NeurIPS 2020

Rethinking Learnable Tree Filter for Generic Feature Transform

宋林 (Lin Song)
stevengrove@stu.xjtu.edu.cn

Introduction

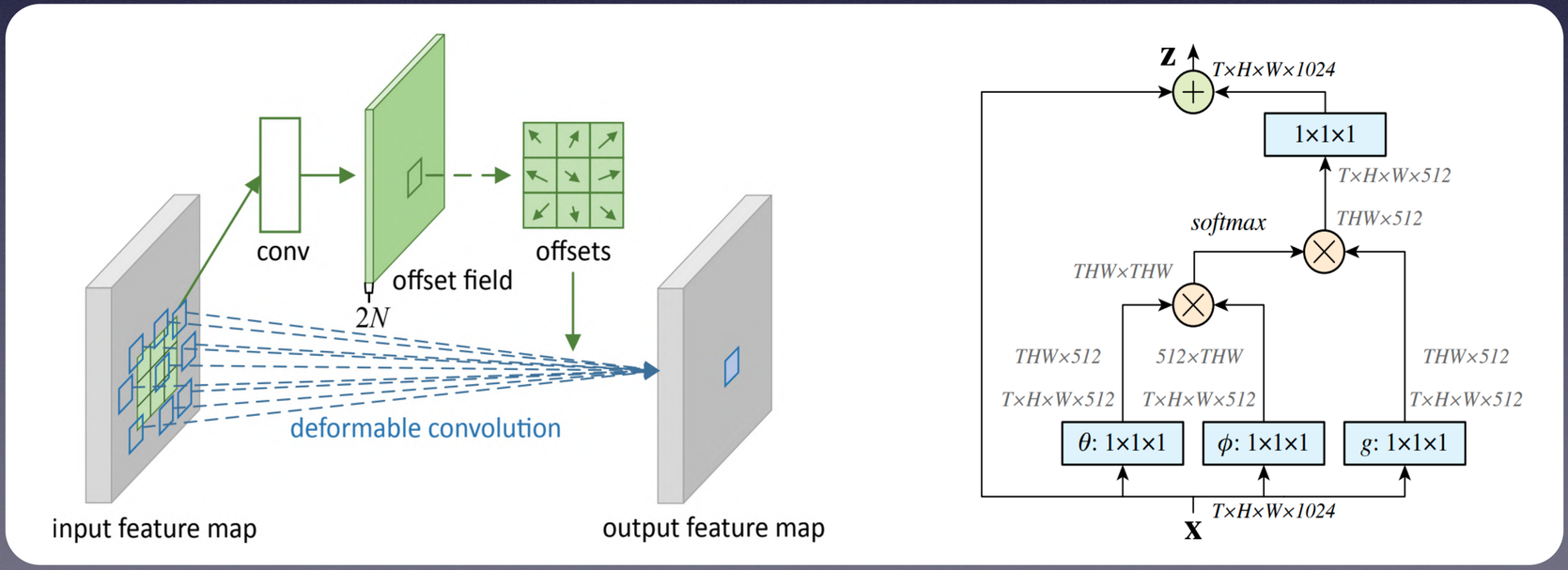
Conventional methods for visual context modeling

Local-based: *increase the receptive region of convolutional kernels*

e.g., Dilated Convolution, Deformable Convolution, ASPP, PSP

Global-based: *modeling the pairwise relations based on the visual attention mechanism*

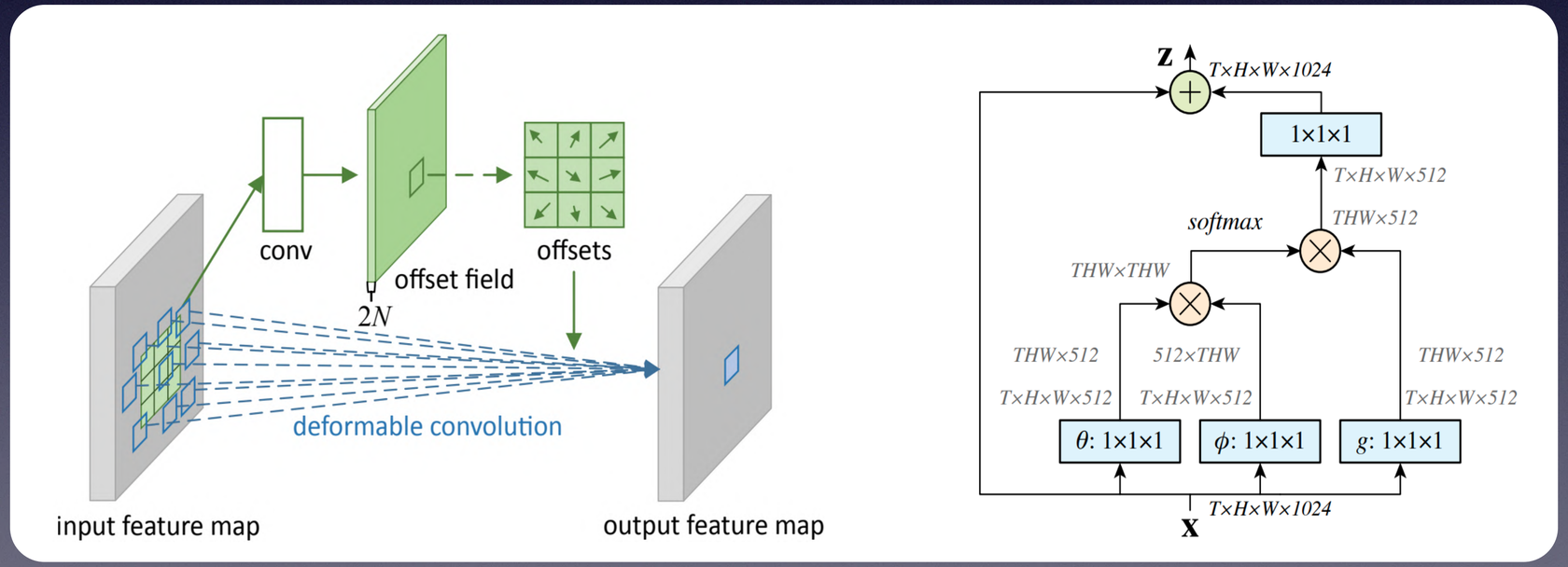
e.g., Non-Local, CCNet, LatentGNN



Introduction

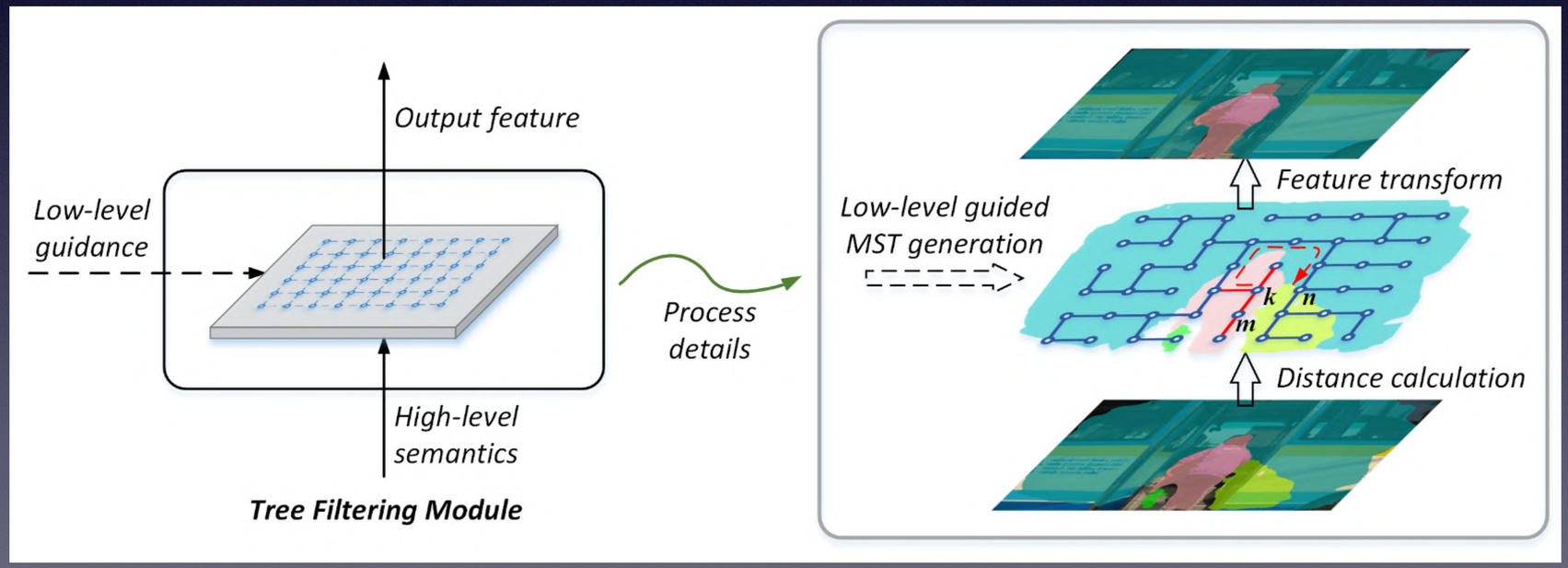
Conventional methods for visual context modeling

*However, there is still a conflict between **long-range** dependencies modeling and **object details** preserving.*



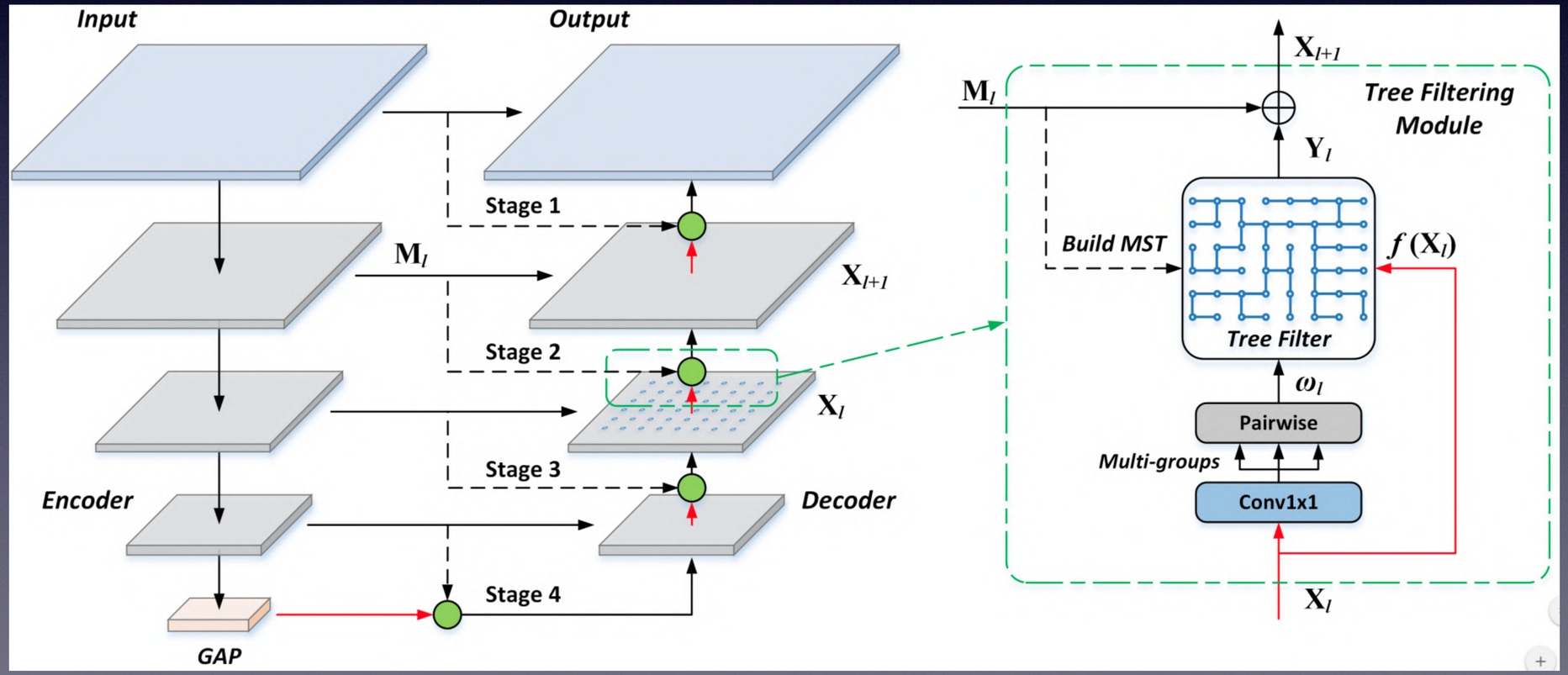
LTF-V1: Learnable Tree Filter for Structure-preserving Feature Transform

LTF-V1 uses the minimum spanning tree generated by the low-level guided features, which can ***retain the structural details*** *with* ***linear complexity w.r.t. vertex number***



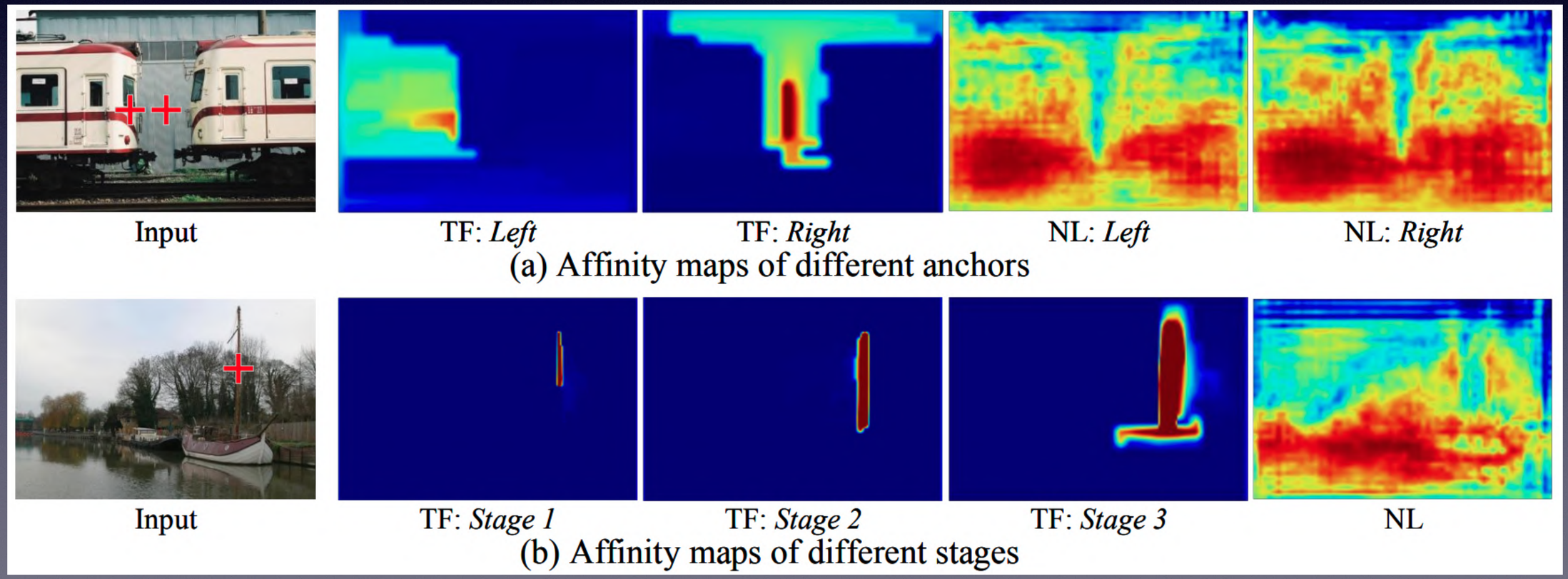
LTF-V1: Learnable Tree Filter for Structure-preserving Feature Transform

*LTF-V1 is a **differentiable** and **plug-and-play** module*



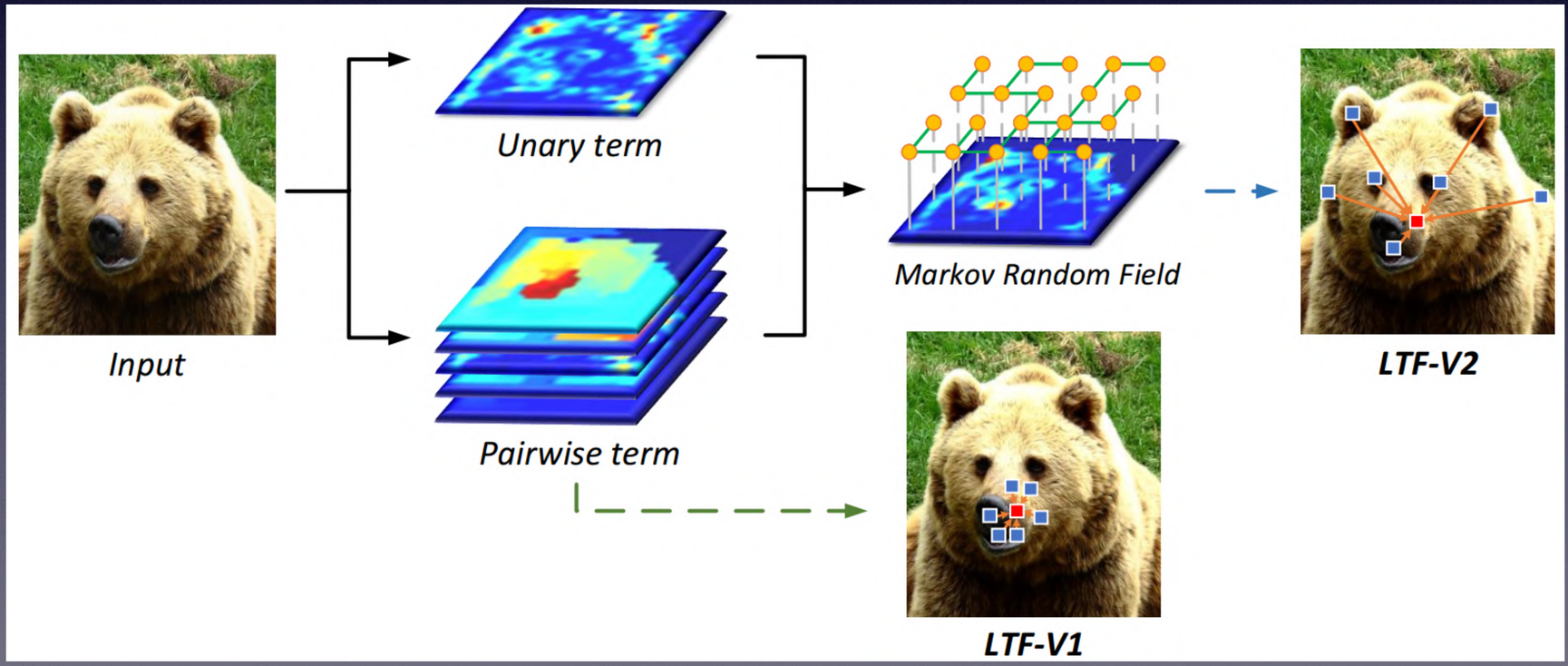
LTF-V1: Learnable Tree Filter for Structure-preserving Feature Transform

*LTF-V1 can model **higher-order relations**, which is crucial for feature discriminability*



LTF-V1: Learnable Tree Filter for Structure-preserving Feature Transform

*Nevertheless, the **geometric constraint** and the **non-differentiable spanning tree construction** in the LTF-V1 module are found to be its Achilles' heels, which impede the usage for more generic feature transform*



Reformulation of LTF-V1

The statistical expectation of the sampling from input features under a specific distribution:

$$y_i = \mathbb{E}_{h_i \sim P_{\mathcal{G}_T}}[x_{h_i}] = \sum_{\forall j \in \mathcal{V}} P_{\mathcal{G}_T}(h_i = j) x_j.$$

The distribution is modeled by Markov Random Field:

$$P_{\mathcal{G}_T}(\mathbf{H}|\mathbf{O} = X) = \frac{1}{Z} \prod_{\forall i \in \mathcal{V}} \phi_i(h_i, x_i) \prod_{\forall (i,j) \in \mathcal{E}} \psi_{i,j}(h_i, h_j)$$

The distribution for LTF-V1 can be considered as a specific Markov Random Field:

$$\phi_i(h_i, x_i) \equiv 1$$

$$\psi_{i,j}(h_i, h_j) := \begin{cases} \delta(h_i - h_j) & h_i \notin \mathrm{Desc}_{\mathcal{G}_T}(i,j) \\ \exp(-\omega_{i,j})\delta(h_i - h_j) & h_i \in \mathrm{Desc}_{\mathcal{G}_T}(i,j) \end{cases}$$

Reformulation of LTF-VI

The statistical expectation of the sampling from input features under a specific distribution:

$$y_i = \mathbb{E}_{h_i \sim P_{\mathcal{G}_T}}[x_{h_i}] = \sum_{\forall j \in \mathcal{V}} P_{\mathcal{G}_T}(h_i = j) x_j.$$

The distribution is modeled by Markov Random Field:

$$P_{\mathcal{G}_T}(\mathbf{H}|\mathbf{O} = X) = \frac{1}{Z} \prod_{\forall i \in \mathcal{V}} \phi_i(h_i, x_i) \prod_{\forall (i,j) \in \mathcal{E}} \psi_{i,j}(h_i, h_j)$$

The distribution for LTF-VI can be considered as a specific Markov Random Field:

$\phi_i(h_i, x_i) \equiv 1$ ***Constant modeling*** *and* ***geometric constraint*** *force the LTF-VI to focus on the nearby region, leading to the difficulty of long-range interactions*

$$\psi_{i,j}(h_i, h_j) := \begin{cases} \delta(h_i - h_j) & h_i \notin \mathrm{Desc}_{\mathcal{G}_T}(i,j) \\ \exp(-\omega_{i,j})\delta(h_i - h_j) & h_i \in \mathrm{Desc}_{\mathcal{G}_T}(i,j) \end{cases}$$

Proposed LTF-V2

We use data-dependent modeling for unary term:

$$\phi_i(h_i, x_i) := \begin{cases} f(x_i) & h_i = i \\ \exp(-\beta) & h_i \neq i \end{cases}$$

$$\psi_{i,j}(h_i, h_j) := \begin{cases} \delta(h_i - h_j) & h_i \notin \mathrm{Desc}_{\mathcal{G}_T}(i,j) \\ \exp(-\omega_{i,j})\delta(h_i - h_j) & h_i \in \mathrm{Desc}_{\mathcal{G}_T}(i,j) \end{cases}$$

Proposed LTF-V2

We use data-dependent modeling for unary term:

$$\phi_i(h_i, x_i) := \begin{cases} f(x_i) & h_i = i \\ \exp(-\beta) & h_i \neq i \end{cases}$$

$$\psi_{i,j}(h_i, h_j) := \begin{cases} \delta(h_i - h_j) & h_i \notin \mathrm{Desc}_{\mathcal{G}_T}(i,j) \\ \exp(-\omega_{i,j})\delta(h_i - h_j) & h_i \in \mathrm{Desc}_{\mathcal{G}_T}(i,j) \end{cases}$$

We can derive the closed-form solution by using belief propagation algorithm:

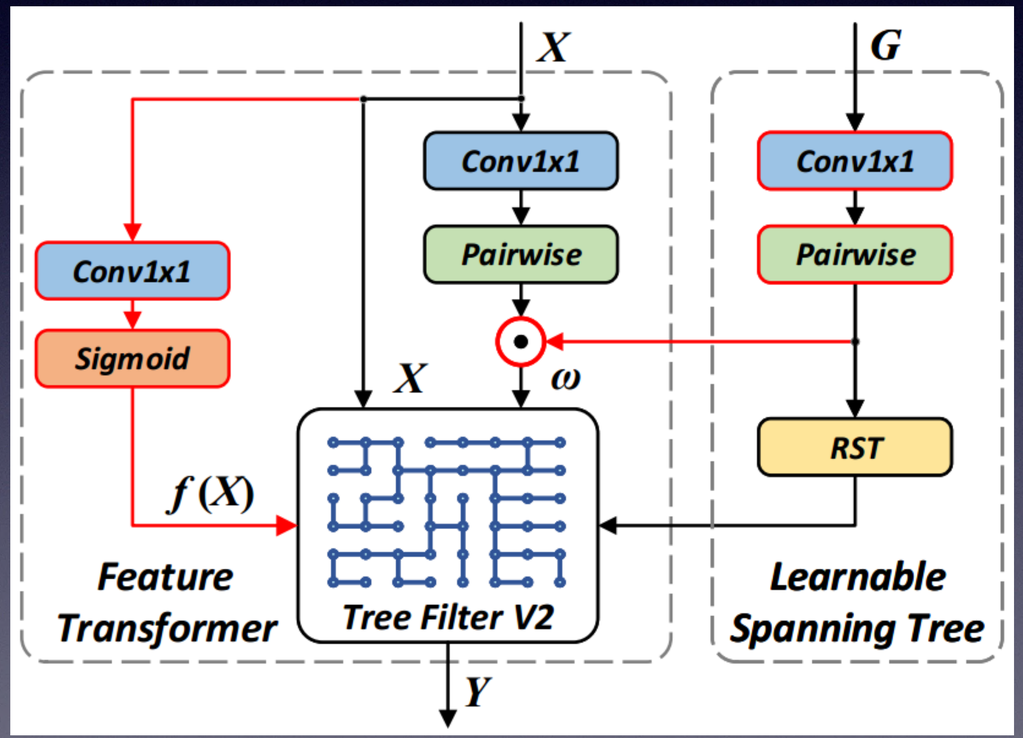
$$y_i = \frac{1}{z_i} \sum_{\forall x_j \in X} S_{\mathcal{G}_T}(E_{j,i}) \exp(-\beta)^{|E_{j,i}|} f(x_j) x_j$$

***Learnable Tree Filter V2 (LTF-V2)**:*
relaxes the geometric constraint and enables efficient long-range interactions

Our Method

Framework of Learnable Tree Filter V2 Module

*The learnable spanning tree module enables **fully end-to-end training**:*



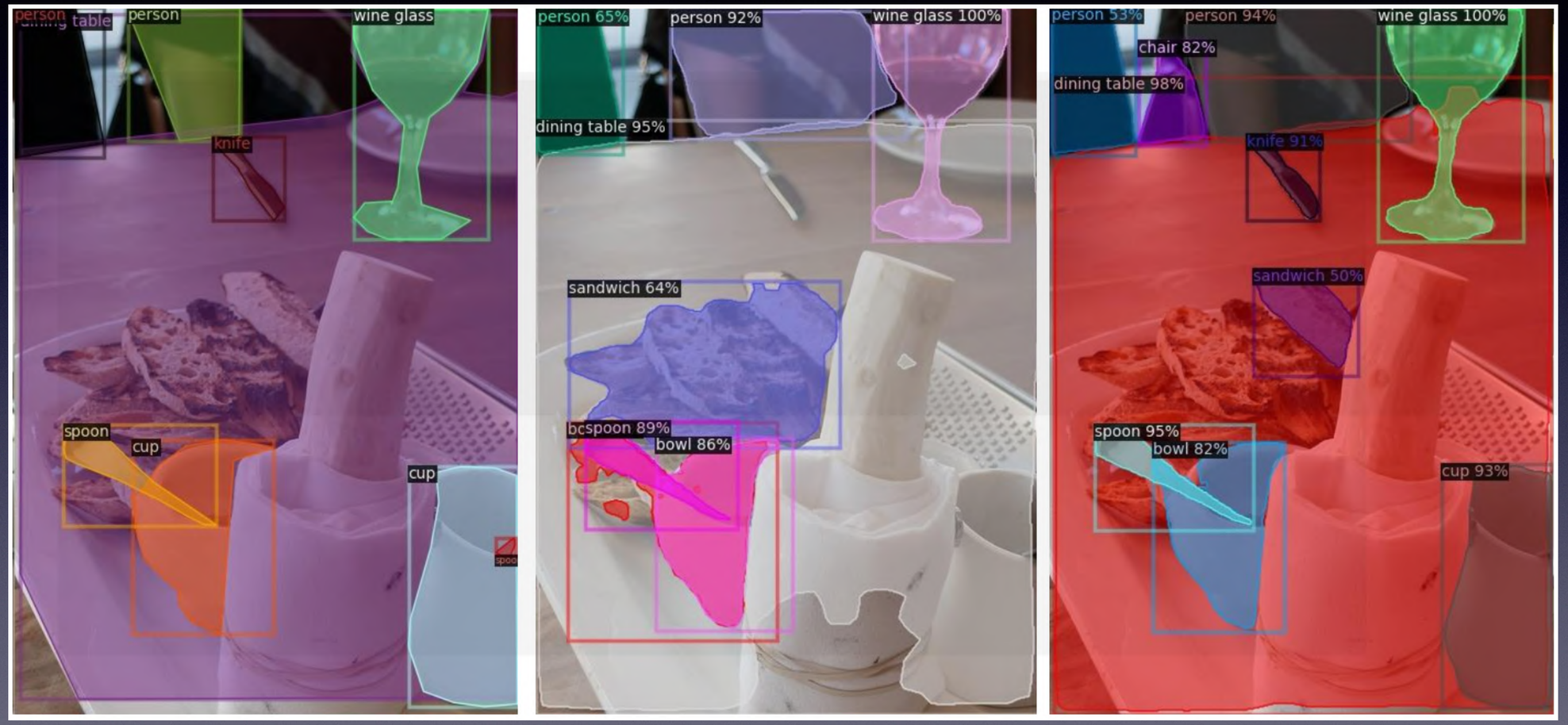
Algorithm 1: Close random spanning tree

Input: A 4-connected graph $\mathcal{G}$.
Output: Random spanning tree $\mathcal{G}_T$.

$\mathcal{G}_T \leftarrow \emptyset$.
for $e \in E(\mathcal{G})$ **do**
  $l(e) \leftarrow e$.
while $|V(\mathcal{G})| > 1$ **do**
  for $v_i \in V(\mathcal{G})$ **do**
    $e_i \sim E_{\mathcal{G}}(v_i)$.
    $\mathcal{G}_T \leftarrow \mathcal{G}_T \cup \{l(e_i)\}$.
  Contract$(E(\mathcal{G}))$.
  Flatten$(\mathcal{G})$.
return $\mathcal{G}_T$.

Experiments

Visualization on Instance Segmentation/Object Detection



Visualization of ground-truth (left), vanilla mask-rcnn (mid) and learnable tree filter (right) on COCO val set

Visualization on Semantic Segmentation

Visulaization of learnable tree filter (middle row) and ground-truth (bottom row) on VOC2012 val set

Experiments

Ablation studies for instance-aware tasks

*Compared with previous work, our method (LTF-V2) achieves **higher** performance with much **less** resource consumption on COCO dataset*

Ablation study on COCO2017 val set

Model	Stage	AP_{box}	AP^{50}_{box}	AP^{75}_{box}	AP_{seg}	AP^{50}_{seg}	AP^{75}_{seg}	#FLOPs	#Params
ResNet-50 (1x)	-	38.8	58.7	42.4	35.2	55.6	37.6	279.4B	44.4M
+Non-Local [11]	4	39.5	59.6	42.7	35.6	56.7	37.6	+10.67B	+2.09M
+CCNet [12]	345	40.1	60.4	44.1	36.0	57.4	38.4	+16.62B	+6.88M
+LatentGNN [14]	345	40.6	61.3	44.5	36.6	58.1	39.2	+3.59B	+1.07M
+GCNet [15]	All	40.7	61.0	44.2	36.7	58.1	39.2	+0.35B	+10.0M
+LTF-V1	345	40.0	60.4	43.7	36.1	57.5	38.4	+0.31B	+0.06M
+LTF-V2	3	40.1	59.9	43.9	36.0	57.1	38.3	+0.43B	+0.02M
	4	40.6	61.0	44.4	36.6	58.2	39.0	+0.26B	+0.04M
	5	40.2	60.5	43.6	36.1	57.5	38.4	+0.17B	+0.08M
	345	**41.2**	**61.6**	**45.2**	**37.0**	**58.4**	**39.5**	+0.68B	+0.14M

Experiments

Ablation studies for semantic segmentation

*Compared with the state-of-the-art work, our method (LTF-V2) based on a simple FPN architecture achieves **superior performance** on Cityscapes dataset*

Ablation study on Cityscapes val set (fine only)

Model	MS&Flip	mIoU (%)	mAcc (%)
ResNet-50	✗	72.9	95.5
+LTF-V1	✗	75.9	95.8
+LTF-V2	✗	**77.4**	**96.0**
ResNet-50	✓	75.5	95.9
+LTF-V1	✓	77.2	96.0
+LTF-V2	✓	**78.9**	**96.1**

Results on Cityscapes test set (fine only)

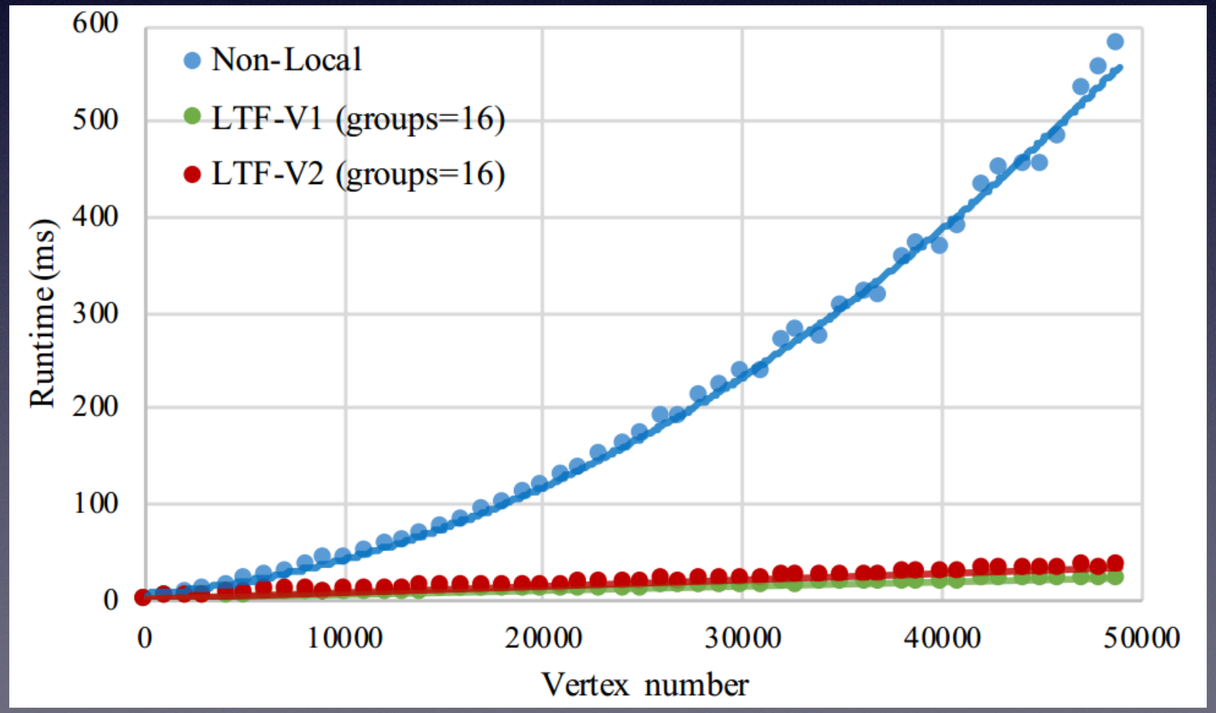
Model	Backbone	mIoU (%)
PSPNet [9]	ResNet-101	78.4
DFN [39]	ResNet-101	79.3
DenseASPP [50]	DenseNet-161	80.6
LTF-V1 [16]	ResNet-101	80.8
CCNet [12]	ResNet-101	81.4
DANet [51]	ResNet-101	81.5
SPNet [52]	ResNet-101	82.0
Ours (LTF-V2)	ResNet-101	**82.1**

Experiments

Empirical Runtime

For **GPU** devices, we parallelize the algorithm along with **batches**, **channels**, and **nodes of the same depth.**

Learnable Tree Filter is easy to use. You only need to add **2 lines** to your PyTorch code.

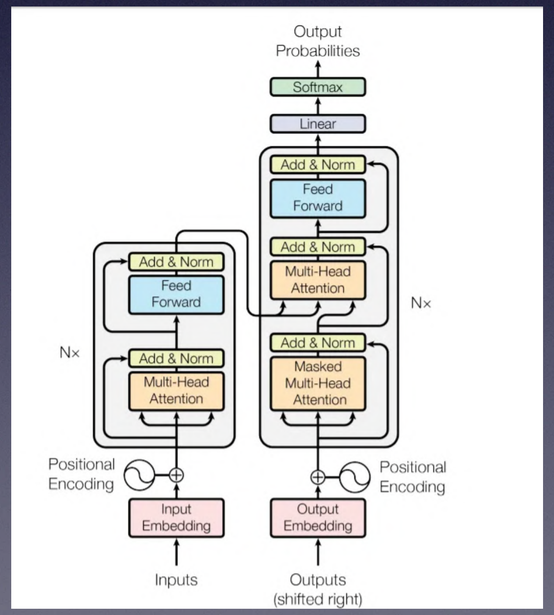


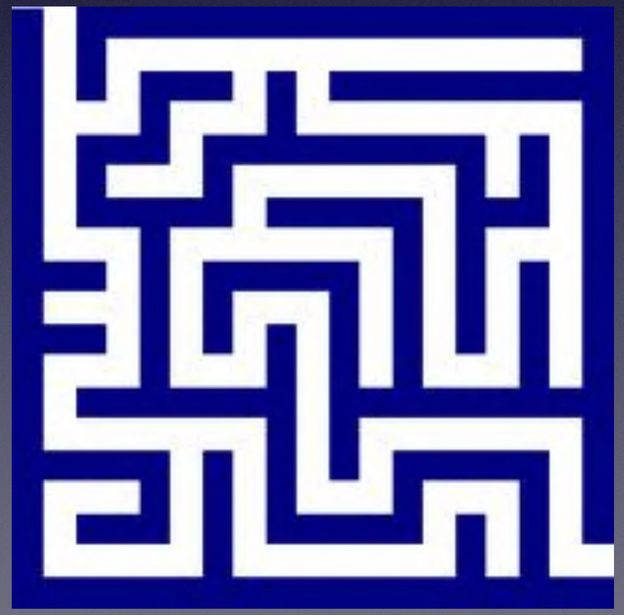
Runtime on a Tesla V100 GPU

Future Work

Potential Applications

- *Replacing the attention module of **transformer** for natural language processing*
- *Efficiently modeling **higher-order relations**, e.g., solving maze problem*
- *Enhancing sequential representation for **video analysis***





Thanks

For any questions, please contact

stevengrove@stu.xjtu.edu.cn

Source code is available

https://github.com/Megvii-BaseDetection/TreeFilter-Torch

https://github.com/StevenGrove/LearnableTreeFilterV2