# Scaling Local Self-Attention for Parameter Efficient Visual Backbones

CVPR '21 (*oral*)

(Uploaded to Arxiv on 23 Mar, 2021 )

Presenter: Sungwon Hwang

*May 3, 2021*


Ashish Vaswani
Google Research

Prajit Ramachandran
Google Research

Aravind Srinivas
UC Berkeley

Niki Parmar
Google Research

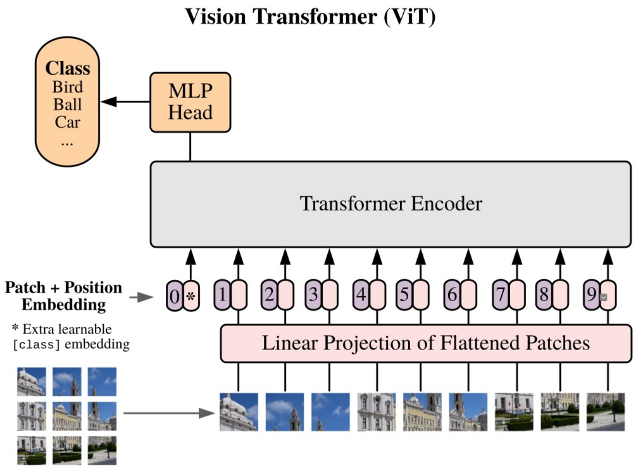
Blake Hechtman
Google Research

Jonathon Shlens
Google Research

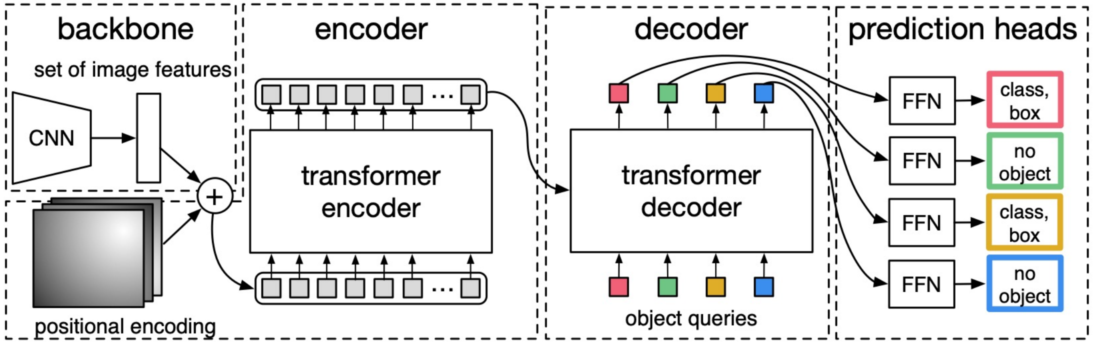
# Introduction

Advents of different non global self-attention for Vision

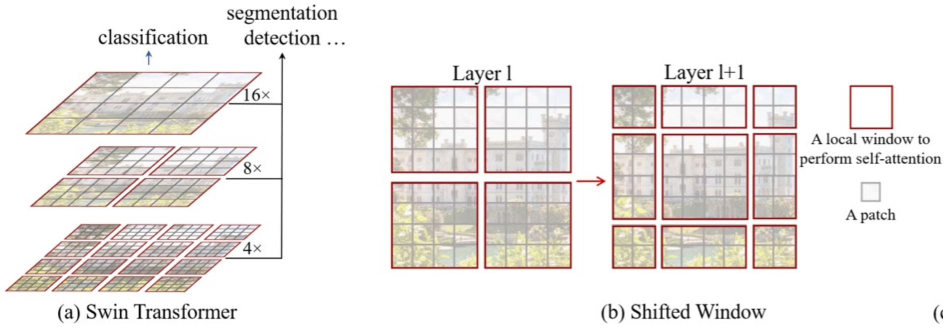
- **Global Attention**
  - ViT (Vision Transformer)




  - DETR (Detection Transformer)




- **(Spatially) Local Attention**
  - Swin (Sliding window) Transformer




- **(Selective) Local Attention**
  - Deformable DETR

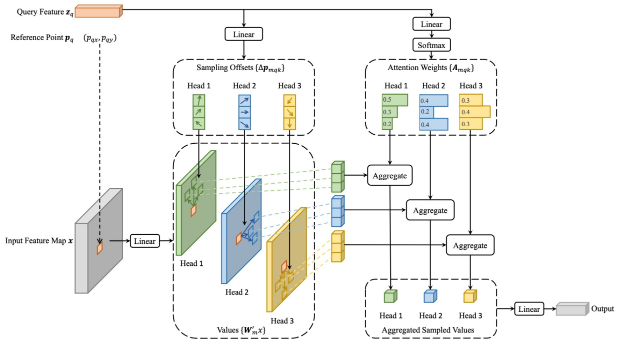



Figure 2: Illustration of the proposed deformable attention module.

# Introduction

Advantages of self-attention over Convolution and why local self-attention for vision?

- ## Advantages of self-attention

1. **Content-based interactions** as opposed to content-independent interactions of convolutions

2. Parameter-independent scaling of receptive field size as opposed to parameter-dependent scaling of convolution

3. Empirical ability to **capture long-range dependencies** for use in larger images

4. Flexibility to handle and integrate multiple types of data
    - Pixels
    - Point Clouds
    - Any type of sequential information
    - Graphs

- ## Why local?

1. **Nature of linguistic vs visual data?**
    - Language: unpredictable dependencies between words

      -> May be beneficial to attend global relationships

    - Vision: Spatial locality of physical beings usually exists

      -> Is it THAT MUCH critical to understand global context in order to featurize local objects?

      (ex. A picture of anyone's face can be taken anywhere)

2. **Global attention requires high computational complexity.**
    - DETR Encoder: $O(H^2W^2C)$

      -> Quadratic growth of computation burden for a given increment of spatial size.

# Methodology

Local self-attention as spatially varying convolutional filters

***Design purpose: What if we can design self-attention as spatially-varying convolutional filters?***

$$y_{ij} = \sum_{a,b \in \mathcal{N}(i,j)} f(i,j,a,b) x_{ab},$$

- **Convolutions**

$$f(i,j,a,b)^{conv} = W_{a-i,b-j}$$

- **Local Self-Attention**

*Content – content interaction*

$$f(i,j,a,b)^{self-att} = \mathtt{softmax}_{ab}\Big(\boxed{(W_Q x_{ij})^\top W_K x_{ab}} + \boxed{(W_Q x_{ij})^\top r_{a-i,b-j}}\Big) W_V$$

*Content – geometry interaction*

$$= p^{ij}_{a-i,b-j} W_v$$

# Methodology

## Local self-attention as spatially varying convolutional filters

$$y_{ij} = \sum_{a,b \in \mathcal{N}(i,j)} f(i,j,a,b)\, x_{ab},$$

- **Relative Spatial Attention**

$$f(i,j,a,b)^{self-att} = \texttt{softmax}_{ab}\Big( (W_Q x_{ij})^\top W_K x_{ab} + (W_Q x_{ij})^\top \boxed{r_{a-i,b-j}} \Big) W_V$$

$$= p^{ij}_{a-i,b-j} W_v$$

*Content – geometry interaction*

| | | | |
|---|---|---|---|
| -1, -1 | -1, 0 | -1, 1 | -1, 2 |
| 0, -1 | 0, 0 | 0, 1 | 0, 2 |
| 1, -1 | 1, 0 | 1, 1 | 1, 2 |
| 2, -1 | 2, 0 | 2, 1 | 2, 2 |

$$(a-i, b-j) \xrightarrow{\text{embed}} r_{a-i,b-j} = \left(r_{a-i}, r_{b-j}\right),$$

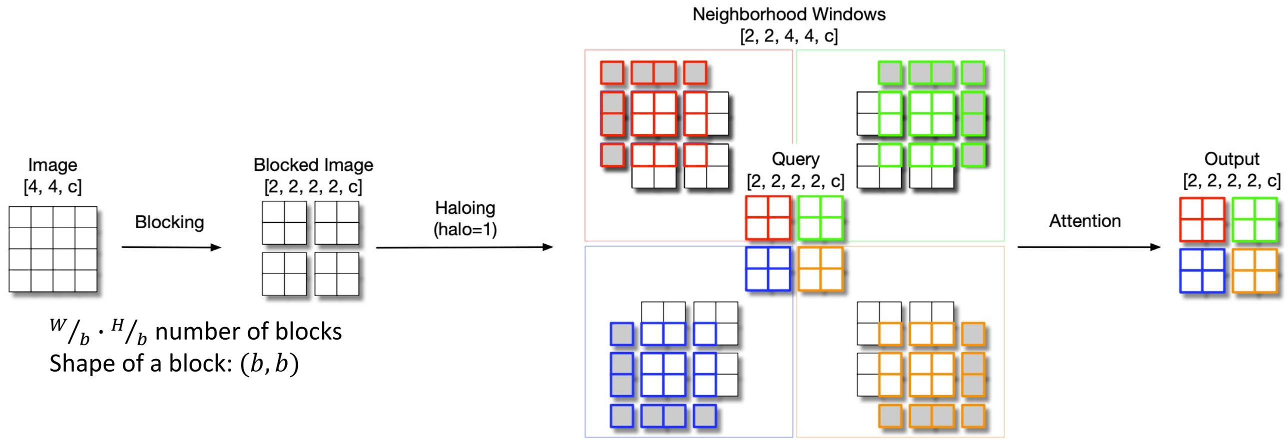
$$r_{a-i}, r_{b-j} \in \mathbb{R}^{d_{out}/2}$$

→ Relative Spatial Attention instead of positional encoding to get CLOSER to **TRANSLATIONAL EQUIVARIANCE**

*(ViT does this positional embedding PATCHWHISE => less translational equivariant => requires large-scale pretraining to have more access to larger samples of translations???)*

# Methodology

## Blocked Local Self-Attention




$^{W}/_{b} \cdot {}^{H}/_{b}$ number of blocks
Shape of a block: $(b, b)$

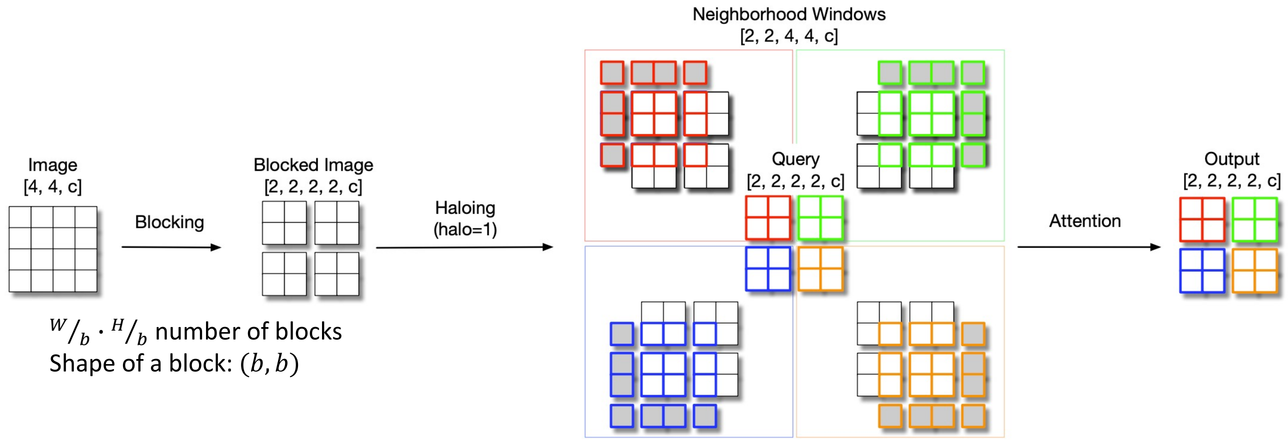
| Method | Neighborhood Memory | Receptive Field | FLOPs Per Pixel | |
|---|---|---|---|---|
| Global | $HWc$ | $HW \times HW$ | $4(HW)^2c$ | $\mathbf{O}((\boldsymbol{HW})^2)$ |
| Per pixel windows | $HWk^2c$ | $k \times k$ | $4k^2c$ | $\mathbf{O}(\boldsymbol{k}^2)$ |
| SASA [43] | $\frac{HW}{b^2}(b+2h)^2c$ | $k \times k$, where $h = \lfloor \frac{k}{2} \rfloor$ | $4(b+2h)^2c$ | $\mathbf{O}((\boldsymbol{b}+\mathbf{2}\boldsymbol{h})^2)$ |
| Blocked local (ours) | $\frac{HW}{b^2}(b+2h)^2c$ | $(b+2h) \times (b+2h)$ | $4(b+2h)^2c$ | $\mathbf{O}((\boldsymbol{b}+\mathbf{2}\boldsymbol{h})^2)$ |

- Two straightforward methods are at the end of the spectrum
  - Global (No local window, or $k = HW$): **Quadratic computation** w.r.t spatial size
  - Per-pixel windows (The most similar to conventional convolutions): **Quadratic increment of neighborhood memory size** w.r.t window size

- <u>**Intuition for solution**</u>: Adjacent pixels share most neighbors. ($k \times (k-1)$ shared neighbors for $k \times k$ window)

# Methodology

## Blocked Local Self-Attention




| Method | Neighborhood Memory | Receptive Field | FLOPs Per Pixel | |
|---|---|---|---|---|
| Global | $HWc$ | $HW \times HW$ | $4(HW)^2c$ | $O((HW)^2)$ |
| Per pixel windows | $HWk^2c$ | $k \times k$ | $4k^2c$ | $O(k^2)$ |
| SASA [43] | $\frac{HW}{b^2}(b+2h)^2c$ | $k \times k$, where $h = \lfloor \frac{k}{2} \rfloor$ | $4(b+2h)^2c$ | $O((b+2h)^2)$ |
| Blocked local (ours) | $\frac{HW}{b^2}(b+2h)^2c$ | $(b+2h) \times (b+2h)$ | $4(b+2h)^2c$ | $O((b+2h)^2)$ |

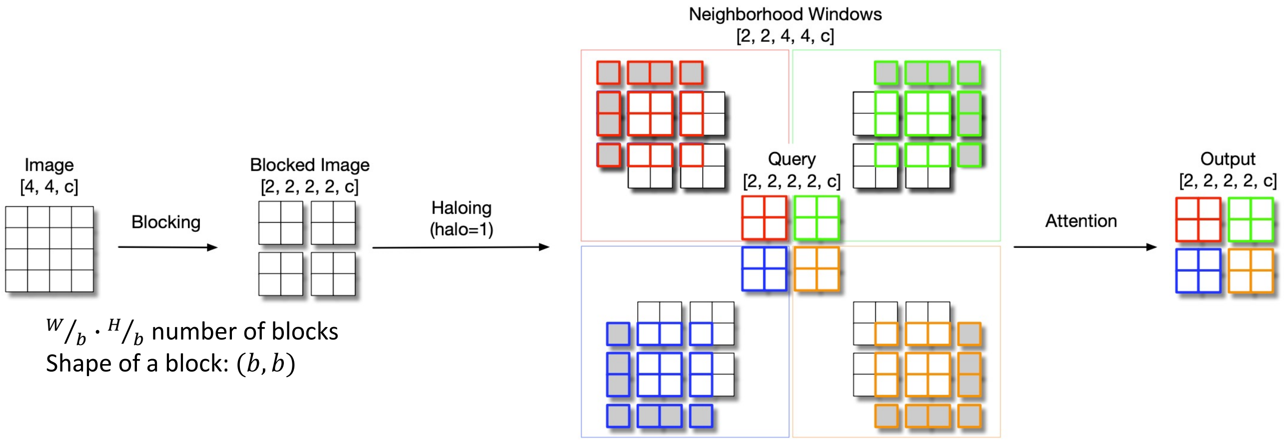
- **<u>Solution</u>**: Blocks can safely share the same neighbors: **Pixels in blocks attend to shared, haloed neighbors**
  - Query: Pixels in **blocks** -> $({}^{W}/_{b} \cdot {}^{H}/_{b}, c)$
  - Key, value: Pixels in **halos** -> $((b+2h)^2, c)$

-> Manageable neighborhood memory while controlling FLOPs per pixel

-> Do NOT need to have larger number of parameters for larger receptive field compared to CNN.

# Methodology

## Blocked Local Self-Attention



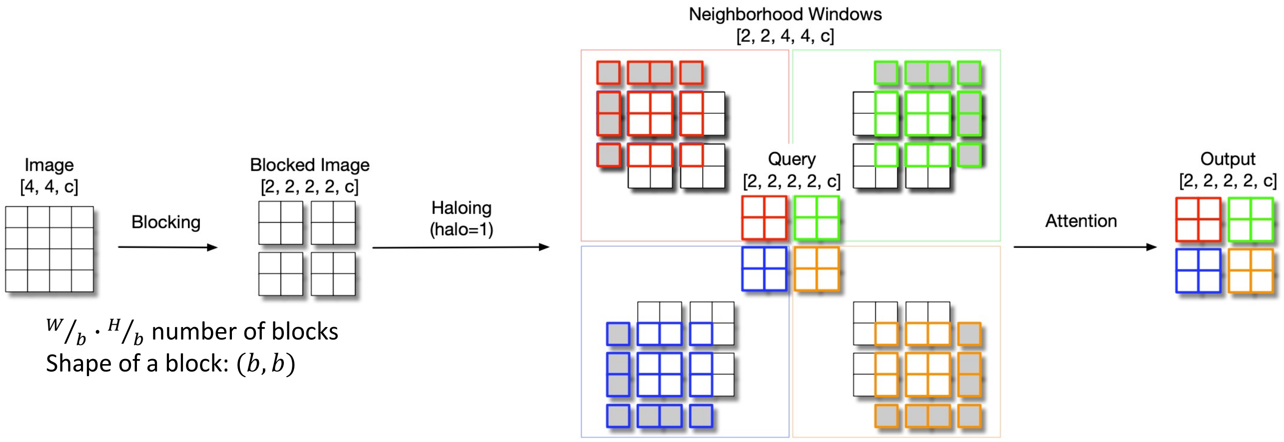

$^{W}/_{b} \cdot {}^{H}/_{b}$ number of blocks
Shape of a block: $(b, b)$

| Method | Neighborhood Memory | Receptive Field | FLOPs Per Pixel | |
|---|---|---|---|---|
| Global | $HWc$ | $HW \times HW$ | $4(HW)^2c$ | $\mathbf{O}((\boldsymbol{HW})^2)$ |
| Per pixel windows | $HWk^2c$ | $k \times k$ | $4k^2c$ | $\mathbf{O}(\boldsymbol{k}^2)$ |
| SASA [43] | $\frac{HW}{b^2}(b+2h)^2c$ | $k \times k$, where $h = \lfloor \frac{k}{2} \rfloor$ | $4(b+2h)^2c$ | $\mathbf{O}((\boldsymbol{b}+\mathbf{2}\boldsymbol{h})^2)$ |
| Blocked local (ours) | $\frac{HW}{b^2}(b+2h)^2c$ | $(b+2h) \times (b+2h)$ | $4(b+2h)^2c$ | $\mathbf{O}((\boldsymbol{b}+\mathbf{2}\boldsymbol{h})^2)$ |

- Comparisons of Neighborhood Memory & FLOPS: per-pixel windows vs. Blocked local
  - Set equal receptive field for fair comparison: $k = b + 2h$
  - Neighborhood Memory: $\frac{Per-pixel}{Blocked\ local} = b^2$
  - FLOPs per pixel: $\frac{Per-pixel}{Blocked\ local} = 1$
  - => **Manageable neighborhood memory while keeping FLOPs per pixel**

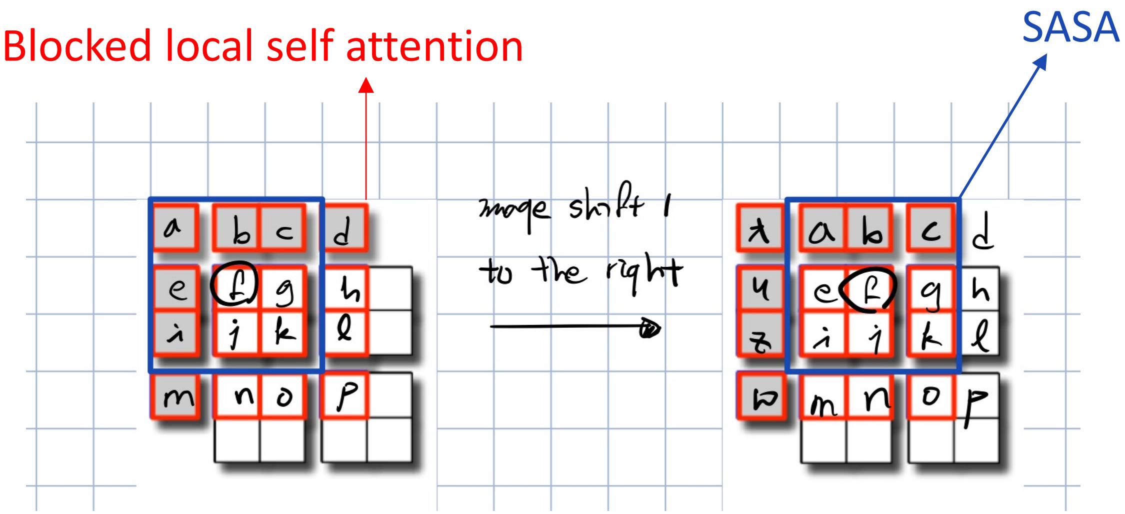
# Methodology

## Blocked Local Self-Attention vs. SASA: Larger receptive fields are more important than translational equivariance



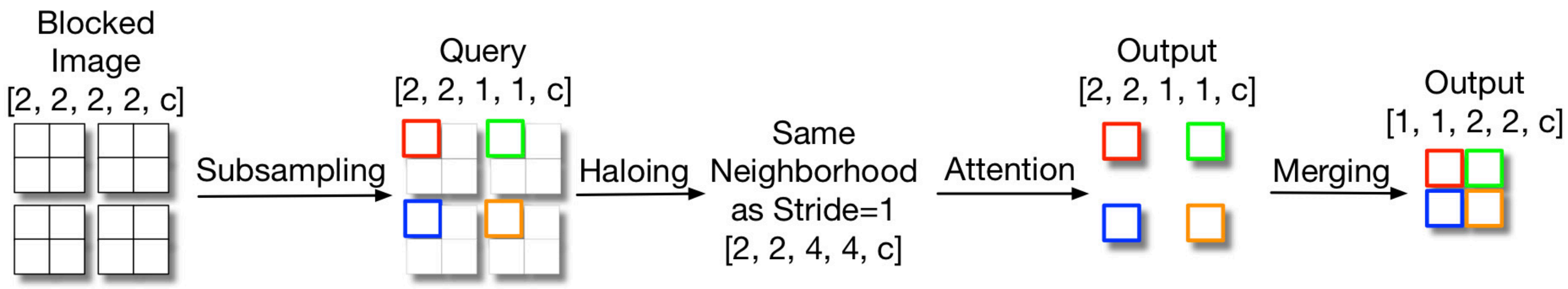

| Method | Neighborhood Memory | Receptive Field | FLOPs Per Pixel | |
|---|---|---|---|---|
| Global | $HWc$ | $HW \times HW$ | $4(HW)^2c$ | $O((HW)^2)$ |
| Per pixel windows | $HWk^2c$ | $k \times k$ | $4k^2c$ | $O(k^2)$ |
| SASA [43] | $\frac{HW}{b^2}(b+2h)^2c$ | $k \times k$, where $h = \lfloor \frac{k}{2} \rfloor$ | $4(b+2h)^2c$ | $O((b+2h)^2)$ |
| Blocked local (ours) | $\frac{HW}{b^2}(b+2h)^2c$ | $(b+2h) \times (b+2h)$ | $4(b+2h)^2c$ | $O((b+2h)^2)$ |

- SASA: gives up receptive field for translational equivariance



# Methodology

**Down-sampling with local self-attention**




- Just like strided convolution with stride of $b$

# Model Configuration

## HaloNet: ResNet-like structure & empirical comparisons with Efficient Net

### HaloNet

| Output Resolution | Layers |
|---|---|
| $\frac{s}{4} \times \frac{s}{4}$ | $7 \times 7$ conv stride 2, 64<br>$3 \times 3$ max pool stride 2 |
| $\frac{s}{4} \times \frac{s}{4}$ | $\left\{\begin{array}{c} 1 \times 1, 64 \\ \text{attention}(b,h), \boxed{64 \cdot r_v} \\ 1 \times 1, 64 \cdot r_b \end{array}\right\} \times 3$ |
| $\frac{s}{8} \times \frac{s}{8}$ | $\left\{\begin{array}{c} 1 \times 1, 128 \\ \text{attention}(b,h), 128 \cdot r_v \\ 1 \times 1, 128 \cdot r_b \end{array}\right\} \times 3$ |
| $\frac{s}{16} \times \frac{s}{16}$ | $\left\{\begin{array}{c} 1 \times 1, 256 \\ \text{attention}(b,h), 256 \cdot r_v \\ 1 \times 1, 256 \cdot r_b \end{array}\right\} \times l_3$ |
| $\frac{s}{32} \times \frac{s}{32}$ | $\left\{\begin{array}{c} 1 \times 1, 512 \\ \text{attention}(b,h), 512 \cdot r_v \\ 1 \times 1, 512 \cdot r_b \end{array}\right\} \times 3$ |
| $\frac{s}{32} \times \frac{s}{32}$ | $1 \times 1, d_f$ |
| $1 \times 1$ | global average pooling<br>fc, 1000 |

Table 2. **HaloNet model family specification.**

Value 의 linear projection 시 dimension 을 늘려주면 됨

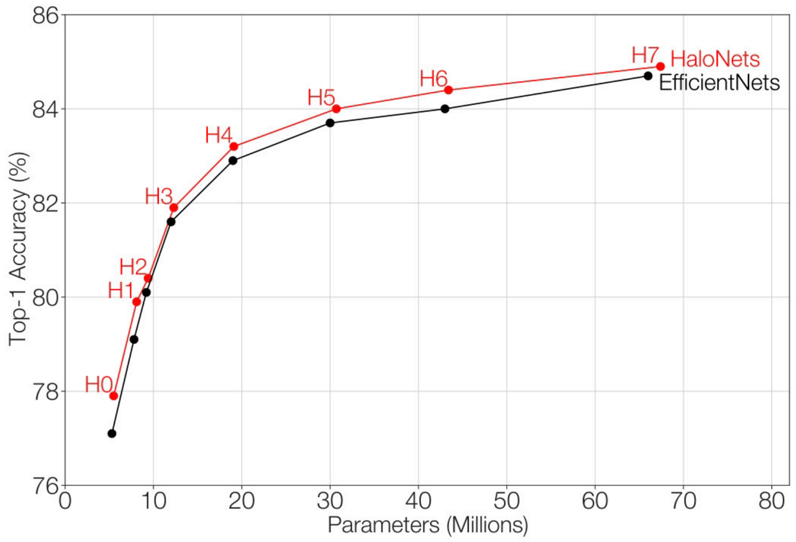
### ResNet

| layer name | output size | 18-layer | 34-layer | 50-layer | 101-layer | 152-layer |
|---|---|---|---|---|---|---|
| conv1 | 112×112 | | | 7×7, 64, stride 2 | | |
| | | | | 3×3 max pool, stride 2 | | |
| conv2_x | 56×56 | [3×3, 64<br>3×3, 64] ×2 | [3×3, 64<br>3×3, 64] ×3 | [1×1, 64<br>3×3, 64<br>1×1, 256] ×3 | [1×1, 64<br>3×3, 64<br>1×1, 256] ×3 | [1×1, 64<br>3×3, 64<br>1×1, 256] ×3 |
| conv3_x | 28×28 | [3×3, 128<br>3×3, 128] ×2 | [3×3, 128<br>3×3, 128] ×4 | [1×1, 128<br>3×3, 128<br>1×1, 512] ×4 | [1×1, 128<br>3×3, 128<br>1×1, 512] ×4 | [1×1, 128<br>3×3, 128<br>1×1, 512] ×8 |
| conv4_x | 14×14 | [3×3, 256<br>3×3, 256] ×2 | [3×3, 256<br>3×3, 256] ×6 | [1×1, 256<br>3×3, 256<br>1×1, 1024] ×6 | [1×1, 256<br>3×3, 256<br>1×1, 1024] ×23 | [1×1, 256<br>3×3, 256<br>1×1, 1024] ×36 |
| conv5_x | 7×7 | [3×3, 512<br>3×3, 512] ×2 | [3×3, 512<br>3×3, 512] ×3 | [1×1, 512<br>3×3, 512<br>1×1, 2048] ×3 | [1×1, 512<br>3×3, 512<br>1×1, 2048] ×3 | [1×1, 512<br>3×3, 512<br>1×1, 2048] ×3 |
| | 1×1 | | | average pool, 1000-d fc, softmax | | |
| FLOPs | | $1.8\times10^9$ | $3.6\times10^9$ | $3.8\times10^9$ | $7.6\times10^9$ | $11.3\times10^9$ |

# Experiments

## Comparison to Conv Exp #1) ImageNet classification accuracy vs. EfficientNet

- HaloNet can be **trained over ImageNet from scratch** & yield state-of-the-art performance
  - ViT / BiT: Requires pre-training on larger datasets
    - Imagenet-21k
    - JFT-300M




| HaloNet Model | $b$ | $h$ | $r_v$ | $r_b$ | Total Layers | $l_3$ | $s$ | $d_f$ | Params (M) | EfficientNet Params (M) | EfficientNet Image Size (M) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H0 | 8 | 3 | 1.0 | 0.5 | 50 | 7 | 256 | – | 5.5 | B0: 5.3 | 224 |
| H1 | 8 | 3 | 1.0 | 1.0 | 59 | 10 | 256 | – | 8.1 | B1: 7.8 | 240 |
| H2 | 8 | 3 | 1.0 | 1.25 | 62 | 11 | 256 | – | 9.4 | B2: 9.2 | 260 |
| H3 | 10 | 3 | 1.0 | 1.5 | 65 | 12 | 320 | 1024 | 12.3 | B3: 12 | 300 |
| H4 | 12 | 2 | 1.0 | 3 | 65 | 12 | 384 | 1280 | 19.1 | B4: 19 | 380 |
| H5 | 14 | 2 | 2.5 | 2 | 98 | 23 | 448 | 1536 | 30.7 | B5: 30 | 456 |
| H6 | 8 | 4 | 3 | 2.75 | 101 | 24 | 512 | 1536 | 43.4 | B6: 43 | 528 |
| H7 | 10 | 3 | 4 | 3.5 | 107 | 26 | 600 | 2048 | 67 | B7: 66 | 600 |

# Experiments

## Comparison to Conv Exp #2) Transfer of convolutional components to self-attention

- Architecture variation: Squeeze and Excitation (SE) / channel attention after spatial convolution (SiLU/Swish-1)
- Regularization: Random Augment (RA) / Label Smoothing (LS)
- HaloNet leverages from regularization more than ResNet-50.
  - **Label Smoothing(LS) + RandomAugment (RA)** yields 1.3% increase for Halo / 0.8 for ResNet-50
  - Usually larger convolutional models benefit from regularization -> HIGHER EXPRESSIBILITY OF HALONET?
- Components that doesn't affect HaloNet
  - **Squeeze and Excitation (SE)**
    - Self-attention module already has higher DOF of expressibility than channel-wise attention block (SE).

| Components | HaloNet Accuracy | Baseline $\Delta$ | ResNet Accuracy | Baseline $\Delta$ |
|---|---|---|---|---|
| Baseline | 78.6 | 0.0 | 77.6 | 0.0 |
| + LS | 79.7 | 1.1 | 78.1 | 0.5 |
| + LS, RA | 79.9 | 1.3 | 78.4 | 0.8 |
| + SE | 78.6 | 0.0 | 78.6 | 1.0 |
| + SE, SiLU/Sw1 | 79.0 | 0.4 | 78.9 | 1.3 |
| + LS, SE | 79.7 | 1.1 | 78.9 | 1.3 |
| + LS, SE, SiLU/Sw1 | 79.9 | 1.3 | 79.1 | 1.5 |
| + LS, SE, SiLU/Sw1, RA | 80.5 | 1.9 | 79.5 | 1.9 |

# Experiments

## Comparison to Conv Exp #3) Comparative study on Object Detection / Instance Segmentation (COCO)

- Last 3 layers of ResNet baselines are added with local attention block to compare with ResNet
- Results to note:
  - Still suffering from localizing small objects
  - Feels like translational equivariance is critical for fine-grained localization. Maybe try SASA (local attention mask)?
  - Larger box size perform better for localizing smaller objects?

| Model | $\mathrm{AP^{bb}}$ | $\mathrm{AP^{bb}_s}$ | $\mathrm{AP^{bb}_m}$ | $\mathrm{AP^{bb}_l}$ | $\mathrm{AP^{mk}}$ | $\mathrm{AP^{mk}_s}$ | $\mathrm{AP^{mk}_m}$ | $\mathrm{AP^{mk}_l}$ | Speed (ms) | Train time (hrs) |
|---|---|---|---|---|---|---|---|---|---|---|
| R50 baseline in lit | 42.1 | 22.5 | 44.8 | 59.1 | 37.7 | 18.3 | 40.5 | 54.9 | 409 | 14.6 |
| R50 + SE (our baseline) | 44.5 (+2.4) | 25.5 | 47.7 | 61.2 | 39.6 (+1.9) | 20.4 | 42.6 | 57.6 | 446 | 15.2 |
| R50 + SE + Local Att ($b = 8$) | 45.2 (++0.7) | 25.4 | 48.1 | 63.3 | 40.3 (++0.7) | 20.5 | 43.1 | 59.0 | 540 | 15.8 |
| R50 + SE + Local Att ($b = 32$) | 45.4 (++0.9) | 25.9 | 48.2 | 63.0 | 40.5 (++0.9) | 21.2 | 43.5 | 58.8 | 613 | 16.5 |
| R101 + SE (our baseline) | 45.9 (+3.8) | 25.8 | 49.5 | 62.9 | 40.6 (+2.9) | 20.9 | 43.7 | 58.7 | 740 | 17.9 |
| R101 + SE + Local Att ($b = 8$) | 46.8 (++0.9) | 26.3 | 50.0 | 64.5 | 41.2 (++0.6) | 21.4 | 44.3 | 59.8 | 799 | 18.4 |

Table 5. **Accuracies on object detection and instance segmentation.** We experiment with two settings for self-attention in the last stage: A block size of ($b$) of 8 and a halo size ($h$) of 3 and also with ($b = 32, h = 3$) for ResNet-50. *bb* (bounding box) refers to detection, and $mk$ (mask) refers to segmentation. The identifiers $s$, $m$, and $l$ refer to small, medium, and large objects respectively. Speed is measured as the milliseconds taken by only the backbone (and not the FPN) for a batch size of 32 on 2 TPUv3 cores. The train time the total training time calculated from the peak images/sec of the Mask-RCNN training run on 8 TPUv3 cores with a batch size of 64.

# Experiments

## Comparison to Transformers for Vision

- Pretrain: Public ImageNet-21k
- Finetune: ImageNet.
- *4×4 patch:* Replacing the convolutional stem with linear projection of *4×4 patch (Just like ViT)*
- *Conv-12: First 2 stages are made with convolutions*
- **Good parameter trade-off / inference speed trade-off for given accuracy**

| Conv Stages | Attention Stages | Top-1 Acc (%) | Norm. Train Time |
|---|---|---|---|
| - | 1, 2, 3, 4 | 84.9 | 1.9 |
| 1 | 2, 3, 4 | 84.6 | 1.4 |
| 1, 2 | 3, 4 | 84.7 | 1.0 |
| 1, 2, 3 | 4 | 83.8 | 0.5 |

Table 4. **Replacing attention layers with convolutions in stages 1 and 2 exhibit the best speed vs. accuracy tradeoff.** All the models had about 67 million parameters and the train and inference times are normalized to the corresponding times for EfficientNet B7. Please see Figure 8 for a comparison of step time with other HaloNet models.

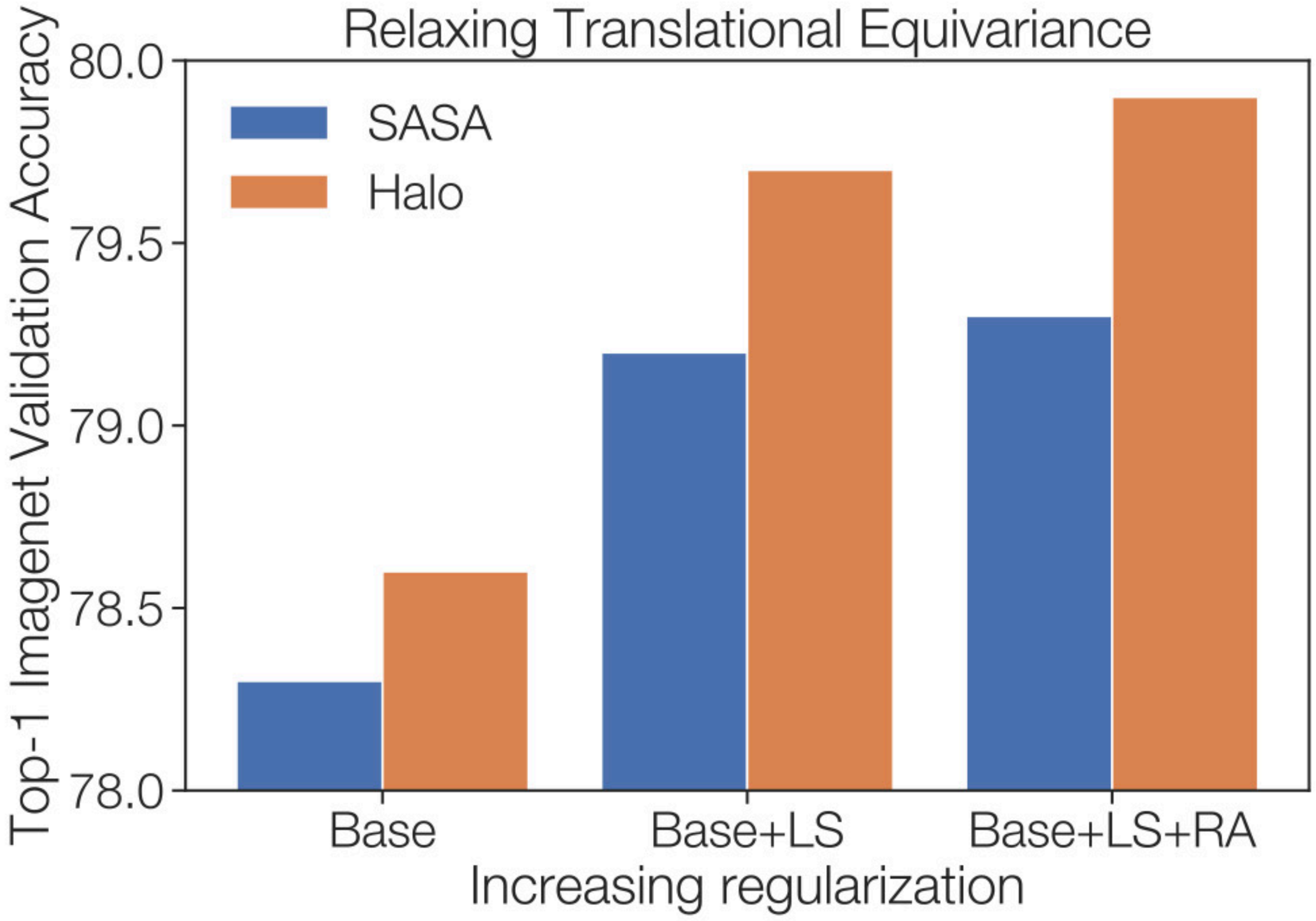
| Model | Parameters (Millions) | Pretraining Image Size (Pixels) | Pretraining Step Time (32 per core) | Finetuning Image Size | Finetuning Top-1 Accuracy (%) | Inference Speed img/sec/core |
|---|---|---|---|---|---|---|
| H4 (base 128) | 85 | 256 | 377 ms | 384/512 | 85.6/85.8 | 121.3/48.6 |
| H4 (base 128, $4 \times 4$ patch) | 85 | 256 | 366 ms | 384/512 | 85.4/85.4 | 125.7/56.5 |
| H4 (base 128, Conv-12) | 87 | 256 | 213 ms | 384/512 | 85.5/85.8 | 257.6/120.2 |
| ViT-L/16 | 300 | 224 | 445 ms | 384/512 | 85.2/85.3 | 74.6/27.4 |
| BiT-M | 928 | 224 | 1021 ms | 384 | 85.4 | 54.2 |

Table 6. **HaloNet models pretrained on ImagetNet-21k perform well when finetuned on ImageNet**. For HaloNet and ViT, we finetuned on $384 \times 384$ and $512 \times 512$ size images. The pretraining step time reports the TPUv3 compute time for a batch size of 32 per core. The inference speed is also computed on a single TPUv3 core.

# Experiments

**HaloNet model study #1) HaloNet vs. SASA: Is it okay to relax translational equivariance for larger receptive field?**

- $b = 8, h = 3, k = 7$



# Experiments

## HaloNet model study #2) Size of receptive field

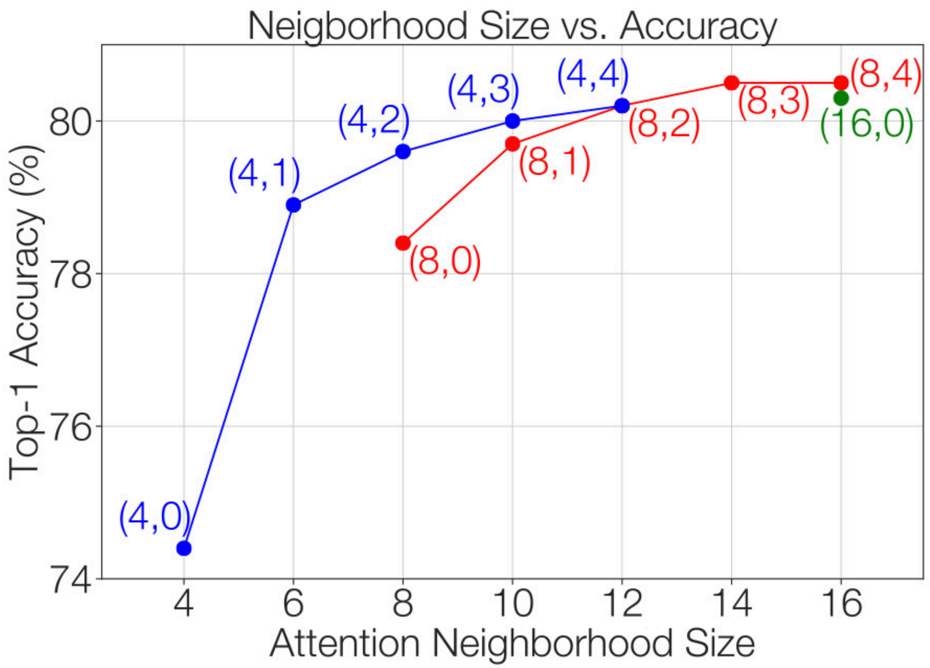



Figure 7. **Increasing window sizes improves accuracy up to a point.** The experiments in the graph have been annotated with their block size ($b$), halo size ($h$), $h = 0$ implies attention with *non-overlapping* blocks