THE UNIVERSITY OF HONG KONG
DEPARTMENT OF COMPUTER SCIENCE

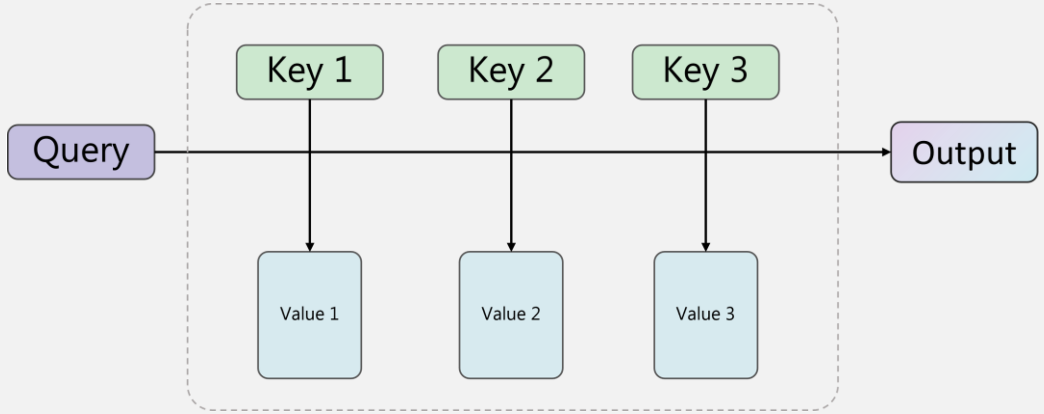
# Ripple Attention for Visual Perception with Sub-quadratic Complexity

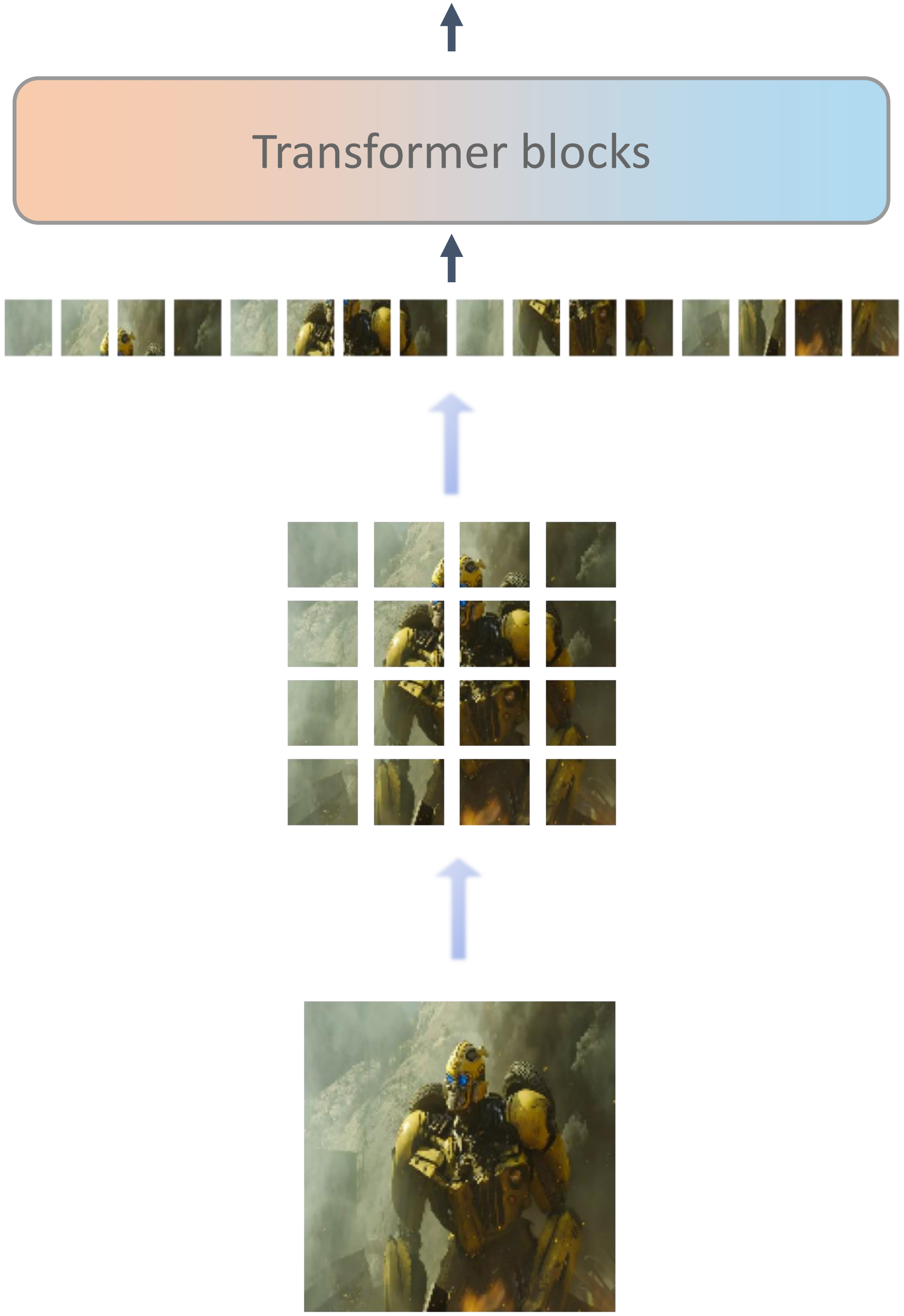
Lin Zheng, Huijie Pan, Lingpeng Kong

# Vision Transformers

## Attention Mechanism

$$\mathrm{Attn}\left(\mathbf{q}_n, \{\mathbf{k}_m\}, \{\mathbf{v}_m\}\right) = \sum_m \frac{\exp\left(\mathbf{q}_n^\top \mathbf{k}_m\right)}{\sum_{m'} \exp(\mathbf{q}_n^\top \mathbf{k}_{m'})} \mathbf{v}_m^\top$$



# Problems

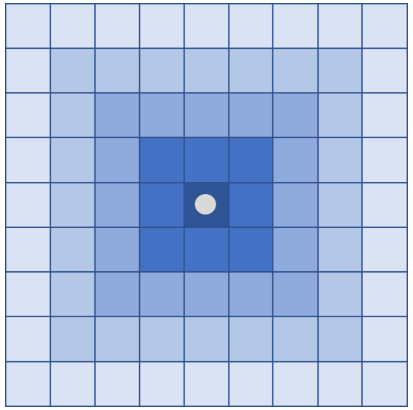
✗ **Scattered** visual context:

- Flattening 2D images to 1D sequences undermines the inherent local correlations of images, which often bear important visual clues.

✗ **Quadratic** time/space complexity**:**

- Prohibitive to process higher-resolution images or smaller patch sizes**.**

# Ripple Attention



- We propose ripple attention, an efficient mechanism that
  - incorporates the notion of **spatial vicinity** into the transformer, and
  - runs with **sub-quadratic** complexity.
- In ripple attention, contributions of different patches to a query are reweighted with respect to their spatial distances in the 2D space.
- Built upon linearized attention, we develop a dynamic programming algorithm to execute ripple attention in linear observed time.

# Linearized Attention

- The key idea of linearization is using dot-product of feature maps to approximate the exponential kernels:

$$\kappa(\mathbf{x},\mathbf{y}) := \exp(\mathbf{x}^\top \mathbf{y}) \approx \phi(\mathbf{x})^\top \phi(\mathbf{y}), \quad \phi : \mathbb{R}^d \to \mathbb{R}^D$$

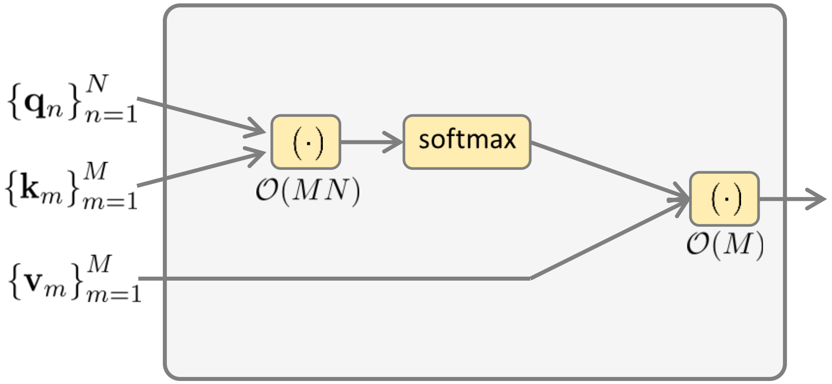
- Plugging in such approximation yields Linearized Attention (LA):

$$\sum_m \frac{\exp\left(\mathbf{q}_n^\top \mathbf{k}_m\right)}{\sum_{m'} \exp\left(\mathbf{q}_n^\top \mathbf{k}_{m'}\right)} \mathbf{v}_m^\top \approx \sum_m \frac{\phi\left(\mathbf{q}_n\right)^\top \phi\left(\mathbf{k}_m\right) \mathbf{v}_m^\top}{\sum_{m'} \phi\left(\mathbf{q}_n\right)^\top \phi\left(\mathbf{k}_{m'}\right)} := \mathrm{LA}\left(\mathbf{q}_n, \{\mathbf{k}_m\}, \{\mathbf{v}_m\}\right)$$

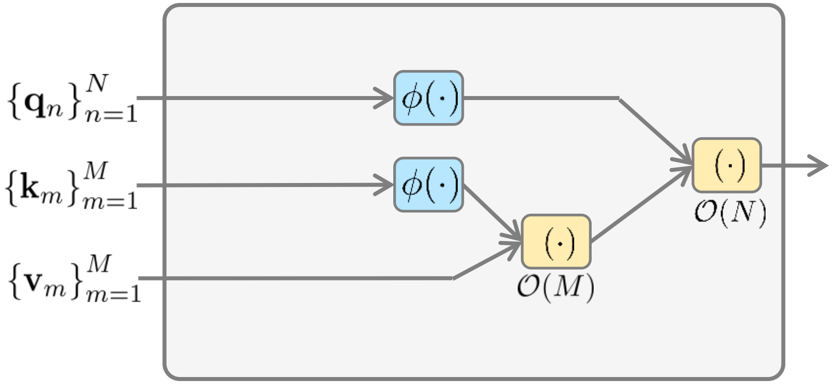
# Linearized Attention

- LA achieves **linear** complexity due to the re-order of computation.
- Reduce complexity from $\mathcal{O}(MN)$ to $\mathcal{O}(M+N)$.

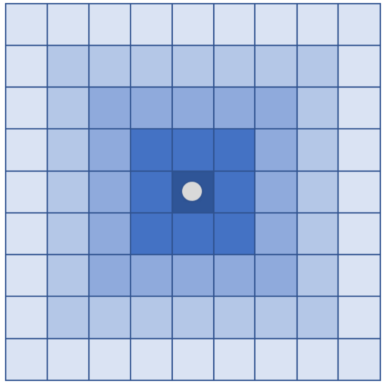
$$\sum_{m} \frac{\phi(\mathbf{q}_n)^\top \phi(\mathbf{k}_m)\mathbf{v}_m^\top}{\sum_{m'} \phi(\mathbf{q}_n)^\top \phi(\mathbf{k}_{m'})} = \frac{\phi(\mathbf{q}_n)^\top \sum_{m=1}^{M} \phi(\mathbf{k}_m) \otimes \mathbf{v}_m}{\phi(\mathbf{q}_n)^\top \sum_{m'=1}^{M} \phi(\mathbf{k}_{m'})}$$




Vanilla softmax attention




Linearized attention



# Reformulation with Vicinal Groups

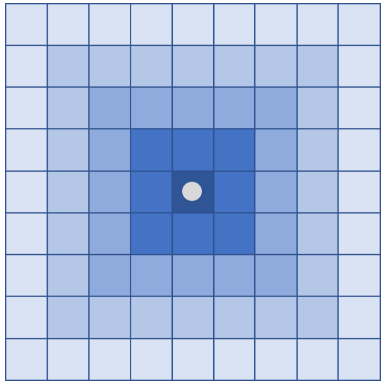
- We partition the grid $H \times W$ into $R+1$ **vicinal groups** $\mathcal{N}_0(i,j), \ldots, \mathcal{N}_r(i,j), \mathcal{N}_R(i,j)$ according to a reference index $(i,j)$, where

$$\max(|m-i|, |n-j|) = r, \quad \forall (m,n) \in \mathcal{N}_r(i,j)$$

- The vicinal group reflects the spatial vicinity for each index.
- The linearized attention can be expressed as sum over individual vicinal groups:

$$\frac{\phi\left(\mathbf{q}_{ij}\right)^{\top} \sum_{r=0}^{R} \sum_{(m,n)\in\mathcal{N}_r(i,j)} \phi\left(\mathbf{k}_{mn}\right) \otimes \mathbf{v}_{mn}}{\phi\left(\mathbf{q}_{ij}\right)^{\top} \sum_{r=0}^{R} \sum_{(m',n')\in\mathcal{N}_r(i,j)} \phi\left(\mathbf{k}_{m'n'}\right)}$$

# Reweighting Vicinal Groups



- We associate each vicinal group with a scalar $\alpha_r(i,j)$
  - to **reweight the contribution** from different groups according to their relative distances.

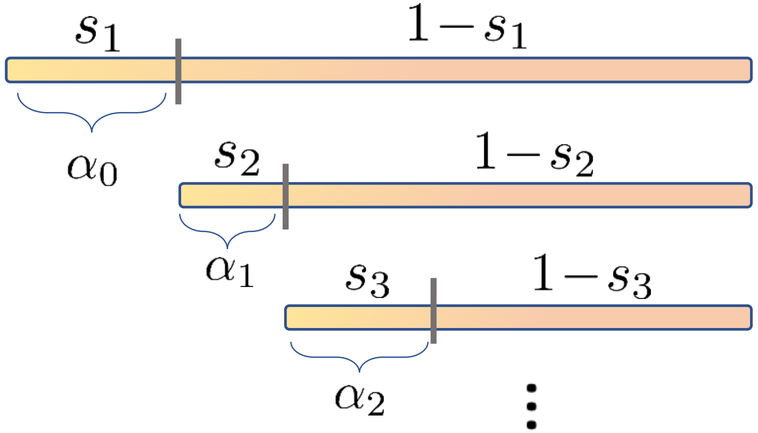
$$\operatorname{RippleAttn}\left(\mathbf{q}_{ij}, \{\mathbf{k}_{mn}\}, \{\mathbf{v}_{mn}\}\right) := \frac{\phi\left(\mathbf{q}_{ij}\right)^{\top} \sum_{r=0}^{R} \alpha_r(i,j) \sum_{(m,n)\in\mathcal{N}_r(i,j)} \phi\left(\mathbf{k}_{mn}\right) \otimes \mathbf{v}_{mn}}{\phi\left(\mathbf{q}_{ij}\right)^{\top} \sum_{r=0}^{R} \alpha_r(i,j) \sum_{(m',n')\in\mathcal{N}_r(i,j)} \phi\left(\mathbf{k}_{m'n'}\right)}$$

- The weighting scheme $\{\alpha_r(i,j)\}$ is defined via **stick breaking transform,**
  - promotes local correlations but still captures long-range dependencies.

Initialize a sequence of scalars $s_r \in (0,1) \quad \forall r = 1, \ldots, R$;

$$\text{Set } \alpha_r = \begin{cases} s_1, & \text{if } r = 0 \\ s_{r+1} \prod_{r' \leq r} \left(1 - s_{r'}\right), & \text{otherwise} \end{cases}$$

We have $\sup \alpha_r \geq \sup \alpha_{r'}$ if $r < r'$.



# Efficient Computation via Dynamic Programming

- A naïve implementation explicitly sum over each vicinal group for each query, still requiring quadratic complexity.
- Thanks to the additive structure in linearized attention, we present a **dynamic programming** algorithm to achieve sub-quadratic complexity.

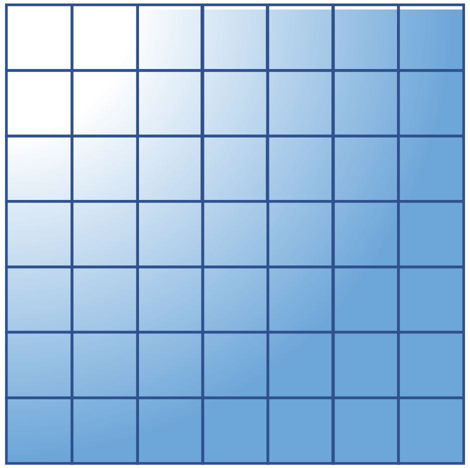
# Efficient Computation via Dynamic Programming

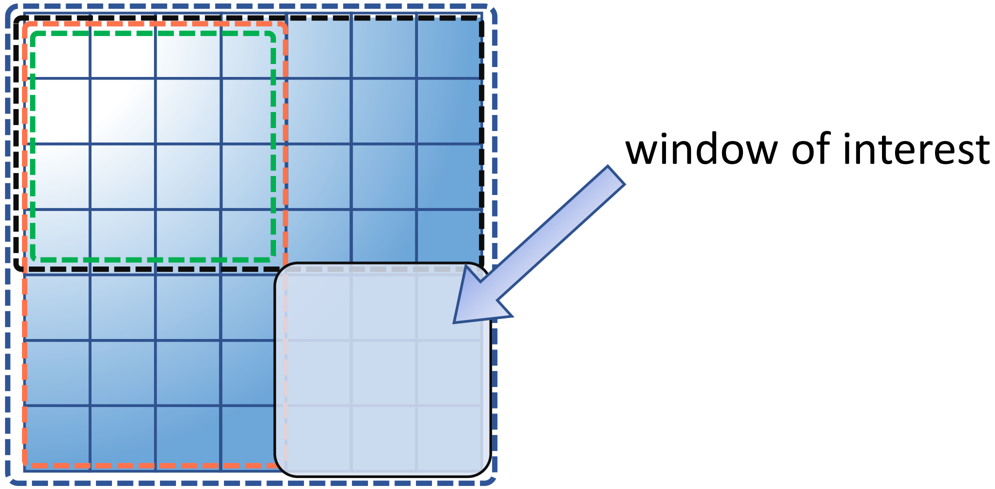
- We maintain a summed-area table (SAT), storing the **prefix sum** of all tokens in image $\mathrm{I}$ above and to the left of each index $(i, j)$,

$$\mathbf{S}(i,j) = \sum_{i'=1}^{i} \sum_{j'=1}^{j} \mathbf{I}(i',j')$$

- The sum over any window can then be retrieved in constant time:

$$\mathcal{W}(i,j,r) \coloneqq \mathbf{S}(i+r, j+r) - \mathbf{S}(i-r-1, j+r) - \mathbf{S}(i+r, j-r-1) + \mathbf{S}(i-r-1, j-r-1)$$





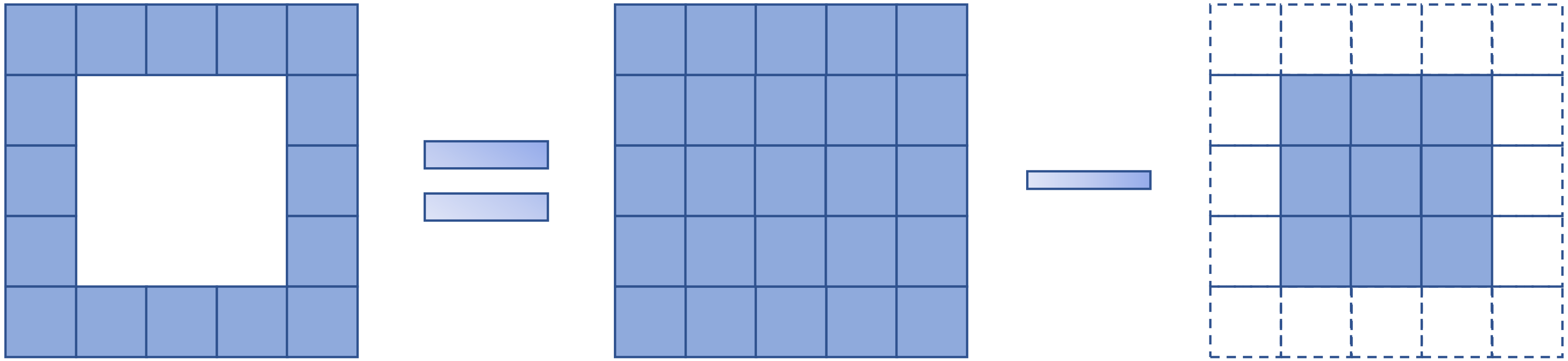
# Efficient Computation via Dynamic Programming

- The sum over vicinal groups in ripple attention can also be computed in constant time:

$$\sum_{(m,n)\in\mathcal{N}_r(i,j)} \phi(\mathbf{k}_{mn}) \otimes \mathbf{v}_{mn} = \mathcal{W}_1(i,j,r) - \mathcal{W}_1(i,j,r-1)$$

$$\sum_{(m,n)\in\mathcal{N}_r(i,j)} \phi(\mathbf{k}_{mn}) = \mathcal{W}_2(i,j,r) - \mathcal{W}_2(i,j,r-1)$$

$$\mathrm{RippleAttn}\left(\mathbf{q}_{ij}, \{\mathbf{k}_{mn}\}, \{\mathbf{v}_{mn}\}\right) = \frac{\phi\left(\mathbf{q}_{ij}\right)^\top \sum_{r=0}^{R} \alpha_r(i,j) \left(\mathcal{W}_1(i,j,r) - \mathcal{W}_1(i,j,r-1)\right)}{\phi\left(\mathbf{q}_{ij}\right)^\top \sum_{r=0}^{R} \alpha_r(i,j) \left(\mathcal{W}_2(i,j,r) - \mathcal{W}_2(i,j,r-1)\right)}$$



# Experiments: Image Classification with ViTs

- Ripple improves Linearized Attention (LA) and outperforms conventional quadratic attention even **without** positional encodings.
- It can be considered as an approach to incorporating relative positional information:
  - outperforms previous baselines that try to incorporate relative positional information into LA.

Image classification results on ImageNet1k dataset.

| Model | # Params | Top-1 Acc. | Top-5 Acc. |
|---|---|---|---|
| Models with quadratic complexity | | | |
| DEIT | 5.72M | 72.20 | 91.10 |
| CONVIT (d'Ascoli et al., 2021) | 5.72M | **73.11** | **91.71** |
| Models with sub-quadratic complexity | | | |
| DEIT-LA | 5.76M | 70.67 | 90.16 |
| DEIT-LA + SINCSPE (Liutkus et al., 2021) | 5.84M | 67.32 | 88.14 |
| DEIT-LA + CONVSPE (Liutkus et al., 2021) | 6.69M | 67.64 | 88.40 |
| DEIT-LA + ROPE (Su et al., 2021) | 5.76M | 71.19 | 90.48 |
| PERMUTEFORMER (Chen, 2021) | 5.76M | 71.42 | 90.51 |
| RIPPLE | 5.78M | 73.02 | 91.56 |

Image classification results on CIFAR-100 dataset.

| Model | w/ APE | | | w/o APE | | |
|---|---|---|---|---|---|---|
| | # Params | Top-1 Acc. | Top-5 Acc. | # Params | Top-1 Acc. | Top-5 Acc. |
| DEIT-LA | 5.42M | 67.00 | 88.57 | 5.36M | 54.04 | 79.66 |
| DEIT | 5.42M | 67.87 | 89.71 | 5.36M | 53.64 | 80.30 |
| CONVIT | 5.42M | **74.34** | **92.87** | 5.36M | **73.88** | **92.20** |
| RIPPLE | 5.47M | 73.94 | 92.37 | 5.42M | 72.94 | 91.86 |

# Experiments: Object Detection

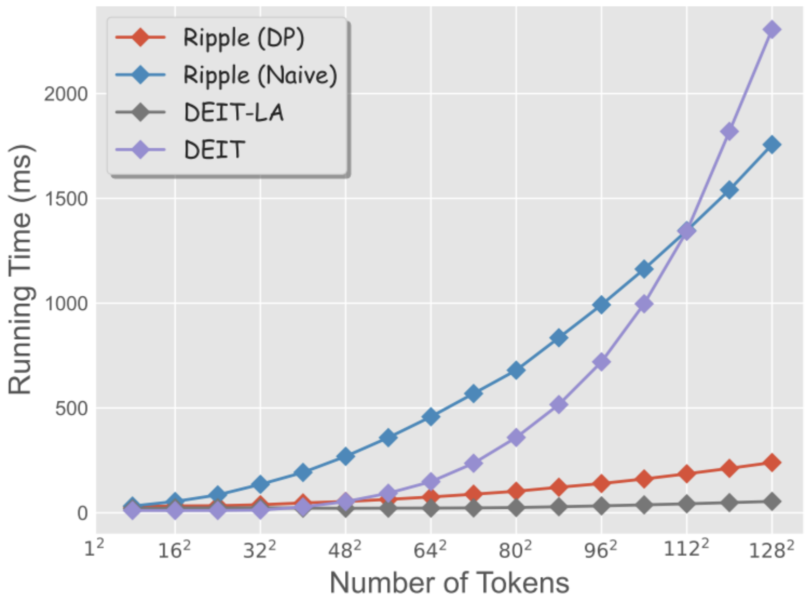
- Ripple gives clear performance boost over LA on object detection.
- It also achieves better results than baselines on detecting **small** scale objects, which might be due to the promoted local correlations.

Object detection on COCO benchmark.

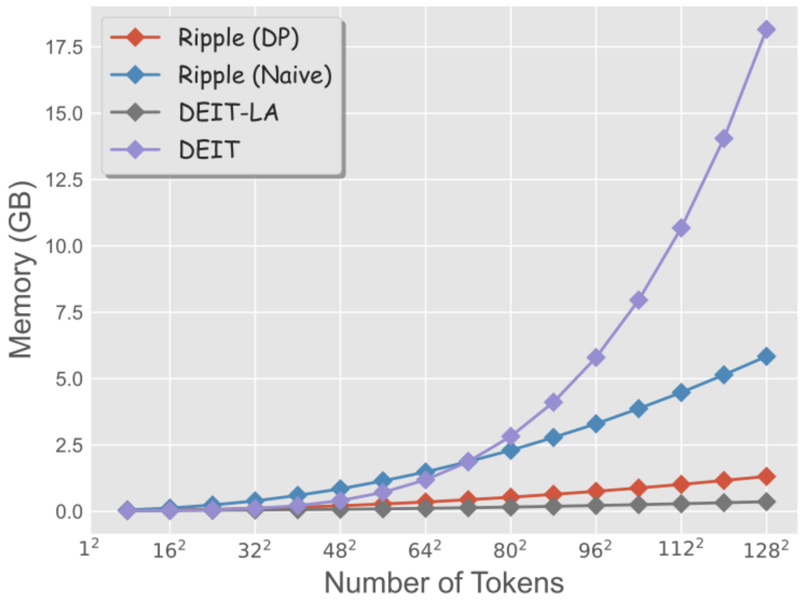
| Model | # Params | GFLOPs | Inference time(s) | 50 epochs | | | | 108 epochs | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | AP | $AP_S$ | $AP_M$ | $AP_L$ | AP | $AP_S$ | $AP_M$ | $AP_L$ |
| SMCA | 41.5M | 88 | 0.059 | **41.0** | 21.9 | **44.3** | **59.1** | **42.7** | 22.8 | **46.1** | **60.0** |
| SMCA-LA | 41.7M | 79 | 0.062 | 39.1 | 19.8 | 42.8 | 56.5 | 41.1 | 22.0 | 44.5 | 59.0 |
| SMCA-RIPPLE | 41.8M | 80 | 0.065 | 40.5 | **22.1** | 44.1 | 57.7 | 42.3 | **23.2** | 45.6 | **60.0** |

# Experiments: Efficiency

- Ripple attention (w/ dynamic programming) scales better to higher-resolution images with lower computational costs.




Running time




Memory consumption

# Thanks!