5M	0.564G	76.6
UniConvNet-A	3	7, 9, 11	3.4M	0.589G	77.0
	3	9, 11, 13	3.5M	0.579G	76.9
	3	5, 7, 9	5.1M	0.810G	78.8
UniConvNet-P0	3	7, 9, 11	5.2M	0.832G	79.1
	3	9, 11, 13	5.3M	0.868G	79.3
	3	11, 13, 15	5.1M	0.845G	78.8
	3	5, 7, 9	30.0M	5.0G	84.1
UniConvNet-T	3	7, 9, 11	30.3M	5.1G	84.2
	3	9, 11, 13	29.6M	5.0G	84.1
UniConvNet-N0	3	7, 9, 11	10.2M	1.65G	81.6
	4	5, 7, 9, 11	9.8M	1.64G	81.5
	4	7, 9, 11, 13	10.0M	1.70G	81.3
	3	27, 29, 31	14.9M	2.87G	81.8
UniConvNet-N2	3	7, 9, 11	15.0M	2.47G	82.7
	3	7, 11, 31	3.56M	0.687G	76.0
	3	7, 9, 29	3.45M	0.647G	75.9
UniConvNet-A	3	7, 9, 11	3.4M	0.589G	77.0
	3	7, 11, 31	6.2M	1.162G	79.4
UniConvNet-P1	3	7, 9, 11	6.1M	0.895G	79.6

Table 7. **Ablation studies on layer number and progressive kernel size.** "Acc" is the TOP-1 accuracy. "N" represents the number in the pyramid order.

performance compared to state-of-the-art models, offering lighter parameters and reduced FLOPs. This demonstrates the effectiveness and efficiency of the proposed Three-layer RFA and its improved capability for downstream tasks.

4. Analysis

4.1. Three-layer RFA properly expand ERF while maintaining AGD

Ablation Studies on Layer Number and Kernel Size Section 2.3 presents the constraints on layer N and progressive kernel size. Based on UniConvNet-T, we adjust layer N and progressive kernel size for ablation studies. The results indicate that the proposed principles for these two hyperparameters are effective and accurate. As illustrated in Tab. 7, for layer 3, progressive kernel sizes smaller than $(5, 7, 9)$ may be insufficient for expanding ERF to a level of existing large-kernel ConvNets and result in inferior performance. Progressive kernel sizes larger than $(9, 11, 13)$ achieve better TOP-1 accuracy than UniConvNet-P0, with slightly higher parameters and FLOPs. In contrast, UniConvNet-A and UniConvNet-T perform better with kernel sizes of $(7, 9, 11)$. Progressive kernel sizes of $(9, 11, 13)$ are inefficient for constructing deep models with equivalent parameters, which is crucial for model perception. Therefore, we choose a kernel size of $(7, 9, 11)$ for efficiency. For layer 4, progressive kernels result in a theoretical receptive field much larger than the image size of 14×14 in stage 3, which is wasteful for an image resolution of 224×224. It contradicts the original goal of alleviating parameters and FLOPs burden in contemporary large-kernel ConvNets.

Investigation on different AGD of ERF We also examine the perceptual capabilities of large kernel sizes such as $(27, 29, 31)$ as used in RepLKNet [17]. The performance results show that using large kernels is neither efficient nor effective for constructing long-range ERF following AGD, disrupting AGD on small-scale pixels. We further use relatively small kernels in the first two layers, then use an extremely large kernel in the latter layer, such as $(7, 9, 29)$ and $(7, 11, 31)$. This establishes a small-scale AGD in a smaller area, compared to the kernel sizes of $(7, 9, 11)$, and expands the ERF by the latter extremely large kernel. The inferior performance demonstrates that a large ERF with continuous AGD, generated by three-layer RFA, from center to edge is vital and proper. We analyze the ERF of several models and demonstrate that a proper AGD of small-scale pixels is more important than expanding the ERF. Please refer to Appendix A for detailed analysis.

UniConvNet	Overall	RFA^{3-l}	Modified DCNV3	Feed-Forward	Class Head
UniConvNet-A	3.4/0.589	**0.747/0.157**	**0.91/0.161**	1.331/0.234	0.409/0.037
UniConvNet-P0	5.17/0.832	**1.06/0.205**	**1.41/0.229**	2.06/0.334	0.64/0.066
UniConvNet-P1	6.1/0.895	**1.26/0.22**	**1.70/0.247**	2.495/0.362	0.635/0.064
UniConvNet-P2	7.57/1.25	**1.625/0.315**	**2.16/0.354**	3.153/0.516	0.648/0.065
UniConvNet-N0	10.23/1.65	**2.18/0.405**	**2.16/0.354**	4.62/0.851	1.27/0.04
UniConvNet-N1	13.06/1.88	**2.85/0.458**	**2.85/0.409**	6.095/0.867	1.275/0.134
UniConvNet-N2	15.0/2.47	**3.34/0.622**	**3.31/0.546**	7.08/1.161	1.27/0.139
UniConvNet-N3	19.7/3.37	**4.54/0.849**	**4.45/0.762**	9.48/1.619	1.25/0.142
UniConvNet-T	30.3/5.1	**5.41/1.06**	**5.74/1.0**	14.61/2.46	4.54/0.58
UniConvNet-S	50/8.48	**4.54/0.849**	**4.45/0.762**	9.48/1.619	1.25/0.142
UniConvNet-B	97.6/15.9	**10.89/1.97**	**11.44/1.95**	24.61/4.18	3.06/0.38
UniConvNet-L	201.8/100.1	**22.01/21.79**	**25.77/23.48**	80.24/50.56	11.11/4.28
UniConvNet-XL	226.7/115.2	**24.7/25.07**	**28.95/27.02**	90.14/58.19	12.48/4.93

Table 8. **#Parameters and FLOPs distributions of UniConvNet variants.** #Parameters(M)/FLOPs(G) are #parameters and FLOPs for each block, respectively (*e.g.*, 3.4/0.589 are the overall #parameters and FLOPs for UniConvNet-A, respectively).

4.2. Three-layer RFA is Efficient and Effective

Efficiency We present the parameters and FLOPs of UniConvNet variants in Tab. 8. UniConvNet comprises Three-layer RFA, modified DCNV3, a feed-forward layer, and a class head. Generally, compared to small-kernel modified DCNV3, Three-layer RFA has fewer or comparable parameters and FLOPs. This suggests that our proposed Three-layer RFA can establish long-range dependencies while reducing FLOP costs and enhancing parameter efficiency.

Effectiveness We perform an ablation study comparing various combinations of large-kernel and small-kernel convolutions to evaluate the effectiveness of different modules. As illustrated in Tab. 9, models using only Three-layer RFA or Modified DCNV3 achieve similar top-1 accuracies of 78.4 and 78.5, respectively. This indicates that the proposed Three-layer RFA retains comparable feature perception capabilities without relying on the basic small-scale information typically used in conventional ConvNets. Additionally, we replace Modified DCNV3 in UniConvNet-P0 with a depth-wise 3×3 convolution. The top-1 accu-

Model	#Params	Large Kernel	Small Kernel	Acc(%)
UniConvNet-P0	5.2M	Three-layer RFA	Modified DCNV3	79.1
	5.2M	Three-layer RFA	✗	78.4
FlashInternImage [72]	5.3M	✗	DCNV4	78.5
	5.1M	Three-layer RFA	DW 3×3	78.9
ConvNeXt [40]	5.2M	DW 7×7	DW 3×3	77.0
	5.3M	✗	DW 3×3	77.0
UniConvNet-N2	15.0M	Three-layer RFA	Modified DCNV3	82.7
FlashInternImage [72]	15.3M	✗	DCNV4	82.2
	14.9M	Three-layer RFA	DW 3×3	81.9
ConvNeXt [40]	15.1M	DW 7×7	DW 3×3	81.0
UniConvNet-T	30.3M	Three-layer RFA	Modified DCNV3	84.2
FlashInternImage-T [72]	30.0M	✗	DCNV4	83.6
	30.7M	Three-layer RFA	DW 3×3	83.7
ConvNeXt-T [40]	29.0M	DW 7×7	DW 3×3	82.1

Table 9. **Ablation comparisons of different large-kernel and small-kernel convolutions.** "Acc" is the TOP-1 accuracy.

Model	#Params	Scale	Acc(%)	Throughput
Swin-T [38]	29.0M	224^2	81.3	1989/3619
ConvNeXt-T [40]	29.0M	224^2	82.1	2485/4305
InternImage-T [69]	30.0M	224^2	83.5	1409/1746
UniConvNet-T	30.3M	224^2	84.2	1480/1825
ConvNeXt-XL [40]	350.0M	224^2	87.7	170/299
InternImage-XL [69]	335.0M	224^2	88.0	125/174
InternImage-L [69]	223.0M	224^2	87.7	158/214
UniConvNet-XL	226.7M	224^2	88.4	168/228

Table 10. **Image classification throughput on ImageNet-1K.** "Acc(%)" is the TOP-1 Accuracy. The overall throughput of each model, measured as the number of images processed per second, is reported in FP32/FP16 data formats.

racy is 78.9, 0.2 lower than UniConvNet-P0, indicating that UniConvNet-P0 is slightly improved by Modified DCNV3 compared to when it uses simple depth-wise 3×3 convolution. In contrast, replacing Three-layer RFA with another large-kernel convolution (depth-wise 7×7 convolution) results in a drop in top-1 accuracy to 77.0, highlighting the effectiveness of our proposed Three-layer RFA. Furthermore, models using only depth-wise 3×3 convolution achieve a top-1 accuracy of 77.0, indicating that depth-wise 7×7 convolution does not enhance the model's perception capabilities. This further validates the effectiveness of the proposed Three-layer RFA. The ablation studies on models of $15M$ and $30M$ parameters manifest a consistent performance improvement ability of the proposed Three-layer RFA.

4.3. Throughput analysis

We benchmark throughput on several classic and relevant models. As shown in Tab. 10, UniConvNets improve the throughput compared to InternImage [69] with similar 3×3 convolution. Please refer to Appendix B for detailed configurations.

5. Related Work

Convolutional neural networks (ConvNets) [16, 21, 23, 26, 27, 43, 51, 53, 54, 69, 71, 72, 78, 80] have long been the standard architecture for vision recognition due to their intrinsic inductive bias. CNNs utilize a stack of small kernels to establish local dependencies, which limits their ability to perceive. However, their dominance is being challenged by the emergence of attention mechanisms in computer vision. In recent years, attention-based models [1, 4, 8, 9, 19, 29, 31, 33, 38, 44, 52, 64, 67, 68, 76] have gradually become essential for computer vision tasks due to their ability to establish global perception by building long-range dependencies through self-attention.

Inspired by this, large kernels are increasingly recognized for their ability to establish long-range dependencies in CNNs, thereby improving the accuracy of various vision tasks. The use of large kernels in convolutional networks originated with AlexNet [30] and Inception [55, 56, 58], which utilized 7×7 or 11×11 kernels in the low-level layers. ConvNeXt [40] examines the feasibility of large-kernel convolution within conventional ResNet-like architectures, where performance saturates at a kernel size of 7×7.

Recently, RepLKNet [17] first scales the convolution kernel up to 31×31 using structural re-parameterization techniques and provides several guidelines for architectural design. VAN [22] acquires a 21×21 receptive field to establish long-term dependencies through sequential stacks of large-kernel depth-wise convolutions (DWConv) and depth-wise dilation convolutions. SLaK [37] constructs a pure CNN architecture using sparse factorized 51×51 kernels, consisting of parallel 51×5 and 5×51 kernels. MogaNet [32] employs a spatial aggregation block utilizing 5×5 and 7×7 convolutions to adaptively aggregate discriminative features. PeLK [5] employs a human-like peripheral convolution to reduce parameters via parameter sharing while scaling the kernel size to a substantial 101×101. However, recent large-kernel ConvNets suffer from parameters and FLOPs burden and disrupted AGD of ERF. In this work, we focus on expanding ERF while maintaining AGD for ConvNets of Any Scale.

6. Conclusion

We introduce a Receptive Field Aggregator (RFA) to expand effective receptive field (ERF) while maintaining the Asymptotically Gaussian Distribution (AGD) of ERF. Accordingly, we design a Three-layer RFA for input images with a resolution of 224×224, which can be a plug-and-play module for ConvNets or replace the convolutional layers within them. Based on these designs, we propose a **uni**versal **conv**olutional neural **net**work (ConvNet), termed UniConvNet, and evaluate its performance across a wide range of vision recognition tasks. All variants of UniConvNet demonstrate superior performance with reduced parameters and FLOPs. This work may draw attention to the design of large ERF following an AGD, enhancing the ConvNet of any scale.

References

[1] Alaaeldin Ali, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. Xcit: Cross-covariance image transformers. *Advances in neural information processing systems*, 34:20014–20027, 2021. 4, 8

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 2

[3] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6154–6162, 2018. 5

[4] Chun-Fu Richard Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 357–366, 2021. 5, 8

[5] Honghao Chen, Xiangxiang Chu, Yongjian Ren, Xin Zhao, and Kaiqi Huang. Pelk: Parameter-efficient large kernel convnets with peripheral convolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5557–5567, 2024. 1, 2, 5, 6, 8

[6] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 4

[7] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 5, 4

[8] Yinpeng Chen, Xiyang Dai, Dongdong Chen, Mengchen Liu, Xiaoyi Dong, Lu Yuan, and Zicheng Liu. Mobile-former: Bridging mobilenet and transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5270–5279, 2022. 5, 8

[9] Zhe Chen, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. Vision transformer adapter for dense predictions. *arXiv preprint arXiv:2205.08534*, 2022. 6, 8

[10] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Twins: Revisiting the design of spatial attention in vision transformers. *Advances in neural information processing systems*, 34:9355–9366, 2021. 5

[11] MMSegmentation Contributors. Mmsegmentation: Openmmlab semantic segmentation toolbox and benchmark, 2020. 4

[12] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020. 2

[13] Zihang Dai, Hanxiao Liu, Quoc V Le, and Mingxing Tan. Coatnet: Marrying convolution and attention for all data sizes. *Advances in neural information processing systems*, 34:3965–3977, 2021. 2, 4, 5

[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 5, 2

[15] Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Diverse branch block: Building a convolution as an inception-like unit. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10886–10895, 2021. 1

[16] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13733–13742, 2021. 8, 1

[17] Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11963–11975, 2022. 1, 2, 5, 6, 7, 8

[18] Xiaohan Ding, Yiyuan Zhang, Yixiao Ge, Sijie Zhao, Lin Song, Xiangyu Yue, and Ying Shan. Unireplknet: A universal perception large-kernel convnet for audio, video, point cloud, time-series and image recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024. 1, 4, 5, 6, 2

[19] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2021. 1, 8

[20] Shahaf E Finder, Roy Amoyal, Eran Treister, and Oren Freifeld. Wavelet convolutions for large receptive fields. *arXiv preprint arXiv:2407.05848*, 2024. 5

[21] Shang-Hua Gao, Ming-Ming Cheng, Kai Zhao, Xin-Yu Zhang, Ming-Hsuan Yang, and Philip Torr. Res2net: A new multi-scale backbone architecture. *IEEE transactions on pattern analysis and machine intelligence*, 43(2):652–662, 2019. 1, 8

[22] Meng-Hao Guo, Cheng-Ze Lu, Zheng-Ning Liu, Ming-Ming Cheng, and Shi-Min Hu. Visual attention network. *Computational Visual Media*, 9(4):733–752, 2023. 8

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 1, 5, 6, 8

[24] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 5

[25] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 2

[26] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international*

conference on computer vision, pages 1314–1324, 2019. 5, 8

[27] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 5, 8

[28] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 646–661. Springer, 2016. 2

[29] Jiayu Jiao, Yu-Ming Tang, Kun-Yu Lin, Yipeng Gao, Andy J Ma, Yaowei Wang, and Wei-Shi Zheng. Dilateformer: Multi-scale dilated transformer for visual recognition. *IEEE Transactions on Multimedia*, 25:8906–8919, 2023. 4, 8

[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012. 8

[31] Youngwan Lee, Jonghee Kim, Jeffrey Willette, and Sung Ju Hwang. Mpvit: Multi-path vision transformer for dense prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7287–7296, 2022. 4, 5, 8

[32] Siyuan Li, Zedong Wang, Zicheng Liu, Cheng Tan, Haitao Lin, Di Wu, Zhiyuan Chen, Jiangbin Zheng, and Stan Z Li. Moganet: Multi-order gated aggregation network. In *The Twelfth International Conference on Learning Representations*, 2023. 1, 2, 5, 6, 8

[33] Yanyu Li, Geng Yuan, Yang Wen, Ju Hu, Georgios Evangelidis, Sergey Tulyakov, Yanzhi Wang, and Jian Ren. Efficientformer: Vision transformers at mobilenet speed. *Advances in Neural Information Processing Systems*, 35: 12934–12949, 2022. 4, 5, 8

[34] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*, pages 740–755. Springer, 2014. 2

[35] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 5

[36] Weifeng Lin, Ziheng Wu, Jiayu Chen, Jun Huang, and Lianwen Jin. Scale-aware modulation meet transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6015–6026, 2023. 4

[37] Shiwei Liu, Tianlong Chen, Xiaohan Chen, Xuxi Chen, Qiao Xiao, Boqian Wu, Tommi Kärkkäinen, Mykola Pechenizkiy, Decebal Mocanu, and Zhangyang Wang. More convnets in the 2020s: Scaling up kernels beyond 51x51 using sparsity. *arXiv preprint arXiv:2207.03620*, 2022. 1, 2, 5, 6, 8

[38] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 1, 2, 4, 5, 6, 8

[39] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12009–12019, 2022. 2, 5

[40] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986, 2022. 2, 4, 5, 6, 8, 1

[41] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 4

[42] Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the effective receptive field in deep convolutional neural networks. *Advances in neural information processing systems*, 29, 2016. 2, 1

[43] Xu Ma, Xiyang Dai, Yue Bai, Yizhou Wang, and Yun Fu. Rewrite the stars. *arXiv preprint arXiv:2403.19967*, 2024. 4, 8

[44] Muhammad Maaz, Abdelrahman Shaker, Hisham Cholakkal, Salman Khan, Syed Waqas Zamir, Rao Muhammad Anwer, and Fahad Shahbaz Khan. Edgenext: efficiently amalgamated cnn-transformer architecture for mobile vision applications. In *European Conference on Computer Vision*, pages 3–20. Springer, 2022. 4, 5, 8

[45] Sachin Mehta and Mohammad Rastegari. Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. *arXiv preprint arXiv:2110.02178*, 2021. 1, 4, 5

[46] Sachin Mehta and Mohammad Rastegari. Separable self-attention for mobile vision transformers. *arXiv preprint arXiv:2206.02680*, 2022. 5, 6

[47] Junting Pan, Adrian Bulat, Fuwen Tan, Xiatian Zhu, Lukasz Dudziak, Hongsheng Li, Georgios Tzimiropoulos, and Brais Martinez. Edgevits: Competing light-weight cnns on mobile devices with vision transformers. In *European Conference on Computer Vision*, pages 294–311. Springer, 2022. 4, 5

[48] Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992. 2

[49] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10428–10436, 2020. 1

[50] Yongming Rao, Wenliang Zhao, Yansong Tang, Jie Zhou, Ser Nam Lim, and Jiwen Lu. Hornet: Efficient high-order spatial interactions with recursive gated convolutions. *Advances in Neural Information Processing Systems*, 35: 10353–10366, 2022. 2, 4, 5, 6

[51] Muhammad Imran Razzak, Saeeda Naz, and Ahmad Zaib. Deep learning for medical image processing: Overview, challenges and the future. *Classification in BioApps: Automation of decision making*, pages 323–350, 2018. 8

[52] Sucheng Ren, Daquan Zhou, Shengfeng He, Jiashi Feng, and Xinchao Wang. Shunted self-attention via multi-scale token

aggregation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10853–10862, 2022. 4, 5, 6, 8

[53] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 5, 6, 8

[54] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 8

[55] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 8

[56] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 8

[57] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 2

[58] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI conference on artificial intelligence*, 2017. 8

[59] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 1, 4

[60] Mingxing Tan and Quoc V Le. Mixconv: Mixed depthwise convolutional kernels. In *British Machine Vision Conference*, 2019. 5

[61] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2820–2828, 2019. 5

[62] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021. 2

[63] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 32–42, 2021. 2

[64] Pavan Kumar Anasosalu Vasu, James Gabriel, Jeff Zhu, Oncel Tuzel, and Anurag Ranjan. Fastvit: A fast hybrid vision transformer using structural reparameterization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5785–5795, 2023. 4, 8, 2

[65] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2

[66] Ao Wang, Hui Chen, Zijia Lin, Hengjun Pu, and Guiguang Ding. Repvit: Revisiting mobile cnn from vit perspective. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024. 4

[67] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 568–578, 2021. 4, 5, 8

[68] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pvt v2: Improved baselines with pyramid vision transformer. *Computational Visual Media*, 8(3):415–424, 2022. 4, 5, 6, 8

[69] Wenhai Wang, Jifeng Dai, Zhe Chen, Zhenhang Huang, Zhiqi Li, Xizhou Zhu, Xiaowei Hu, Tong Lu, Lewei Lu, Hongsheng Li, et al. Internimage: Exploring large-scale vision foundation models with deformable convolutions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14408–14419, 2023. 2, 4, 5, 6, 8

[70] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. 1

[71] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. 8

[72] Yuwen Xiong, Zhiqi Li, Yuntao Chen, Feng Wang, Xizhou Zhu, Jiapeng Luo, Wenhai Wang, Tong Lu, Hongsheng Li, Yu Qiao, et al. Efficient deformable convnets: Rethinking dynamic and sparse operator for vision applications. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024. 2, 4, 5, 6, 8

[73] Yufei Xu, Qiming Zhang, Jing Zhang, and Dacheng Tao. Vitae: Vision transformer advanced by exploring intrinsic inductive bias. *Advances in neural information processing systems*, 34:28522–28535, 2021. 4

[74] Chenglin Yang, Siyuan Qiao, Qihang Yu, Xiaoding Yuan, Yukun Zhu, Alan Yuille, Hartwig Adam, and Liang-Chieh Chen. Moat: Alternating mobile convolution and attention brings strong vision models. In *The Twelfth International Conference on Learning Representations*, 2023. 2, 4, 5

[75] Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. Focal self-attention for local-global interactions in vision transformers. *Advances in neural information processing systems*, 2021. 5

[76] Weihao Yu, Mi Luo, Pan Zhou, Chenyang Si, Yichen Zhou, Xinchao Wang, Jiashi Feng, and Shuicheng Yan. Metaformer

is actually what you need for vision. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10819–10829, 2022. 8

[77] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019. 2, 3, 4

[78] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 8

[79] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *The Seventh International Conference on Learning Representations*, 2018. 2, 3, 4

[80] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Haibin Lin, Zhi Zhang, Yue Sun, Tong He, Jonas Mueller, R Manmatha, et al. Resnest: Split-attention networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2736–2746, 2022. 8

[81] Jiangning Zhang, Xiangtai Li, Jian Li, Liang Liu, Zhucun Xue, Boshen Zhang, Zhengkai Jiang, Tianxin Huang, Yabiao Wang, and Chengjie Wang. Rethinking mobile block for efficient attention-based models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1389–1400. IEEE Computer Society, 2023. 4, 5, 6, 2

[82] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6848–6856, 2018. 1

[83] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017. 5, 4

[84] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, pages 13001–13008, 2020. 2, 4

[85] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127:302–321, 2019. 2, 4

UniConvNet: Expanding Effective Receptive Field while Maintaining Asymptotically Gaussian Distribution for ConvNets of Any Scale

Supplementary Material

Appendix

A. A proper Asymptotically Gaussian Distribution of small-scale pixels is more important than expanding the Effective Receptive Field

In dense prediction tasks (e.g., detection and segmentation), integrating contextual information via a large effective receptive field (ERF) [42] and distinguishing pixels of different scales are crucial. As shown in Fig. 6, the ERF is used to visualize the effectiveness of the proposed Three-layer RFA.

MogaNet-S [32] has similar asymptotically Gaussian distribution (AGD) at small-scale pixels, around the center, compared to UniConvNet-T, but UniConvNet-T exhibits a significantly larger ERF. This indicates that **expanding the ERF** while **maintaining the AGD** could help to generate a multi-scale impact, following AGD from center to edge, of a larger ERF, which consequently enhances the performance.

When comparing MogaNet-S [32] with ConvNeXt-T [40], MogaNet-S [32] has a better AGD at small-scale pixels with comparable ERF scale. This enables MogaNet-S [32] to have superior performance, demonstrating that the **AGD of small-scale pixels** is **more important** when the **ERF scale is comparable**.

Compared to UniConvNet-T, SLaK-T [37] achieves comparable ERF scale while disrupting the AGD of ERF. The top-1 accuracy on ImageNet increased by 1.7% point owing to the proper AGD on the area larger than small-scale area in SLaK-T [37]. UniRepLKNet-T [18] achieves much larger ERF, compared to SLaK-T [37], with inferior AGD, which benefits from extremely large ERF. It is constrained by high parameters and FLOPs costs compared with UniConvNet-T. This demonstrates that the sparsity [37] and re-parameterization [15, 16] techniques effectively enlarge the ERF but suffer from improper AGD of smaller-scale pixels (the dark gray area in UniRepLKNet-T [18]). Compared to UniConvNet-B, RepLKNet-31B [17] achieves a larger ERF but compromises small-scale AGD, with a 1.0% TOP-1 accuracy drop on ImageNet. These phenomena typically demonstrate our viewpoint that **a proper asymptotically Gaussian distribution of small-scale pixels is more important than expanding the Effective Receptive Field**.

As shown in Fig. 7, UniConvNet variants consistently

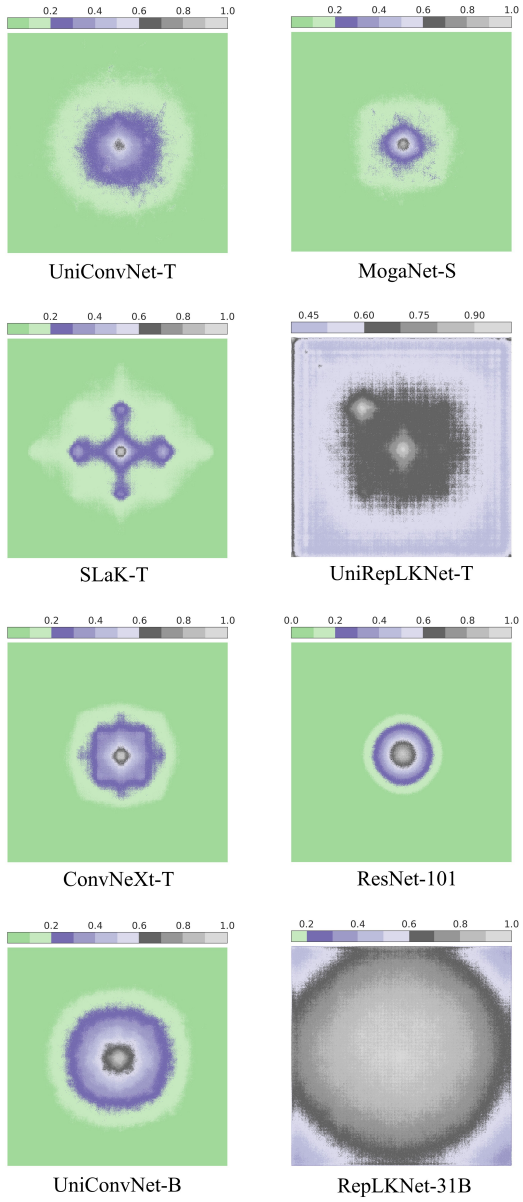


Figure 6. **Effective Receptive Field (ERF) of UniConvNet-T, MogaNet-S, SLaK-T, ConvNeXt-T, UniRepLKNet-T, ResNet-101 and UniConvNet-B, RepLKNet-31B.** The more stepped colour area indicates better AGD. The wider area indicates a larger ERF. Each ERF is based on an average of 1000 images with a resolution of 224×224.

demonstrate large ERF while maintaining AGD. These findings suggest that the Three-layer RFA can extend the ERF with proper combination of smaller kernels (e.g., 7×7, 9×9, 11×11).

B. Throughput Analysis Configuration

We use an A100 40GB GPU to benchmark throughput on several classic and relevant models. The software environment is PyTorch 1.13, CUDA 11.7, cuDNN 8.5. The hardware and software configurations align with InternImage [69] to ensure a fair comparison. The overall throughput of each model, measured as the number of images processed per second, is reported in FP32/FP16 data formats.

C. Illustration of UniConvNet Block

C.1. Stem & Downsampling Block

Similar to ConvNeXt [40] and InternImage [69], our model adopts a pyramid architecture with a stem block and three downsampling blocks to generate multi-scale feature maps. As shown in Fig. 5, The stem block, positioned before the first stage, reduces the input resolution by a factor of 4. The stem block employs a bottleneck design, comprising two stacked 3×3 convolution layers and LayerNorm layers, interspersed with a GELU activation function to introduce nonlinearity to input images. The 3×3 convolutions have strides of 2 and padding of 1. The first convolution output channels are half those of the second. The downsampling block between stages reduces the input resolution by a factor of 2. It consists of a LayerNorm layer followed by a 3×3 convolution with a stride of 2 padding of 1.

C.2. Basic Block

Inspired by the state-of-the-art CNN model InternImage, which integrates LayerNorm [2], feed-forward networks [65], and GELU [25], UniConvNet incorporates three stacked residual components in basic blocks. Each residual component begins with a LayerNorm layer to normalize input features, followed sequentially by the Three-layer RFA, modified DCNV3 [69], and a feed-forward network. The basic block initially employs the Three-layer RFA residual component to extract multi-scale features from small- and large-scale patterns, establishing long-range and multi-scale dependencies. Following the Three-layer RFA residual component, a 3×3 (modified DCNV3 [69]) residual convolution block and a feed-forward network are used for densely local perception, akin to conventional ConvNets.

D. Training Settings

D.1. ImageNet-1K/22K Training

We adopt the commonly used training recipes from state-of-the-art methods [18, 40, 62, 64, 69, 72, 81] and remove some tune-ups for fair comparisons and to better represent the effectiveness of the proposed UniConvNet. Additionally, we apply a weak training recipe, following EMO [81], to improve performance on smaller models (UniConvNet-A/P0/P1/P2), and a strong training recipe, based on common practice [40], for larger variants (UniConvNet-N0/N1/N2/N3/T/S/B). All experiments are conducted on the ImageNet-1K [14] dataset, comprising 1000 object classes and 1.2 million training images.

Super Prameters	**UniConvNet-A /P0/P1/P2(W)** ImageNet-1K	**UniConvNet-N0/N1 /N2/N3/T/S/B(S)** ImageNet-1K	**UniConvNet-L/XL(S)** ImageNet-22K
Input Scale	224^2	224^2	192^2
Training Epochs	300	300	90
Batch Size	4096	4096	4096
Optimizer	AdamW	AdamW	AdamW
Optimizer Momentum	$\beta_1, \beta_1 = 0.9, 0.999$	$\beta_1, \beta_1 = 0.9, 0.999$	$\beta_1, \beta_1 = 0.9, 0.999$
Base Learning Rate	$4e^{-3}$	$4e^{-3}$	$4e^{-3}$
Learning Rate Schedule	cosine	cosine	cosine
Learning Rate Decay	$5e^{-2}$	$5e^{-2}$	$5e^{-2}$
Layer-wise Learning Rate Decay	×	×	×
Warmup Epochs	20	20	20
Warmup Schedule	linear	linear	linear
Label Smoothing ε	0.1	0.1	0.1
Dropout Rate	×	×	×
Drop Path Rate	0.05/0.05/0.05/0.08	0.08/0.1/0.1/0.1/0.2/0.4/0.6	0.2/0.2
Layer Scale	$1e^{-6}$	$1e^{-6}$	$1e^{-6}$
RandAugment	(9,0.5)	(9,0.5)	(9,0.5)
Color Jitter	0.4	0.4	0.4
Horizontal Flip	×	×	×
Random Resized Crop	×	×	×
Repeated Augment	×	×	×
Head Init Scale	×	×	×
Mixup Alpha	×	0.8	0.8
Cutmix Alpha	×	1.0	1.0
Erasing Probability	×	0.25	0.25
Gradient Clip	×	×	×
Loss	Cross Entropy	Cross Entropy	Cross Entropy
Exp. Mov. Avg. (EMA)	0.9999	0.9999	×

Table 11. **(Pre-)Training settings for various model variants on ImageNet-1K/22K.** The training recipes adhere to standard practices [18, 40, 62, 64, 69, 72, 81], with certain tune-ups removed. Multiple stochastic depth drop rates (e.g., 0.08/0.1/0.1/0.1/0.2/0.4/0.6) are assigned to UniConvNet-N0/N1/N2/N3/T/S/B, respectively. "W" and "S" indicate that the UniConvNet variants are trained using the weak and strong training recipes, respectively.

Using the weak training recipe, we train UniConvNet-A/P0/P1/P2 models from scratch with 224×224 inputs for 300 epochs. The AdamW optimizer is used with a learning rate of 4×10^{-3}. Training begins with a 20-epoch linear warmup, followed by a cosine decay learning rate schedule. A batch size of 4096 and a weight decay of 0.05 are employed. RandAugment [12] is applied for data augmentation in the weak training recipe. Regularization techniques, including Stochastic Depth [28] and Label Smoothing [57], are employed. A Layer Scale [63] with an initial value of 1×10^{-6} is used. Exponential Moving Average (EMA) [48] is employed to reduce overfitting in larger models.

For UniConvNet-N0/N1/N2/N3/T/S/B, the strong training recipe is applied, incorporating additional data augmentation techniques such as Mixup [79], Cutmix [77], and Random Erasing [84], to enhancethe dataset for training on larger models. For UniConvNet-L, we follow the strong training recipe and change the input image size to 192×192.

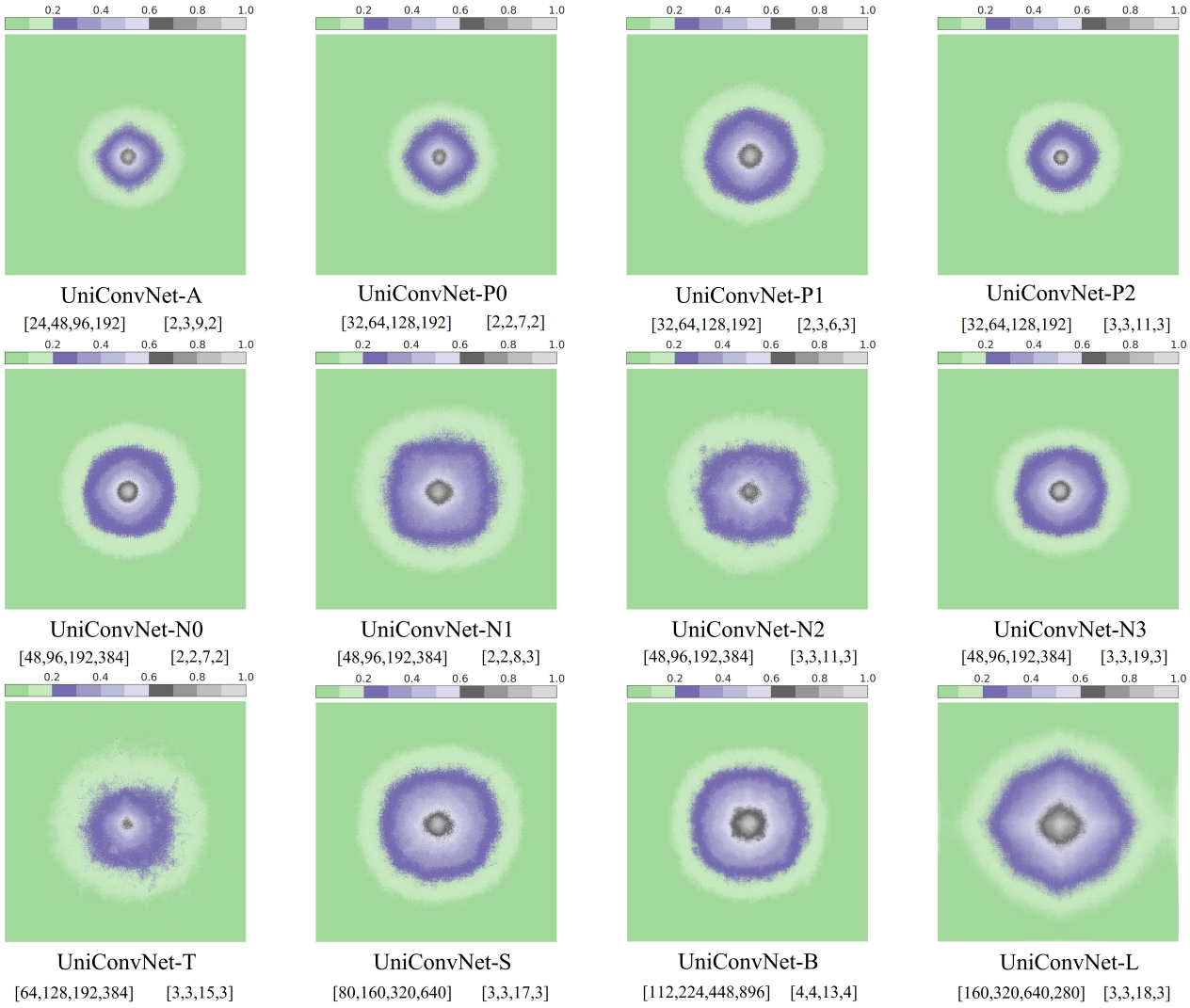


Figure 7. **Effective Receptive Field (ERF) of all UniConvNet variants.** The more stepped colour area indicates better AGD. The wider area indicates a larger ERF. Each ERF is based on an average of 1000 images with a resolution of 224×224.

Detailed training configurations for different model variants are provided in Tab. 11.

D.2. ImageNet-1K Fine-tuning

For ImageNet-1K fine-tuning, compared to the strong training recipes for ImageNet-1K/22K, the base learning rate of the AdamW optimizer is set to 5×10^{-5}. The learning rate decay is set to 1×10^{-8}. ImageNet-1K fine-tuning is performed with a batch size of 512, without requiring warm-up. Different layer-wise learning rate decay factors are applied: 0.7 for UniConvNet-T/S/B and 0.8 for UniConvNet-L. Data augmentation techniques, Mixup [79] and Cutmix [77], are removed to improve fine-tuning results. Additionally, ImageNet-1K pre-trained UniConvNet-T/S/B and ImageNet-22K pre-trained UniConvNet-L are fine-tuned at an increased resolution of 384×384.

D.3. Training Recipes for Classification

We evaluate the performance of UniConvNet-A/P0/P1/P2/N0/N1/N2 on ImageNet-1K to conduct an ablation study comparing weak and strong training recipes. As illustrated in Tab. 13, the weak training recipe exhibits overfitting with models having $10.2M$ parameters, while models with $13.2M$ parameters start benefiting from the strong training recipe. Consequently, the choice of training recipes is straightforward: UniConvNet-A/P0/P1/P2/N0 adopts the weak training recipe to leverage smaller-scale datasets. UniConvNet-N1/N2 and larger models employ the strong

Super Prameters	**UniConvNet-T/S/B** ImageNet-1K pt ImageNet-1K ft	**UniConvNet-L/XL** ImageNet-22K pt ImageNet-1K ft
Input Scale	384^2	384^2
Training Epochs	30	30
Batch Size	512	512
Optimizer	AdamW	AdamW
Optimizer Momentum	$\beta_1, \beta_1 = 0.9, 0.999$	$\beta_1, \beta_1 = 0.9, 0.999$
Base Learning Rate	$5e^{-5}$	$5e^{-5}$
Learning Rate Schedule	cosine	cosine
Learning Rate Decay	$1e^{-8}$	$1e^{-8}$
Layer-wise Learning Rate Decay	0.7	0.8
Warmup Epochs	✗	✗
Warmup Schedule	✗	✗
Label Smoothing ε	0.1	0.1
Dropout Rate	✗	✗
Drop Path Rate	0.4/0.6/0.8	0.3/0.35
Layer Scale	pre-trained	pre-trained
RandAugment	(9,0.5)	(9,0.5)
Color Jitter	0.4	0.4
Horizontal Flip	✗	✗
Random Resized Crop	✗	✗
Repeated Augment	✗	✗
Head Init Scale	0.001	0.001
Mixup Alpha	✗	✗
Cutmix Alpha	✗	✗
Erasing Probability	0.25	0.25
Gradient Clip	✗	✗
Loss	Cross Entropy	Cross Entropy
Exp. Mov. Avg. (EMA)	0.9999	0.9999

Table 12. **Fine-tuning settings for various model variants on ImageNet-1K.** The training recipe follows common practices [18, 40]. Multiple stochastic depth drop rates (e.g., 0.4/0.6/0.8) are for UniConvNet-T/S/B, respectively. "ImageNet-1K pt", "ImageNet-1K ft", and "ImageNet-22K pt" represent ImageNet-1K pre-training, ImageNet-1K fine-tuning and ImageNet-22K pre-training, respectively.

UniConvNet	#Params	ACC-W(%)	ACC-S(%)
UniConvNet-A	3.4M	**77.0**	✗
UniConvNet-P0	5.2M	**79.1**	✗
UniConvNet-P1	6.1M	**79.6**	78.8
UniConvNet-P2	7.6M	**80.5**	79.9
UniConvNet-N0	10.2M	**81.6**	**81.6**
UniConvNet-N1	13.1M	81.8	**82.2**
UniConvNet-N2	15.0M	✗	**82.7**

Table 13. **Explorations of Training Recipes for Classification.** "ACC-W(%)" and "ACC-S(%)" are the TOP-1 accuracy trained by weak traininig recipe and strong traininig recipe, respectively.

training recipe, utilizing augmented [77, 79, 84] datasets to optimize performance with larger parameter sizes.

D.4. Object Detection and Instance Segmentation Fine-tuning

Following EMO [81], we utilize the standard MMDetection [6] library and the AdamW [41] optimizer to train the heavy RetinaNet and light SSDLite models with a batch size of 16 on 8 A100 GPUs.

Following standard practices [40, 69, 72], we further fine-tune the scaled-up UniConvNet using batch sizes of 16 and 8, respectively, for fair comparisons. Under the $1\times$ schedule, images are resized so that the shorter side is 800 pixels and the longer side does not exceed 1333 pixels. During testing, the shorter side is fixed at 800 pixels. Under the $3\times$ schedule, the longer side remains capped at 1333 pixels, while the shorter side is resized to a range of 480–800 pixels. We also employ the standard MMDetection [6] library and the AdamW [41] optimizer for training, using a base learning rate of 1×10^{-4}.

D.5. Semantic Segmentation Fine-tuning

We fine-tune DeepLabv3 [7] and PSPNet [83] using the ImageNet-1K pre-trained UniConvNet on the ADE20K [85] dataset. Following EMO [81], we use the MMSegmentation [11] library and the AdamW [41] optimizer to train DeepLabv3 [7] and PSPNet [83] for 160k iterations on 8 A100 GPUs, ensuring fair comparisons.

The scaled-up UniConvNet is fine-tuned with the UperNet framework on ADE20K for 160k iterations, using a batch size of 16. The AdamW [41] optimizer is used for training. The base learning rates are set to 6×10^{-5} for UniConvNet-N2/N3/T/S/B and 2×10^{-5} for UniConvNet-L. A polynomial decay schedule with a power of 1.0 is applied for learning rate decay. Following common practices [38, 40, 69, 72], images are cropped to 512×512 for UniConvNet-N2/N3/T/S/B and 640×640 for UniConvNet-L to ensure fair comparisons.