APPENDIX

A TRAINING SETTINGS

Image Classification We employ the ImageNet-1K dataset (Deng et al., 2009) for classification that includes 1.2M and 50K training and validation images. The dataset has 1000 categories and we report the performance in terms of top-1 accuracy. In addition, we use ImageNet-21K dataset which has 14M images with 21841 classes for pretraining.

We train all FasterViT models by using LAMB optimizer (You et al., 2019) optimizer for 300 epochs with a learning rate of 5e-3 and a total batch size of 4096 using 32 A100 GPUs. For data augmentation, we follow same strategies as in previous efforts (Liu et al., 2022b; 2021). We also use Exponential Moving Average (EMA) which often improves the performance. Further details on training settings can be found in the appendix. For pre-training on ImageNet-21K, we train the models for 90 epochs with a learning rate of 4e-3. In addition, we fine-tune the models for 60 epochs with a learning rate of 7e-5.

Detection and Segmentation We used the MS COCO dataset (Lin et al., 2014) to finetune a Cascade Mask-RCNN network (He et al., 2017) with pretrained FasterViT backbones. For this purpose, we trained all models with AdamW (Loshchilov & Hutter, 2017) optimizer with an initial learning rate of 1e-4, a 3 $\times$ schedule, weight decay of 5e-2 and a total batch size of 16 on 8 A100 GPUs.

Semantic Segmentation For semantic segmentation, we employed ADE20K dataset (Zhou et al., 2017) to finetune an UperNet network (Xiao et al., 2018) with pre-trained FasterViT backbones. Specifically, we trained all models with Adam-W (Loshchilov & Hutter, 2017) optimizer and by using a learning rate of 6e-5, weight decay of 1e-2 and total batch size of 16 on 8 A100 GPUs.

B ROBUSTNESS ANALYSIS

In this section, we analyze the robustness of FasterViT models on different datasets. We test FasterViT model variants on ImageNet-A Hendrycks et al. (2021b), ImageNet-R Hendrycks et al. (2021a) and ImageNetV2 Recht et al. (2019) datasets. In addition, we did not perform any fine-tuning and simply employed the pre-trained ImageNet-1K Deng et al. (2009) weights for each model. As shown in Table S.2, FasterViT demonstrates promising robustness performance on various datasets for each model variant. Specifically, FasterViT-3 outperforms comparable models such as ConvNeXt-B and Swin-B Liu et al. (2022b) by +7.5% and +8.4% on ImageNet-A Hendrycks et al. (2021b), +0.6% and +5.3% on ImageNet-R Hendrycks et al. (2021a) and +1.3% and +2.7% on ImageNetV2 Recht et al. (2019), respectively. For larger models, FasterViT-4 outperforms ConvNeXt-L Liu et al. (2022b) by +7.9%, +2.6% and +1.5% on ImageNet-A Hendrycks et al. (2021b), ImageNet-R Hendrycks et al. (2021a) and ImageNetV2 Recht et al. (2019), respectively, hence validating the effectiveness of the proposed model in various benchmarks. Similar trends can be observed for smaller models.

C ABLATION

C.1 COMPONENT-WISE STUDY

Table S.1 shows per component ablation. Two settings are considered: (i) when the model is trained without the component, (ii) when the component is disabled after the model is trained. The first shows if the model can operate well without the component, while the second cases shows if the components is used in the final model.

We observe that changing the window resolution to 14×14 in the 3rd stage (effectively removing HAT by have a full global window) improves the model accuracy by $+0.1\%$ while scarifying 10% of throughput. Even though this setup shows better accuracy, it does not scale to high resolution, and HAT is required. Removing the HAT block from the architecture results in -0.24% accuracy drop for re-trained model and -1.49% for post training study at the benefit of 8% throughtput improvement. CT attention is another block of high importance, resulting in -3.85% post training

Table S.2: Robustness analysis of **ImageNet-1K** Deng et al. (2009) pretrained FasterViT models on ImageNet-A Hendrycks et al. (2021b), ImageNet-R Hendrycks et al. (2021a) and ImageNetV2 Recht et al. (2019) datasets.

Model	Size (Px)	#Param (M)	FLOPs (G)	Throughput (Img/Sec)	Clean (%)	A (%)	R (%)	V2 (%)
FasterViT-0	224	31.4	3.3	**5802**	**82.1**	**23.9**	**45.9**	**70.9**
FasterViT-1	224	53.4	5.3	**4188**	**83.2**	**31.2**	**47.5**	**72.6**
Swin-T Liu et al. (2021)	224	28.3	4.4	2758	81.3	21.6	41.3	69.7
ConvNeXt-T Liu et al. (2022b)	224	28.6	4.5	3196	82.0	24.2	47.2	71.0
ConvNeXt-S Liu et al. (2022b)	224	50.2	8.7	2008	83.1	31.3	49.5	72.4
FasterViT-2	224	75.9	8.7	**3161**	**84.2**	**38.2**	**49.6**	**73.7**
Swin-S Liu et al. (2021)	224	49.6	8.5	1720	83.2	32.5	44.7	72.1
Swin-B Liu et al. (2021)	224	87.8	15.4	1232	83.4	35.8	46.6	72.3
ConvNeXt-B Liu et al. (2022b)	224	88.6	15.4	1485	83.8	36.7	51.3	73.7
FasterViT-3	224	159.5	18.2	**1780**	**84.9**	**44.2**	**51.9**	**75.0**
ConvNeXt-L Liu et al. (2022b)	224	198.0	34.4	508	84.3	41.1	53.4	74.2
FasterViT-4	224	424.6	36.6	**849**	**85.4**	**49.0**	**56.0**	**75.7**
FasterViT-5	224	975.5	113.0	**449**	**85.6**	**52.7**	**56.9**	**76.0**
FasterViT-6	224	1360.0	142.0	**352**	**85.8**	**53.7**	**57.1**	**76.1**

removal. Attention bias is an important component of our system, resulting in -0.31% drop in the re-training scenario. Removing CT propagation, results in the requirement to pool and propagate features at every layer (similar to EdgeViT), that costs 7% of total inference and in lower accuracy -0.16%. CT initialization is important to the network, as accuracy drops by -0.48% in post-training removal. Removing all components and having only CNN plus windowed vanilla transformer results in -0.46%.

C.2 SwinV2 Comparison

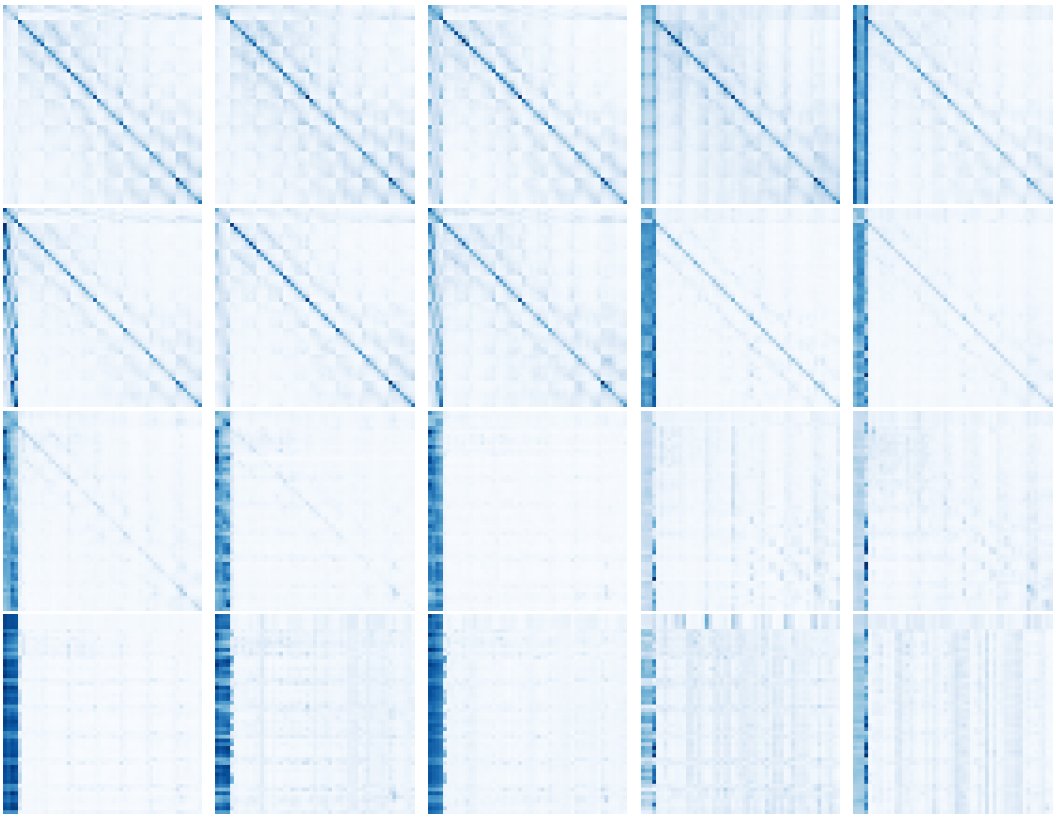
In the Table **??** we compare the performance of SwinV2 Liu et al. (2022a) and FasterViT models on large image resolution. The initial model is pretrained with an image resolution of 256^2px for 300 epochs on ImageNet-1K. Then models are fine-tuned on a larger resolution (I) for an 30 epochs with various window sizes (W). Faster-ViT consistently demonstrates a higher image throughput, sometimes by a significant margin compared to Swin Transformer V2 model. Hence validating the effectiveness of the proposed hierarchical attention for high input resolution.

Ablation	Trained from scratch	Post training removal	Throughput ratio
HAT block	-0.24%	-1.49%	1.08
CT attention	-0.13%	-3.85%	1.00
Attention Bias	-0.31%	-8.90%	1.00
CT propagation	-0.16%	-	0.93
1D pos bias	-0.07%	-24.85%	1.00
CT initialization	-0.05%	-0.48%	1.00
Window 14×14	+0.10%	-	0.90

Table S.1: Ablation study on the effectiveness of different components of HAT.

D Attention Maps

In Fig. S.1, we have illustrated the full attention maps of stage 3 layers for different FasterViT model variants. For this purpose, we use input images of size $224 \times 224 \times 3$ and ImageNet-1K Deng et al. (2009) trained FasterViT models. For each model, from the top to the bottom rows, we show the attention maps from the first to the final layer with an interval of a quarter of the total number of layers at stage 3 (*e.g.* layers 1, 4, 9 and 12 for FasterViT-4).



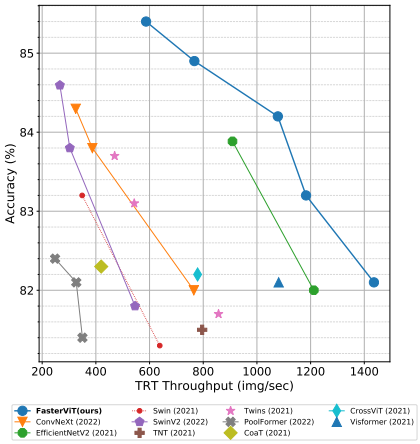
(a) FasterViT-0 (b) FasterViT-1 (c) FasterViT-2 (d) FasterViT-3 (e) FasterViT-4

Figure S.1: (a) FasterViT-0. (b) FasterViT-1. (c) FasterViT-2. (d) FasterViT-3 (e) FasterViT-4. Full attention map visualizations of stage 3 for FasterViT model variants. From top to bottom, we visualize attention maps of first to last layers with an interval of a quarter length of the number of layers in stage 3 for each model. We visualize the attention maps of the same input image for all cases to facilitate comparability.

In particular, Stage 3 for this illustration serves an important purpose, since we use local attention windows of 7×7 with input features that have a resolution of 14×14. Hence, attention is computed in 4 local regions after window partitioning and 4 carrier tokens are designated to each corresponding window. Each illustrated attention map has a size of size 53×53 consisting of a concatenation of 4×4 carrier tokens and 49×49 local window-based attention. The carrier tokens are shown in in the top left position of each map. We observe that for all models, all tokens will attend to the carrier tokens with different patterns.

For FasterViT-0 and FasterViT-1 models, from the first to the last layers, all tokens transition to attend to the the carrier tokens (*i.e.* vertical bar on the left side). In the last layers, in addition to all tokens attending to the carrier tokens, we see a more global attention pattern, hence showing the cross interaction between different regions.

Figure S.2: Comparison of image throughput and ImageNet-1K Top-1 accuracy with TensorRT post-training model optimization. For all models, throughput is measured on A100 GPU with batch size of 1.



For FasterViT-2, FasterViT-3 and FasterViT-4 models, starting from the first layers, all tokens attend to both carrier and local tokens. In the last layers however, the attention pattern shifts from local to global. As discussed in this work and also shown in these illustrations, carrier tokens serve an integral role in modeling cross-region interactions and capturing long-range spatial dependencies.

E TensorRT latency

All throughput numbers and insights presented in the main paper were computed using PyTorch v1.13. In order to demonstrate the scalability with post-training optimization techniques,

we compared throughput using the TensoRT (TRT) framework for *batch size 1*, as illustrated in Fig S.2. FasterViT is still considerably faster than other models, making it a good choice to meet various efficient inference design targets.

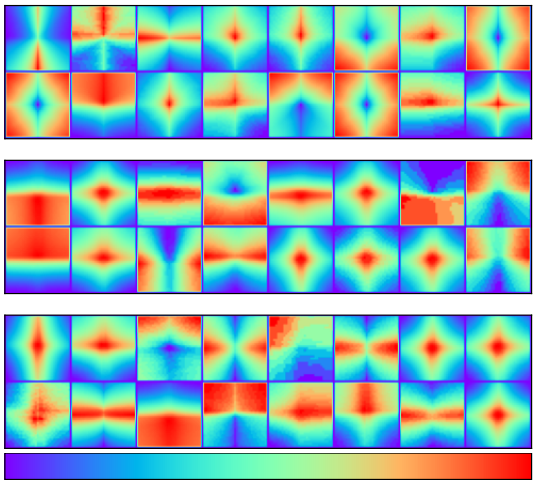


Figure S.3: Learned positional biases for attentions in the 3rd stage of FasterViT-4 model finetuned for 512×512 px. Each kernel corresponds to a bias for a single head in the multi-headed attention. Visualizations demonstrate that the model learns positional dependent features, while also sharing the pattern between pixels.

F Attention bias

We follow the concept of relative positional bias in the attention from Swin Liu et al. (2021). Particularly, we use the implementation with MLP from SwinV2 Liu et al. (2022a), where relative coordinate shift in x, y is transformed to the positional bias in the attention via 2-layer network. This allows the model to learn relative position aware kernels, and to introduce image inductive bias. We visualize learned positional

biases of the MLP in FasterViT-4 finetuned for 512 with window size of 16×16 pixels in Fig S.3. The visualization shows a diverse set of kernels learned by FasterViT model.

G FasterViT Profiling

In Fig. S.4, we provide detailed stage-wise profiling of FasterViT-2 using NVIDIA DLSIM. As expected, stage 3 (HAT) has the highest latency, FLOPs and memory footprint since it is composed of considerably more layers compared to other stages.

H Design Insights

Layer normalization Ba et al. (2016). We found it to be critical for transformer blocks (stage 3 and 4). Replacing it with batch normalization leads to accuracy drop of 0.7%. The LN performs cross token normalization and affects cross-channel interaction.

No feature map reshaping. In our architecture, we have removed windowing and de-windowing functions from transformer layers. They are usually used to perform convolutions between layers (like in Twins Chu et al. (2021), EdgeViT Pan et al. (2022), Visformer Chen et al. (2021)), or window shifting (Swin Liu et al. (2021), SwinV2 Liu et al. (2022a)). We perform windowing only once once in stages 3 and 4, and keep data as tokenized with channel last. This leads to throughput improvement of 5% for PyTorch and 10% for TensorRT.

LAMB optimizer You et al. (2019). We observed incredible stability of LAMB You et al. (2019) optimizer for training our biggest models (FasterVIT-3 and FasterViT-4), more widely used AdamW Loshchilov & Hutter (2017) was leading to NaNs for some trainings. We attribute this to joined usage of batch normalization and layer normalization Ba et al. (2016) in the same model.

Positional bias. We employ 1D positional bias for local and carrier tokens, as well as 2D relative attention bias by MLP introduced in SwinV2 Liu et al. (2022a). For 1D bias we remove *log* scale. This approach yields flexibility to the image size, as positional encoding is interpolated by MLP if resolution change. Those positional biases are quick to compute, however, will block all cores in GPUs until positional biases are computed, and will significantly impact the throughput. To address this, we propose to pre-compute positional biases for a given feature resolution and skip the MLP bottleneck, leading to 6% throughput gain.

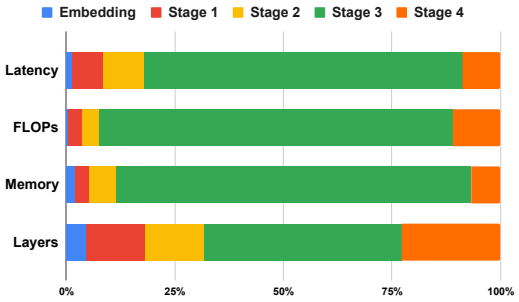


Figure S.4: **FasterViT-2** profiling benchmarks. Stage 3 (HAT) dominates over all metrics.

Drop-out. We found that conventional drop-out on MLP layers and attention has a negative effect on the final accuracy even for big models that overfit. Stochastic depth is helpful; in contrary to recent trends, we found that a small probability (up to 30%) works better than 65% like in DEIT3 Touvron et al. (2022). Better regularization can be achieved by increased weight decay. For example, model 4 with drop-path rate of 50% and weight decay of 0.05 achieves 84.91%, while model 4 with drop-path rate of 30% and weight decay of 0.12 achieves 85.15%.

MESA Du et al.. It is shown to be useful to prevent overfitting of larger models at little overhead. MESA is a simplified version of SAM Foret et al. (2020) that forces optimization to have sharper minima at the convergence, naive implementation slows down training by 2x. In MESA, authors propose to simply apply knowledge distillation loss with respect to the EMA weight computed during training, the training overhead is almost not noticeable. We enable it after 25% of the training, coefficient is set proportionally to the model size in range 0.25 (FasterViT-0)-3.0(FasterViT-4).

Intermediate LN. SwinV2 Liu et al. (2022a) argues that intermediate LN Ba et al. (2016) help to stabilize training of large models, we saw accuracy degradation of this approach.

	Output Size (Downs. Rate)	FasterViT-1	FasterViT-2	FasterViT-3	FasterViT-4
Stem	112×112 (2×)	[Conv-BN-ReLU C:32, S:2] × 1	[Conv-BN-ReLU C:64, S:2] × 1	[Conv-BN-ReLU C:64, S:2] × 1	[Conv-BN-ReLU C:64, S:2] × 1
		[Conv-BN-ReLU C:80] × 1	[Conv-BN-ReLU C:96] × 1	[Conv-BN-ReLU C:128] × 1	[Conv-BN-ReLU C:196] × 1
Stage 1	56×56 (4×)	LN-2D, Conv, C:160, S:2	LN-2D, Conv, C:192, S:2	LN-2D, Conv, C:256, S:2	LN-2D, Conv, C:392, S:2
		[ResBlock C:160] ×1,	[ResBlock C:192] × 3,	[ResBlock C:256] × 3,	[ResBlock C:392] × 3,
Stage 2	28×28 (8×)	LN-2D, Conv, C:320, S:2	LN-2D Conv, C:384, S:2	LN-2D, Conv, C:512, S:2	LN-2D, Conv, C:768, S:2
		[ResBlock C:320] × 3,	[ResBlock C:384] × 3,	[ResBlock C:512] × 3,	[ResBlock C:768] × 3,
Stage 3	14×14 (16×)	LN-2D, Conv, C:640, S:2	LN-2D, Conv, C:768, S:2	LN-2D, Conv, C:1024, S:2	LN-2D, Conv, C:1568, S:2
		[HAT C:640, head:8] × 8 ,	[HAT C:768, head:8] × 8,	[HAT C:1024, head:8] × 12,	[HAT C:1568, head:16] × 12,
Stage 4	7×7 (32×)	LN-2D, Conv, C:1280, S:2	LN-2D, Conv, C:1536, S:2	LN-2D, Conv, C:2048, S:2	LN-2D, Conv, C:3136, S:2
		[HAT C:1280, head:16] × 5,	[HAT C:1536, head:16] × 5,	[HAT C:2048, head:16] × 5,	[HAT C:3136, head:32] × 5,

Table S.3: FasterViT architecture configurations. BN and LN-2D denote Batch Normalization and 2D Layer Normalization, respectively. HAT denotes Hierarchical Attention block.

I Architecture Details

In Table S.3, we show the different architecture configurations of the FasterViT model variants.

J Carrier Token Size

In Table S.4, we investigate the effect of carrier token size and window size on accuracy and latency of the model. We observe that increasing the carrier token window size can improve the performance at the cost of increased latency, sometimes by a significant margin. The 2x2 carrier token window size offers a great trade-off between accuracy and

Table S.5: **MS COCO** dataset (Lin et al., 2014) object detection results with DINO (Zhang et al., 2022) model. ‡ denotes models that are pre-trained on ImageNet-21K dataset.

Backbone	Model	Epochs	FLOPs (G)	Throughput	AP^{box}
Swin-L‡ (Liu et al., 2021)	HTC++ (Chen et al., 2019)	72	1470	-	57.1
Swin-L‡ (Liu et al., 2021)	DINO (Zhang et al., 2022)	36	1285	71	58.5
FasterViT-4‡	DINO (Zhang et al., 2022)	36	1364	**84**	**58.7**

latency. In addition, increasing the window size from 7 to 14 increases the Top-1 accuracy by +0.2%. However, as expected, it increases the latency by 10%. Hence, this shows the advantage of leveraging carrier token as an efficient mechanism to capture long-range contextual information. We also note that although increasing the window size results in better performance, it does not scale properly to higher resolution images. As a result, HAT is a more effective and efficient mechanism that can be employed without sacrificing image throughput.

Window Size	Carrier Token Size	Latency Ratio	Top-1 (%)
7	2	1	84.2
7	1	1.05	83.9
7	9	0.47	84.9
14	0	0.9	84.4

Table S.4: Effect of window and carrier token size on latency and Top-1 accuracy.

K Downstream Experiments

We provide additional experiments for both object detection and semantic segmentation with more models, across different sizes, to demonstrate the effectiveness and efficiency of our work. Firstly, in Table S.5, we present additional object detection experiments with DINO on MS-COCO dataset. The DINO model with FasterViT-4 is 18.30% faster than its counterpart with Swin-L backbone in terms of image throughput and outperforms it by +0.1 in terms of box AP.

We also added a semantic segmentation study on the ADE20K dataset with the FPN network, as shown below. Specifically, we compare against PoolFormer and PVT backbones. In this experiment, the model with FasterViT-1 backbone outperforms counterpart PoolFormer-S36 by +0.7 in terms of mIoU while also being 8.38% faster in terms of image throughput. Similarly, the model with FasterViT-2 backbone significantly outperforms PoolFomer-M36 counterpart by +1.1 in terms of mIOU while being 10.05% faster. We have added these experiments to the manuscript. We believe that the above experiments validate the effectiveness of FasterViT as an efficient backbone for downstream tasks such as segmentation and detection across different model sizes.

Backbone	Model	Throughput	mIoU
PoolFomer-S36 (Yu et al., 2022)	FPN	453	42.0
FasterViT-1	FPN	**491**	**42.7**
PoolFomer-M36 (Yu et al., 2022)	FPN	368	42.4
FasterViT-2	FPN	**405**	**43.5**

Table S.6: Semantic segmentation on **ADE20K** (Zhou et al., 2017) with FPN network.

L Impact of Conv-blocks on Throughput

We conducted an additional ablation study to demonstrate the effect of conv-based block on both accuracy and throughput as shown below. According to our experiments, replacing Conv-based blocks with Transformer-based counterparts significantly reduces the throughput while also reducing the accuracy. As expected, the Conv-based blocks are more efficient than the transformer counterparts for processing larger input sizes. The model with conv-based blocks also has higher accuracy compared to their fully-transformer-based counterparts due to incorporating inductive biases such as locality. The combination of Conv-based (stage 1 and 2) and transformer-based (stage 3 and 4) architecture as presented in FasterViT strikes the right balance between accuracy and efficiency.

M THROUGHPUT ON DIFFERENT PLATFORMS

In order to validate the effectiveness of FasterViT on different platforms, we present additional throughput comparisons on different hardware such as NVIDIA V100, NVIDIA TITAN RTX and NVIDIA A6000 GPUs, Jetson Nano and Intel(R) Xeon(R) E5-2698 v4 CPU. For all comparisons, we use a batch size of 128, unless otherwise stated. Our benchmarks show that FasterViT achieves a Pareto-front for ImageNet Top-1 and throughput trade-off, hence validating the effectiveness and scalability of our model to different hardware platforms.

Model	Top-1	Throughput
FasterViT-0	82.1	5802
FasterViT-0 wo Conv-block	81.7	3616
FasterViT-1	83.2	4188
FasterViT-1 wo Conv-block	82.8	3280
FasterViT-2	84.2	3161
FasterViT-2 wo Conv-block	83.8	2085
FasterViT-3	84.9	1780
FasterViT-3 wo Conv-block	84.5	1397
FasterViT-4	85.4	849
FasterViT-4 wo Conv-block	84.9	712

Table S.7: Effect of Conv-based stages on throughput and accuracy of different FasterViT models.

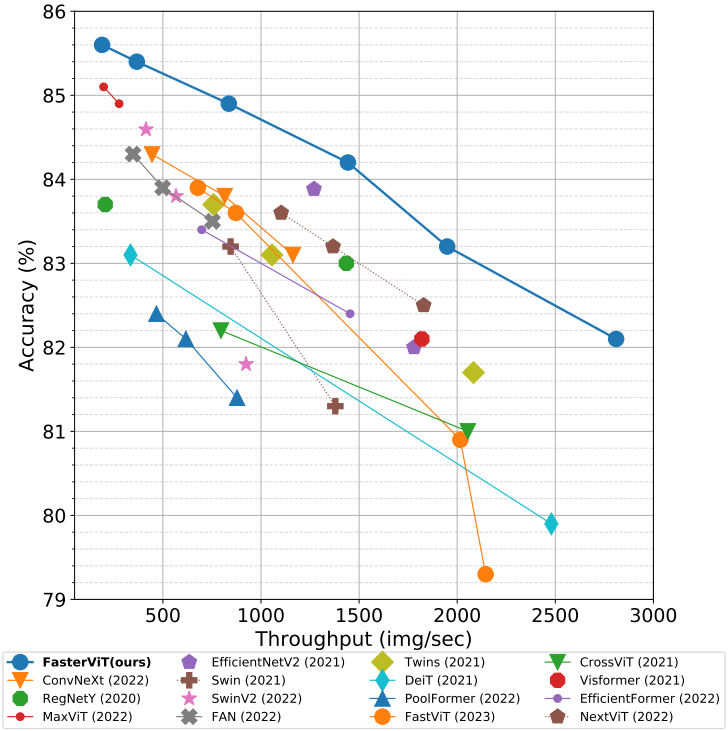


Figure S.5: Comparison of image throughput and ImageNet-1K Top-1 accuracy on NVIDIA V100 GPU for batch size of 128.

In Fig. S.5, we demonstrate the throughput and accuracy trade-off on V100 GPU and observe that FasterViT achieves a Pareto front. Additionally, in Fig. S.6, we illustrate the same comparison for NVIDIA TITAN RTX GPU, which is considered as an enthusiast-class graphics card. Surprisingly, we see that FasterViT attains a Pareto front on this platforms as well.

In addition, as shown in Fig. S.7, we report the throughput for all models using an NVIDIA A6000 GPU to confirm the scalability of our proposed architecture to various types of hardware. On A6000 GPU, FasterViT still demonstrates a strong performance and achieves a SOTA Pareto front except for an EfficientNetV2 variant which achieves a comparable performance to FasterViT-2.

In addition to GPU hardware, we have also measured throughput on a CPU device as well as NVIDIA Jetson Nano which is considered as an embedded system. In Fig. S.8, we demonstrate measurement for Top-1 and image throughput on Intel(R) Xeon(R) E5-2698 v4 CPU. On this device, we still

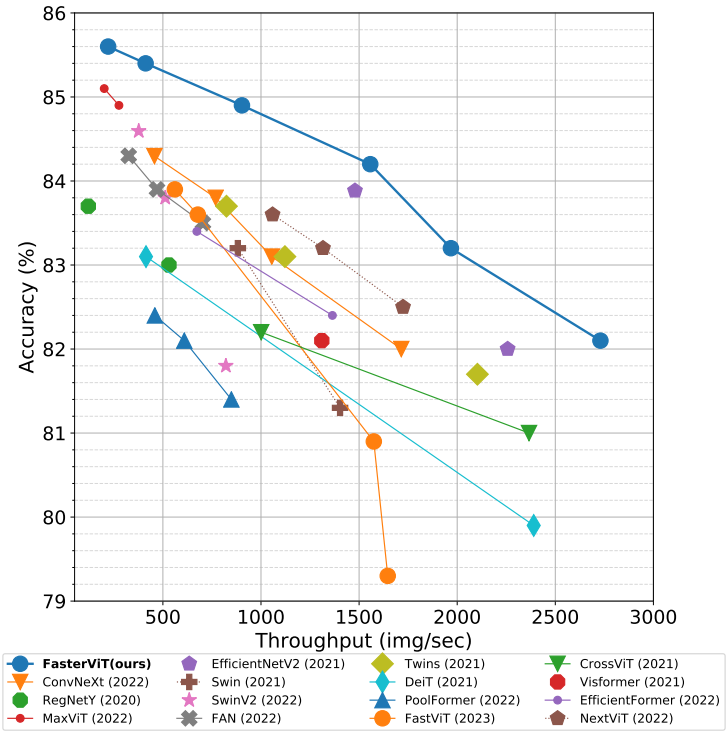


Figure S.6: Comparison of image throughput and ImageNet-1K Top-1 accuracy on NVIDIA TITAN RTX GPU for batch size of 128.

observe a dominant performance from different FasterViT variants. However, two variants from EfficientNetV2 and RegNetY models achieve a comparable performance to counterpart FasterViT models. In Fig. S.9, we present the throughput and accuracy tradeoff for NVIDIA Jetson Nano. Surprisingly, all FasterViT variants demonstrate a strong performance.

REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 4, 5

Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, et al. Hybrid task cascade for instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4974–4983, 2019. 6

Zhengsu Chen, Lingxi Xie, Jianwei Niu, Xuefeng Liu, Longhui Wei, and Qi Tian. Visformer: The vision-friendly transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 589–598, 2021. 4

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, 2021. 4

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*, pp. 248–255. Ieee, 2009. 1, 2

Jiawei Du, Zhou Daquan, Jiashi Feng, Vincent Tan, and Joey Tianyi Zhou. Sharpness-aware training for free. In *Advances in Neural Information Processing Systems*. 5

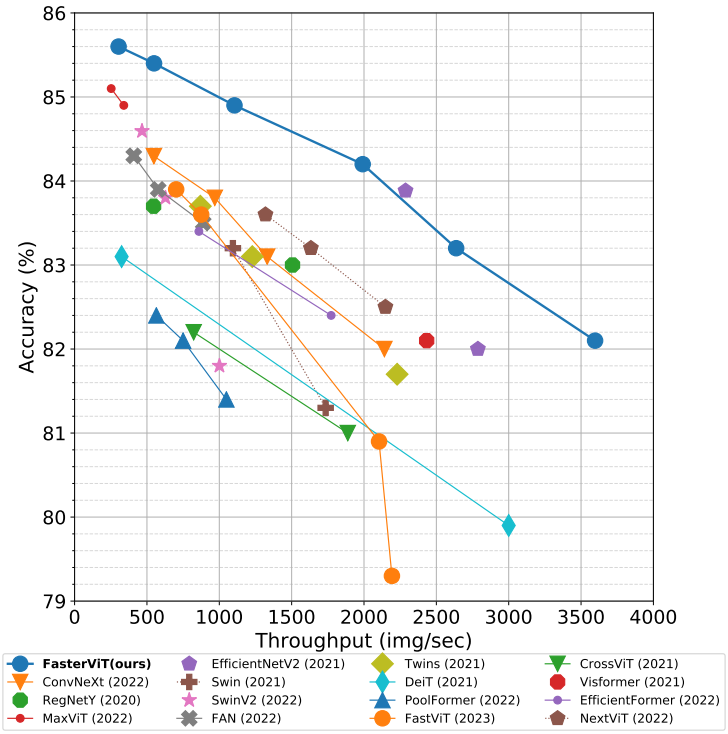


Figure S.7: Comparison of image throughput and ImageNet-1K Top-1 accuracy on NVIDIA A6000 GPU for batch size of 64.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020. 5

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017. 1

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8340–8349, 2021a. 1, 2

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15262–15271, 2021b. 1, 2

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 *ECCV*, 2014. 1, 6

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*, pp. 10012–10022, 2021. 1, 2, 4, 6

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*, pp. 12009–12019, 2022a. 2, 4, 5

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*, pp. 11976–11986, 2022b. 1, 2

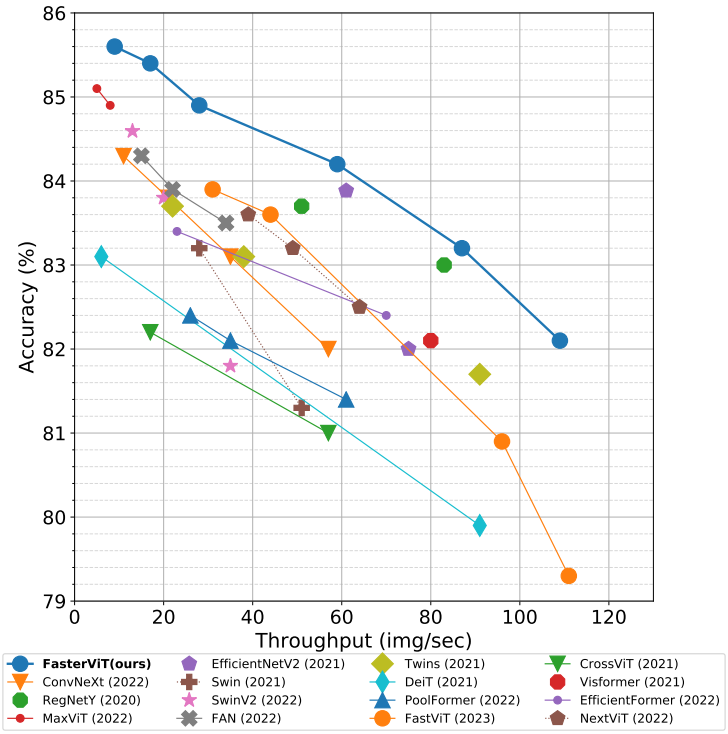


Figure S.8: Comparison of image throughput and ImageNet-1K Top-1 accuracy on Intel(R) Xeon(R) E5-2698 v4 CPU for batch size of 128.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 1, 4

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 *ECCV*, 2022. 4

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, pp. 5389–5400. PMLR, 2019. 1, 2

Hugo Touvron, Matthieu Cord, and Hervé Jégou. Deit iii: Revenge of the vit. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXIV*, pp. 516–533. Springer, 2022. 5

Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 418–434, 2018. 1

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962*, 2019. 1, 4

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*, pp. 10819–10829, 2022. 6

Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*, 2022. 6

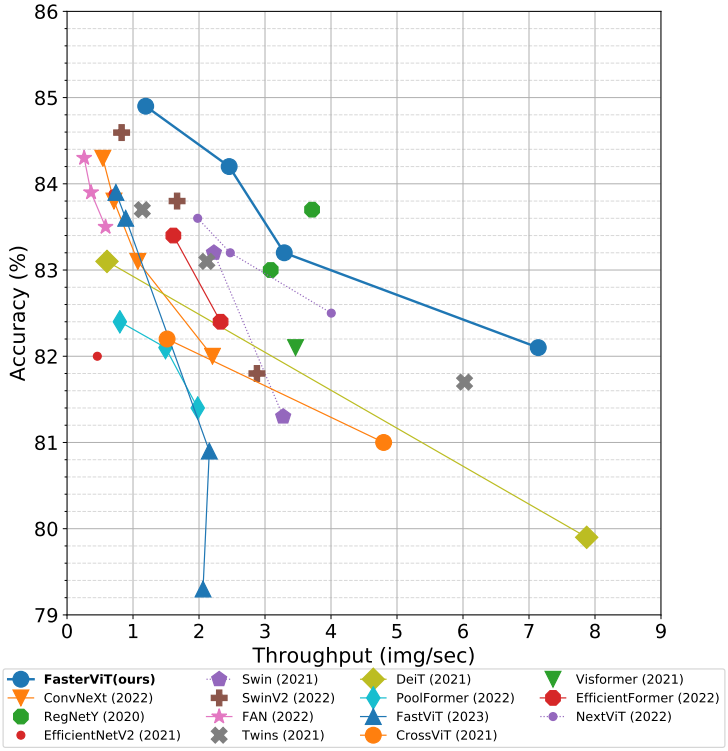


Figure S.9: Comparison of image throughput and ImageNet-1K Top-1 accuracy on NVIDIA Jetson Nano for batch size of 1.

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 633–641, 2017. 1, 6