Neural Architecture Search for Mobile Semantic Segmentation

CS766 Spring Project, 2024

Presenters: Jidong Xu, Mingkai Wang, Ethan Fang

Introduction of NAS and semantic segmentation

What is Semantic Segmentation:

- In computer vision task, **semantic segmentation** is to classify each pixel in an image into a class or object.
- Aims to produce a dense pixel-wise segmentation map of an image and assign a specific class or object to each pixel.

Classes include: Pedestrian, Cars, Road, Train, Guideboard, etc.

Semantic segmentation

Existing solutions

CNN: high local feature extraction capability and computation efficiency, but limited global information capture

Vision Transformer (ViT): high global self-attention capability but lower computation efficiency

Image from: https://thegradient.pub/semantic-segmentation/

Introduction of NAS and semantic segmentation

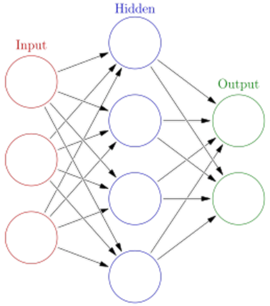
What is Neural Architecture Search (NAS):

- Neural Architecture Search (NAS) is a process that automates the design of neural network architectures within the field of machine learning.

Workflow of neural network:

Training Data

input



Neural Network

Model training

Test Data

input

Trained neural network model

output

Test results (eg. accuracy)

Use NAS to find good architectural hyperparameters

Architectural Hyper-parameters:

- Neural architectures (# of conv layers, # of conv kernels, kernel size etc.)

Search Space of NAS

Neural Network figure is from: https://en.wikipedia.org/wiki/Neural_network_(machine_learning)

Project objectives and challenges

Objective:

- Apply NAS to search for efficient semantic segmentation model on mobile and edge devices.

Main Challenges:

- How to design a search space for efficient semantic segmentation.

- How to design a search space that efficiently explores the proposed search space and obtains the optimal model for semantic segmentation.

Original Model – Topformer

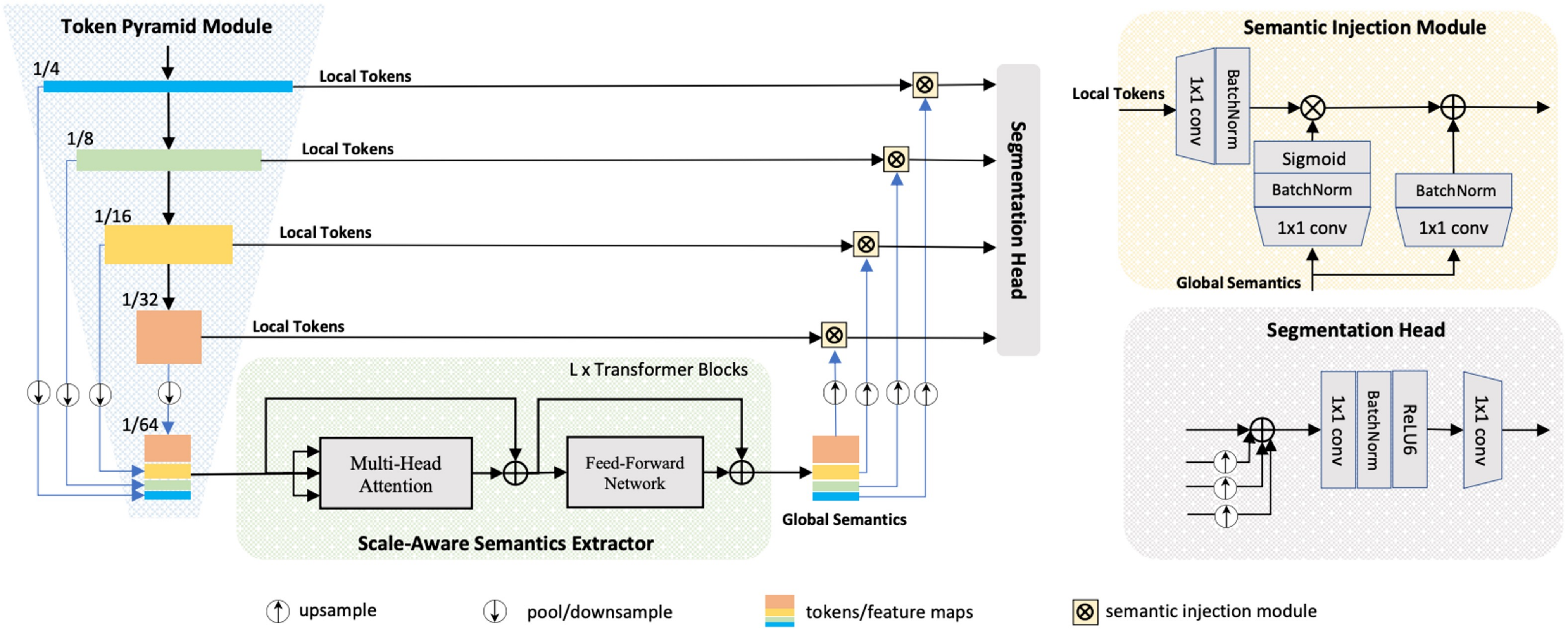


Figure 2 – The architecture of the proposed Token Pyramid Transformer.

Topformer: https://arxiv.org/pdf/2204.05525

Search Space

CNN Block	Width	Depth	Kernel size	Expansion ratio
Stem	{16, 24}	1	3	-
MBConv-1	{16, 24}	{1, 2}	{3, 5}	1
MBConv-2	(16, 40, 8)	{1, 2, 3}	{3, 5}	{3, 4, 5}
MBConv-3	(24, 72, 8)	{1, 2, 3}	{3, 5}	{2, 3, 4, 5}
MBConv-4	(56, 136, 8)	{1, 2, 3}	{3, 5}	{2, 3, 4, 5}
MBConv-5	(88, 176, 8)	{1, 2, 3, 4}	{3, 5}	{4, 5, 6, 7}

ViT Block	Number of heads	Key dim	Attention ratio	MLP ratio	Depth
ViT 1-4	(2, 12, 2)	(12, 20, 2)	(1.6, 2.4, 0.2)	(1.6, 2.4, 0.2)	{1, 2}

Table 1: The search space of Efficient-Topformer. Tuples of three values in parentheses represent the lowest value, the highest value, and steps. **Note:** Query dim = Key dim, Value dim = Attention ratio $\times$ Key dim.

Search Space and Search Pipeline

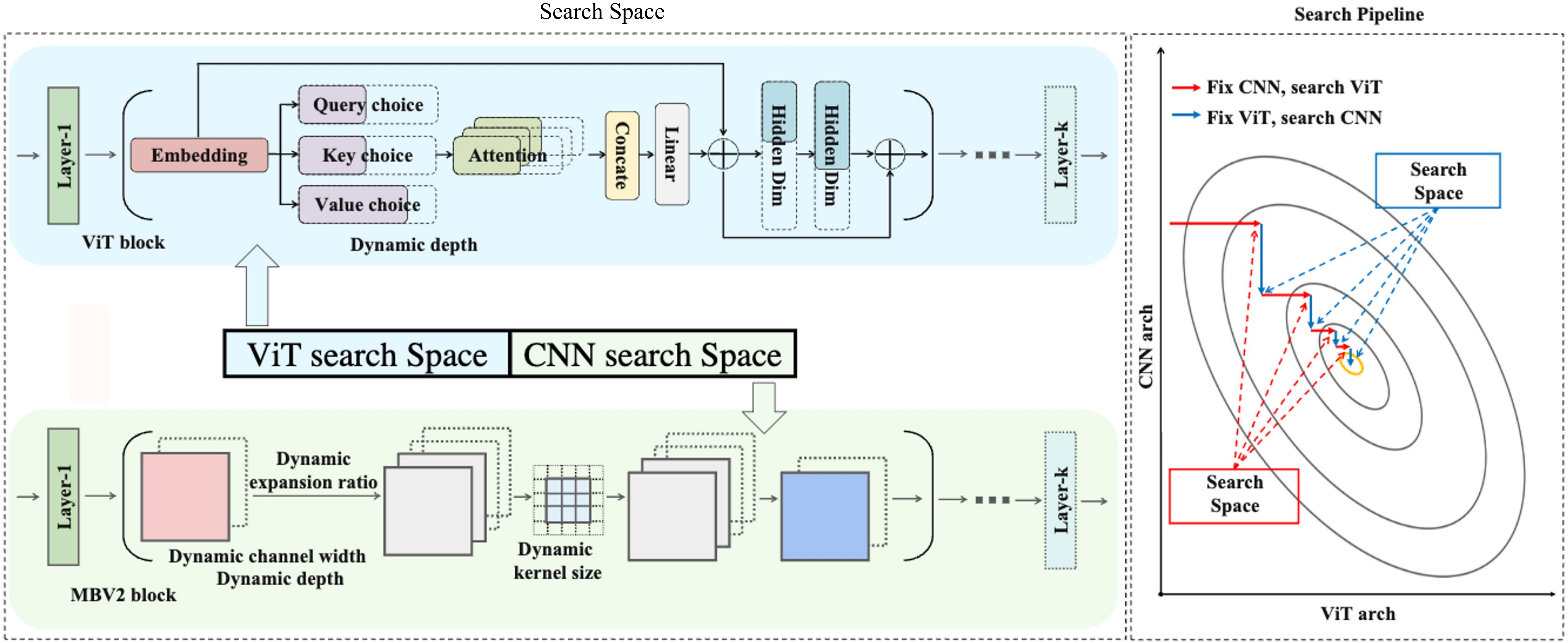


Figure 2: Overview of the proposed Efficient-Topformer. **Left**: the search space. **Right**: the search pipeline. The search space consists of CNN part and ViT part. In addition, we propose Coordinate Descend Search method to iteratively search for the optimal architecture.

Experiment and results

Dataset:
ADE20K [1] and COCO-Stuff [2]

Evaluation metrics:
mIoU: mean of class-wise intersection over union
FLOPs: floating point operations per second
Latency: measurements of inference time on the mobile device

Table 1: Results on ADE20K val set.

Method	Encoder	FLOPs(G)	mIoU(%)	Latency(ms)
PSPNet [31]	MobileNetV2	52.2	29.6	1065
FCN-8s [14]	MobileNetV2	39.6	19.7	1015
Semantic FPN [32]	ConvMLP-S	33.8	35.8	777
DeepLabV3+ [33]	EfficientNet	26.9	36.2	970
DeepLabV3+ [33]	MobileNetV2	25.8	38.1	1035
Lite-ASPP [33]	ResNet18	19.2	37.5	648
DeepLabV3+ [33]	ShuffleNetV2-1.5x	15.3	37.6	960
HRNet-W18-Small [34]	HRNet-W18-Small	10.2	33.4	639
Segformer [16]	MiT-B0	8.4	37.4	770
Lite-ASPP [33]	MobileNetV2	4.4	36.6	235
R-ASPP [27]	MobileNetV2	2.8	32.0	177
HR-NAS-B [10]	Searched	2.2	34.9	-
LR-ASPP [35]	MobileNetV3-Large	2.0	33.1	126
TopFormer [18]	TopFormer-B	1.8	37.8	110
HR-NAS-A [10]	Searched	1.4	33.2	-
LR-ASPP [35]	MobileNetV3-Large-reduce	1.3	32.3	81
TopFormer [18]	TopFormer-S	1.2	36.1	74
TopFormer [18]	TopFormer-T	0.6	32.8	43
TopFormer [18]	TopFormer-T*	0.5	32.5	32
Ours	**Efficient-Topformer-B**	**1.8**	**40.5** (+2.7)	**115**
Ours	**Efficient-Topformer-S**	**1.2**	**38.9** (+2.8)	**76**
Ours	**Efficient-Topformer-T**	**0.6**	**36.4** (+3.6)	**45**
Ours	**Efficient-Topformer-T***	**0.5**	**35.2** (+2.7)	**33**

Table 2: Results on COCO-Stuff val set.

Method	Backbone	FLOPs(G)	mIoU
PSPNet	MobileNetV2-s8	52.9	30.14
DeepLabV3+	MobileNetV2-s16	25.9	29.88
DeepLabV3+	EfficientNet-s16	27.1	31.45
LR-ASPP	MobileNetV3-s16	2.3	25.16
TopFormer	TopFormer-B	1.8	33.43
TopFormer	TopFormer-S	1.2	30.83
TopFormer	TopFormer-T	0.6	28.34
Ours	**Efficient-Topformer-B**	**1.8**	**34.64** (+1.21)
Ours	**Efficient-Topformer-S**	**1.2**	**32.92** (+2.09)
Ours	**Efficient-Topformer-T**	**0.6**	**30.43** (+2.09)

[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, pages 633–641, 2017.
[2] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1209–1218, 2018.

Experiment and results

Visualization on ADE20K validation (val) set :

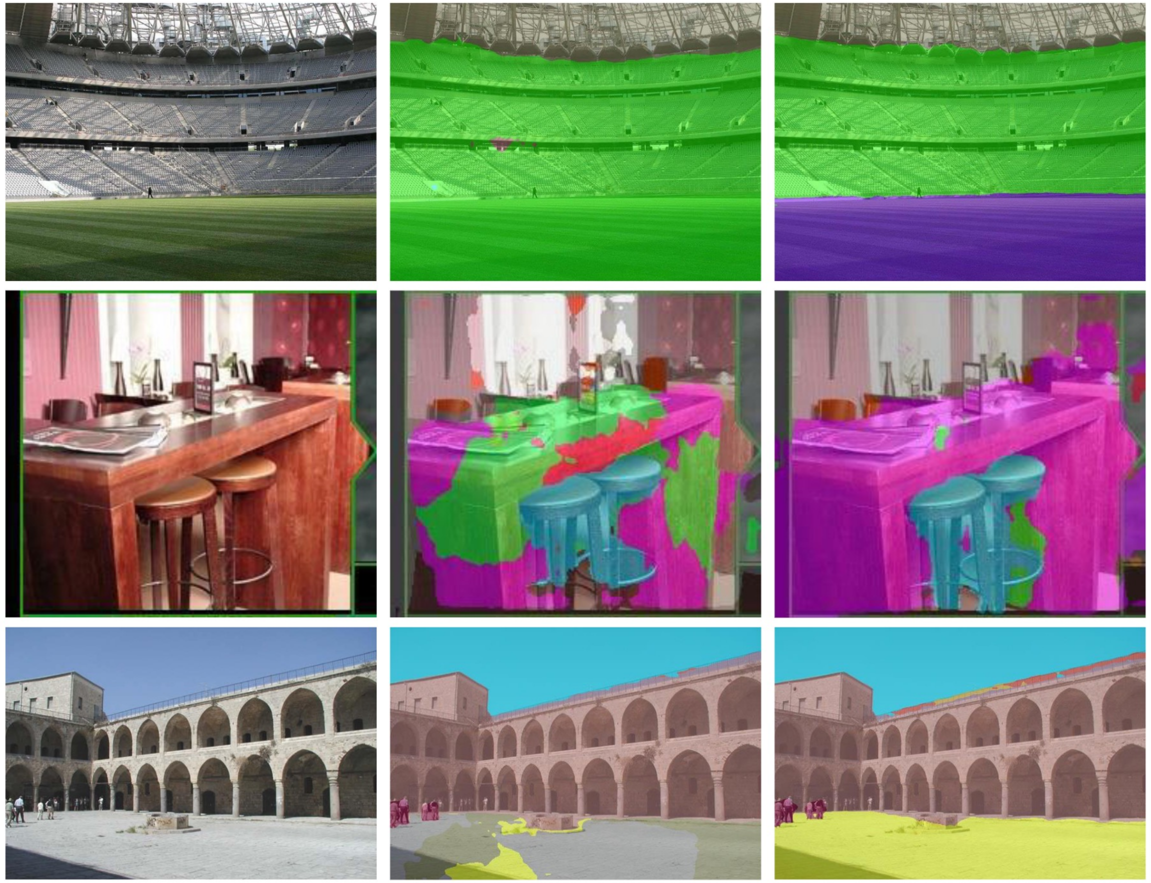


Image TopFormer-B Ours

Conclusion

- Successfully propose a novel architecture search method for efficient semantic segmentation, named **Efficient-Topformer**
- Propose a search space that takes advantage of CNN and ViT simultaneously.
- Propose a Coordinate Descent Search method, which is beneficial to search for the optimal architecture in the aforementioned search spaces.

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON