Supplement Material

Abdelrahman Shaker[1*] Muhammad Maaz[1] Hanoona Rasheed[1] Salman Khan[1]
Ming-Hsuan Yang[2,3,4] Fahad Shahbaz Khan[1,5]
[1]Mohamed bin Zayed University of AI [2]University of California, Merced
[3]Yonsei University [4]Google Research [5]Linköping University

We provide additional details regarding:

- Architecture Details of SwiftFormer (Appendix 1)
- Implementation Details (Appendix 2)
- Additional Ablations (Appendix 3)
- Error Analysis on COCO Dataset (Appendix 4)
- Qualitative Results (Appendix 5)
- Discussion (Appendix 6)

1. Architecture Details of SwiftFormer

The detailed network architectures for SwiftFormer-XS, SwiftFormer-S, SwiftFormer-L1, and SwiftFormer-L3 are provided in Table 1. We report the resolution, the number of channels (C) and the number of repeated blocks (N) of each stage for all the model variants. For all variants, we use an expansion ratio of 4 in the Conv. Encoder. Since our architectures are not built using any neural architecture search, the number of channels and blocks for our models are selected to have a similar model size and GMACs with previous state-of-the-art methods in each variant.

2. Additional Implementation Details

We train and report the accuracy of our SwiftFormer models at 224×224 resolution for a fair comparison with the baseline and previous methods. We use a batch size of 2048 during training. The experiments for the SwiftFormer models were conducted on eight A100 GPUs, with an average training time of 36 hours for the classification. To enhance the robustness of the models, we apply several data augmentations during training. Specifically, we employ color jitter with a ratio of 0.4, RandAugment [1] with a magnitude of 9 and standard deviation of 0.5, gradient clipping of 0.01, Mixup [6] and Cutmix [5] with percentages of 1 and 0.8, respectively, label smoothing with a value of 0.1, and random erase with a probability of 0.25. Similar to EfficientFormer [2], we employ RegNetY-16GF [4] with 82.9% top-1 accuracy as our teacher model for hard distillation.

*Corresponding author: abdelrahman.youssief@mbzuai.ac.ae

3. Additional Ablations

In Table 2, we ablate the effect of the proposed SwiftFormer encoder at each stage gradually. We note that our SwiftFormer encoder can be incorporated in all stages without sacrificing the latency. In addition, it achieves consistent improvement in terms of top-1 accuracy across the stages. Furthermore, we investigate the effect of QKV interactions and observe that eliminating key-value interactions and replacing them with a simple linear transformation results in a 10% reduction in latency. In addition to latency reduction, the top-1 accuracy is improved by 0.4%.

4. Error Analysis on COCO Dataset

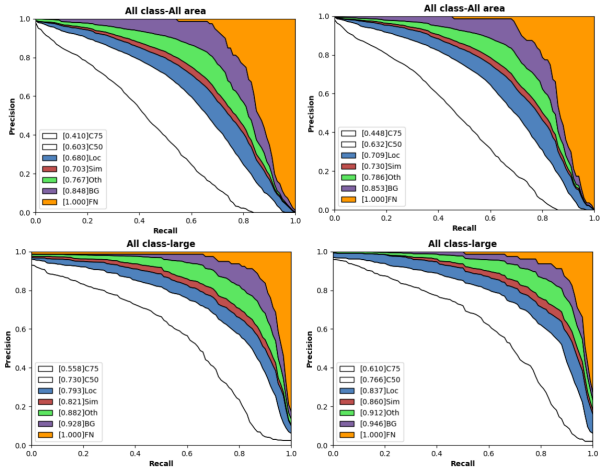

Figure 1: **Error analysis for the performance on COCO**. The baseline EfficientFormer-L1 (**left**) and our SwiftFormer-L1 (**right**) across all categories, on the all-objects (**top**) and large-sized objects (**bottom**). The plot in each image indicates a series of precision-recall curves using different evaluation configurations [3], with the legend indicating the area under each curve in brackets.

Fig. 1 shows the error analysis plot of the baseline EfficientFormer-L1 (left) and our SwiftFormer-L1 (right)

Stage	Output Resolution	Type	Config	SwiftFormer			
				XS	S	L1	L3
stem	$\frac{H}{2} \times \frac{W}{2}$	Patch Embed.	Patch Size	$k = 3 \times 3, s = 2$			
			Embed. Dim.	24	24	24	32
	$\frac{H}{4} \times \frac{W}{4}$	Patch Embed.	Patch Size,	$k = 3 \times 3, s = 2$			
			Embed. Dim.	48	48	48	64
1	$\frac{H}{4} \times \frac{W}{4}$	`Hybrid`	Conv. Encoder $[C, N]$	48, 2	48, 2	48, 3	64, 3
			SwiftFormer Encoder $[C, N]$	48, 1	48, 1	48, 1	64, 1
	$\frac{H}{8} \times \frac{W}{8}$	Down-sampling	Patch Size	$k = 3 \times 3, s = 2$			
			Embed. Dim.	56	64	96	128
2	$\frac{H}{8} \times \frac{W}{8}$	`Hybrid`	Conv. Encoder $[C, N]$	56, 2	64, 2	96, 2	128, 3
			SwiftFormer Encoder $[C, N]$	56, 1	64, 1	96, 1	128, 1
	$\frac{H}{16} \times \frac{W}{16}$	Down-sampling	Patch Size	$k = 3 \times 3, s = 2$			
			Embed. Dim.	112	168	192	320
3	$\frac{H}{16} \times \frac{W}{16}$	`Hybrid`	Conv. Encoder $[C, N]$	112, 5	168, 8	192, 9	320, 11
			SwiftFormer Encoder $[C, N]$	112, 1	168, 1	192, 1	320, 1
	$\frac{H}{32} \times \frac{W}{32}$	Down-sampling	Patch Size	$k = 3 \times 3, s = 2$			
			Embed. Dim.	220	224	384	512
4	$\frac{H}{32} \times \frac{W}{32}$	`Hybrid`	Conv. Encoder $[C, N]$	220, 3	224, 5	384, 4	512, 5
			SwiftFormer Encoder $[C, N]$	220, 1	224, 1	384, 1	512, 1
GMACs				0.6G	1.0G	1.6G	4.0G
Parameters				3.5M	6.1M	12.1M	28.5M

Table 1: **SwiftFormer Architectures.** Description of the configurations of the model variants with respect to the output resolution, the output channels C, the number of blocks N, and the model's GMACs and parameters. Between two consecutive stages, we incorporate a downsampling layer to increase the number of channels and reduce the resolution by two.

for all-objects and the large-sized objects. We show the area under each curve in brackets in the legend. It is noted that our results are better compared to the baseline, especially for large-sized objects. For instance, the overall AP of large-sized objects of EfficientFormer-L1 at IoU=0.75 is 0.558 and perfect localization increases the AP to 0.793. Excluding the background false positives likely increase the performance to 0.928 AP. In the case of SwiftFormer, the overall AP at IoU=0.75 is 0.610 and perfect localization increases the AP to 0.837. Further, excluding the background false positives likely increase the performance to 0.946 AP.

5. Qualitative Results

Fig. 2 and Fig. 3 show additional qualitative results of our SwiftFormer model for instance segmention/detection and semantic segmentation respectively. Our model accurately localizes and segments the objects in diverse scenes.

Figure 2: **Additional qualitative results on COCO.** Detection and instance segmentation results of our model.

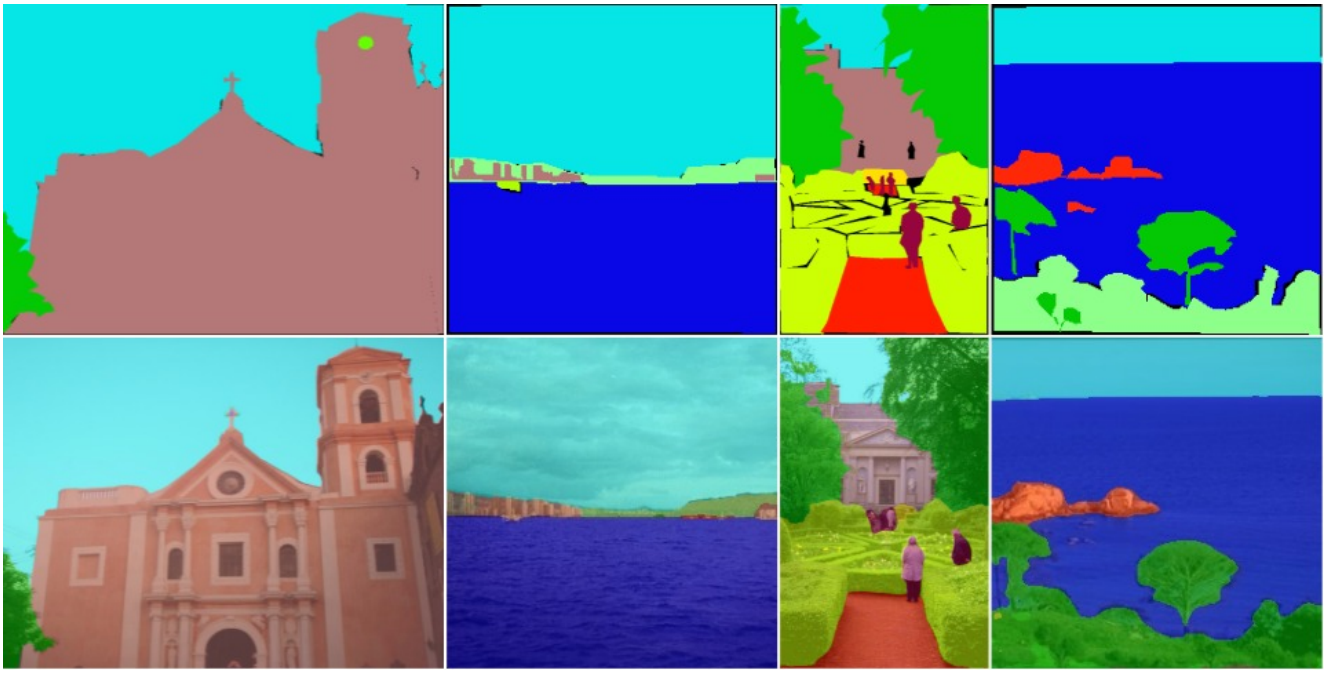

Figure 3: **Additional qualitative results on ADE20K.** Top row shows the ground truth masks and bottom row shows the predictions of our model.

Model configuration	Params	GMACs	Latency (ms)	Top-1 (%)
Conv=[4, 3, 10, 5], SwiftFormer=[0, 0, 0, 0]	10.9	1.4	0.9	79.9
Conv=[4, 3, 10, 4], SwiftFormer=[0, 0, 0, 1]	11.8	1.5	1.0	80.4
Conv=[4, 3, 9, 4], SwiftFormer=[0, 0, 1, 1]	12.0	1.5	1.0	80.7
Conv=[4, 2, 9, 4], SwiftFormer=[0, 1, 1, 1]	12.0	1.6	1.1	80.8
Conv=[3, 2, 9, 4], SwiftFormer=[1, 1, 1, 1]	12.1	1.6	1.1	80.9

Table 2: **Ablation on our SwiftFormer-L1 architecture**. Using one SwiftFormer encoder as the last block in all stages provides an optimal trade-off on the ImageNet-1k dataset. Latency is measured on iPhone14 Neural Engine.

6. Discussion

The positional encoding and attention biases in vision transformers both play a crucial role in providing spatial information about the input sequence, particularly in dense prediction tasks. However, the attention bias is sensitive to input resolution and can make the model fragile when incorporated into these tasks. Meanwhile, typical positional encoding can slow down the inference of the model on resource-constrained devices. We introduce an efficient additive attention mechanism that does not include positional encoding or attention biases, allowing for fast inference speed. To the best of our knowledge, SwiftFormer is the most efficient hybrid architecture for mobile applications.

References

[1] Ekin Dogus Cubuk, Barret Zoph, Jon Shlens, and Quoc Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Advances in Neural Information Processing Systems*, 2020.

[2] Yanyu Li, Geng Yuan, Yang Wen, Ju Hu, Georgios Evangelidis, Sergey Tulyakov, Yanzhi Wang, and Jian Ren. Efficientformer: Vision transformers at mobilenet speed. 2022.

[3] 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 *European Conference on Computer Vision*, 2014.

[4] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *CVPR*, 2020.

[5] 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*, 2019.

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