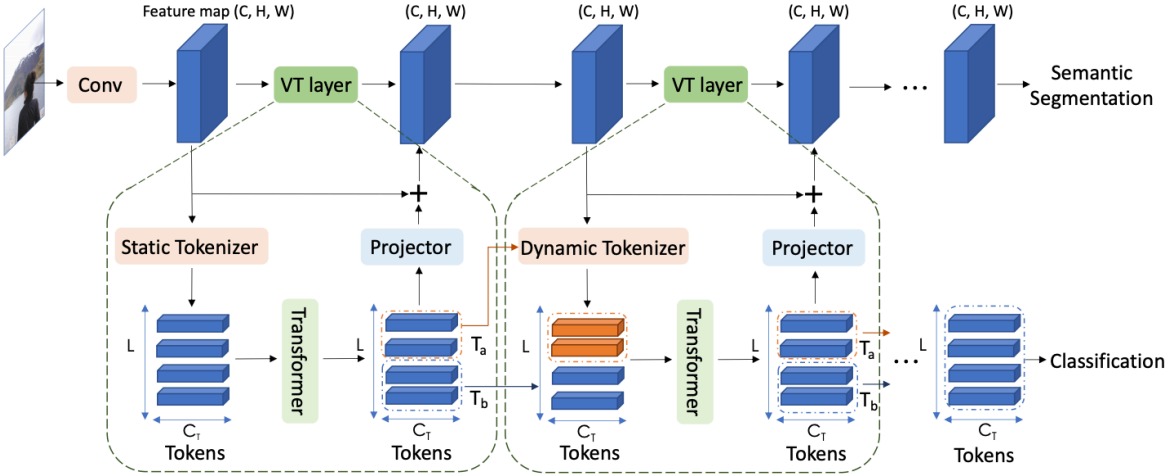
Flexible Learning + Vision Reading Group @TU Berlin - 11/09/2020

Visual Transformers: Token-based Image Representation and Processing for Computer Vision

Bichen Wu[1], **Chenfeng Xu**[2], **Xiaoliang Dai**[1], **Alvin Wan**[1],
Peizhao Zhang[1], **Masayoshi Tomizuka**[2], **Kurt Keutzer**[2], **Peter Vajda**[1]
[1]Facebook, [2]UC Berkeley



A Little History of Natural Language Processing

Bag-of-Words

('the', 8),
(',', 5),
('very', 4),
('.', 4),
('who', 4),
('and', 3),
('good', 2),
('it', 2),
('to', 2),
('a', 2),
('for', 2),
('can', 2),
('this', 2),
('of', 2),
('drama', 1),
('although', 1),
('appeared', 1),
('have', 1),
('few', 1),
('blank', 1)
.....

Limitations

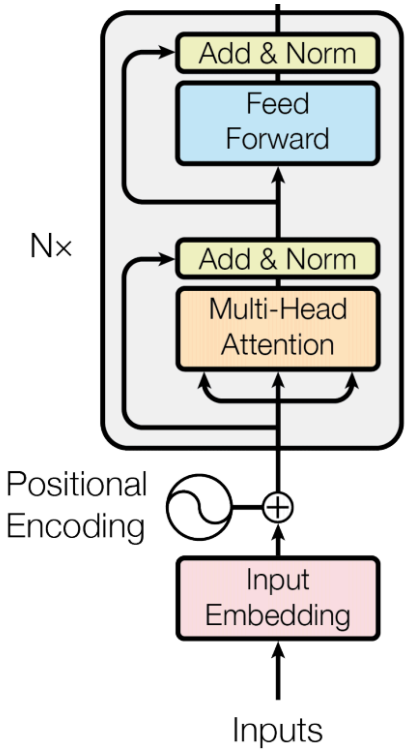
- But order matters! - N-grams?
- Dimensionality explodes

Recurrent Neural Networks

Limitations

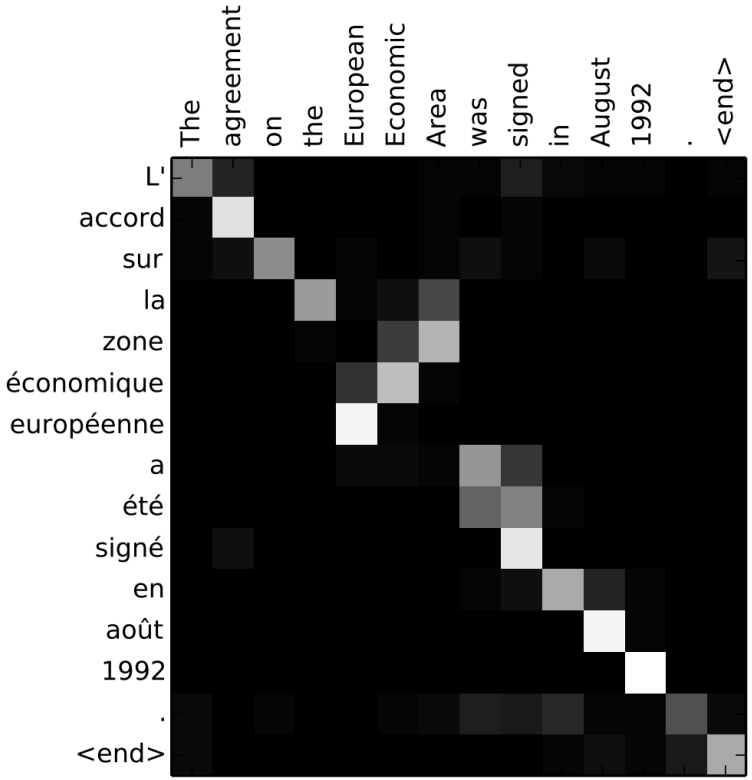
- Hard to scale training
- Long-term dependencies
- Transfer learning is hard

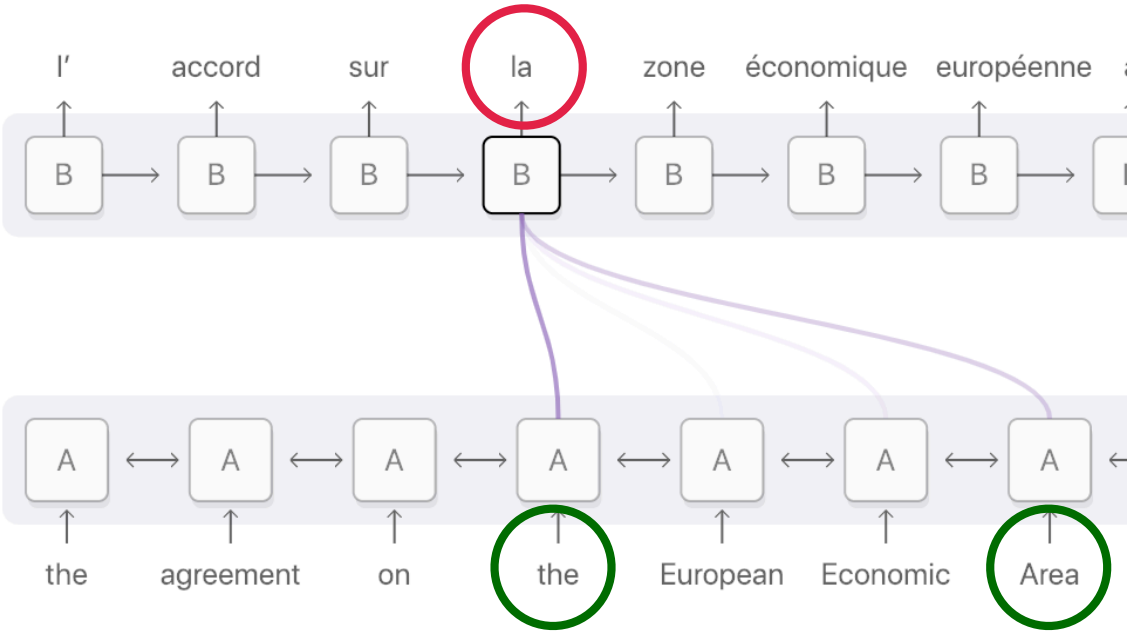
Transformers



Hochreiter & Schmidhuber (1997), Vaswani et al. (2017)

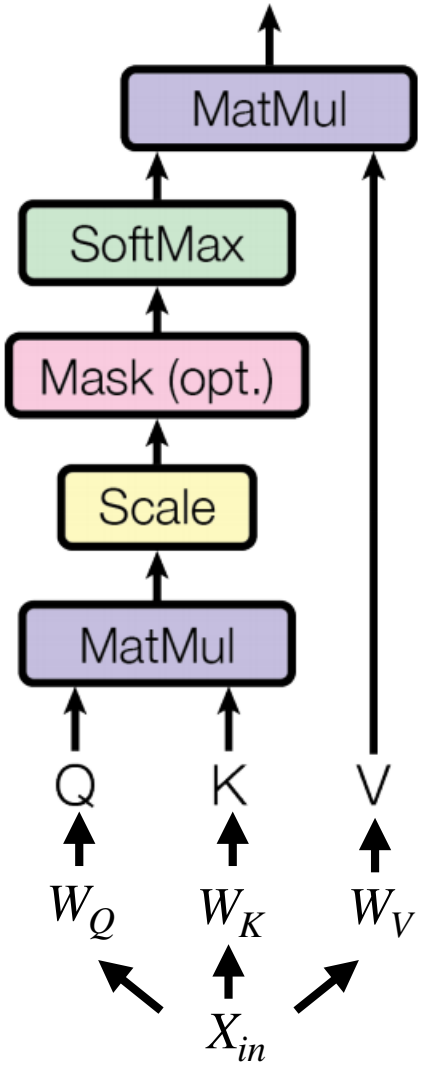
Attention in Language Translation Models





Bahdanu et al. (2015)

Self-Attention in Pseudo-Code




```
def attention(self, X_in:List[Tensor]):
    # For every token transform previous layer's out
    for i in range(self.sequence_length):
        Q[i] = self.W_Q * X_in[i]
        K[i] = self.W_K * X_in[i]

        # compute a weighted sum of the values
        out[i] = 0
        for j in range(self.sequence_length):
            out[i] += relevance[j] * V[j]
```

1. Relative Postional Encoding
2. Multi-Head Attention
3. O(n^2) in Tokens
4. Self-Supervised Training
5. Not sequential processing!
6. ReLU vs. Sigmoid - Precision

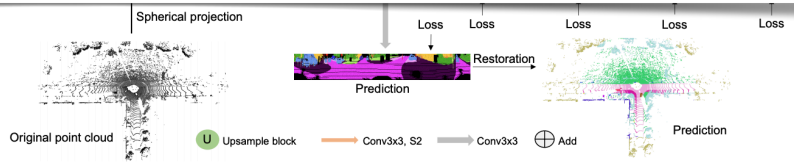
Self-Attention & Transformers in Machine Vision

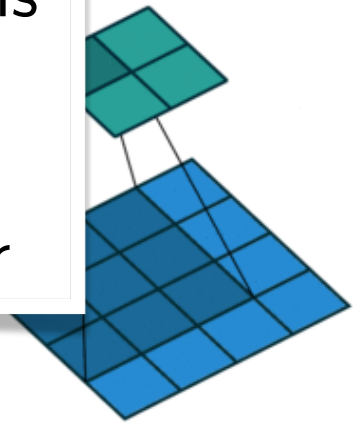
Itti & Koch (1998)

Woo et al. (2018) - Attention on Conv Feature Maps

Paper Motivation

1. Varying Importance of Pixels - Redundant computations
2. Conv needs fixes to handle long-range dependencies
3. Conv is inefficient for sparse high-level concepts
4. Find a solution to bad scaling in #pixels -> Tokenizer

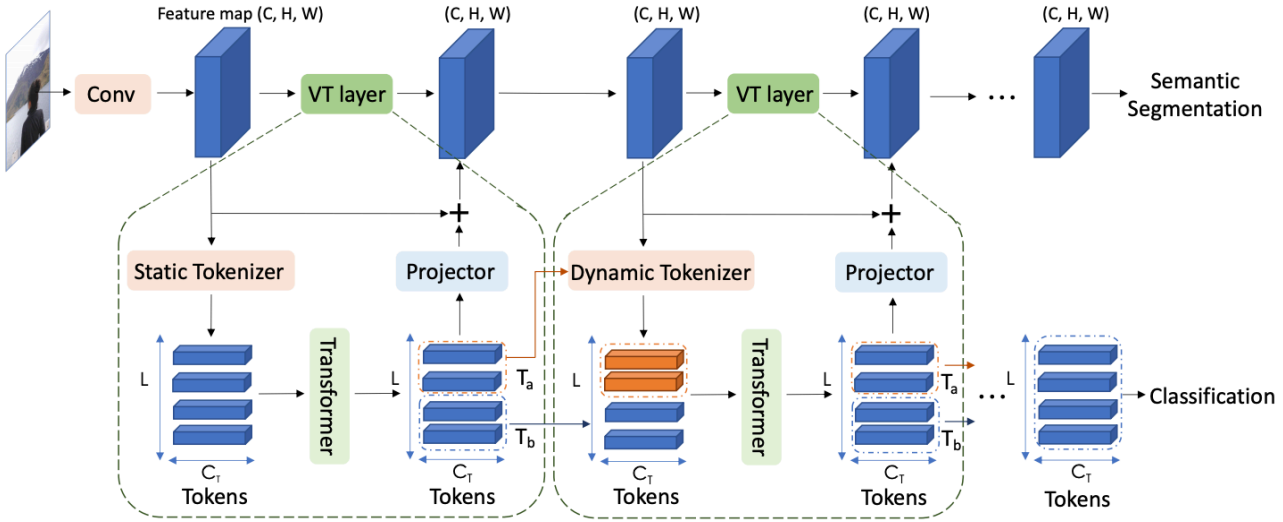




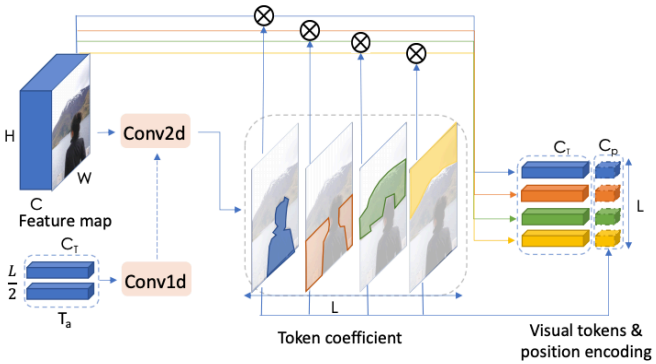
$$Y[m,p,q] = \sigma(\sum_{i,j,n} W(X_0)[m,n,p,q,i,j] \times X[n,p+\hat{i},q+\hat{j}]).$$

$$W[m,n,p,q,i,j] = \hat{W}[m,n,i,j] \times A(X_0)[m,n,p,q,i,j].$$

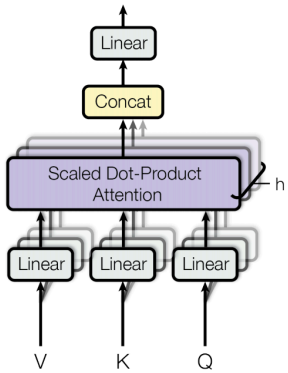
The Visual Transformer: Architecture & Recipe



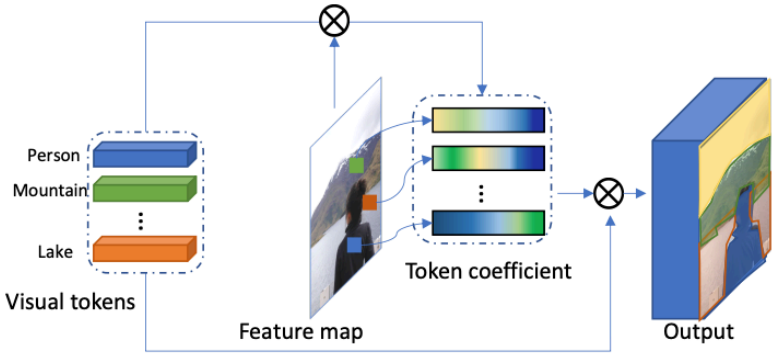
Tokenizer - Dynamic vs. Static



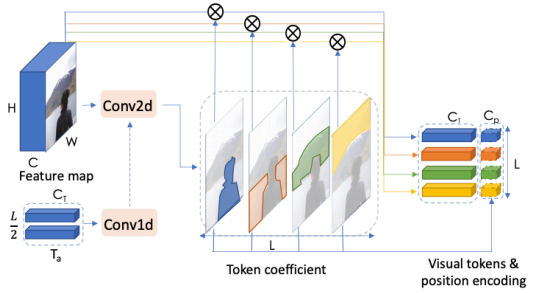
Transformer



Projector - Back to Pixel Space



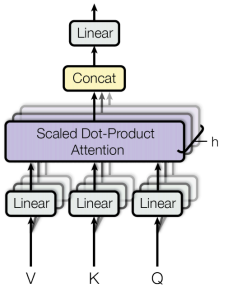
The Visual Transformer: The Details



TOKENS = WEIGHTED AVG. OF PIXELS

$$T = \text{softmax}\left(\bar{X} W_{X\to K}\right)^{T}\left(\bar{X} W_{X\to V}\right)$$

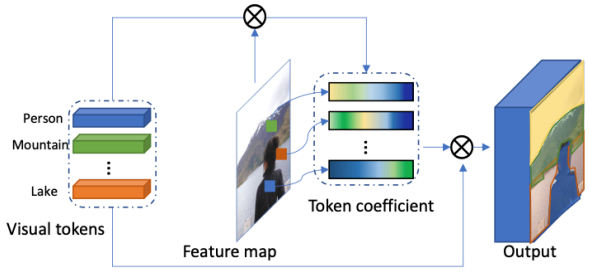
$\bar{X}$: $L \times HW$ (# VIS. TOKENS) — $W_{X\to K}$: $HW \times C_T$ — A: $L \times C_T$ — V: $HW \times C_T$ (# CHANNELS VIS. TOKEN)



$$T_{out} = T_{in} + \left[\text{softmax}\left(\left(T_{in} W_K\right)\left(T_{in} W_Q\right)^{T}\right)\left(T_{in} W_V\right)\right] F$$

"KEY" "QUERY" "VALUE" — F: LEARNED SCALING

T_{out}: $L \times C_T$ — T_{in}: $L \times C_T$ — $\left(T_{in} W_K\right)\left(T_{in} W_Q\right)^{T}$: $L \times L$ — $\left(T_{in} W_V\right)$: $L \times C_T$ — F: $C_T \times C_T$

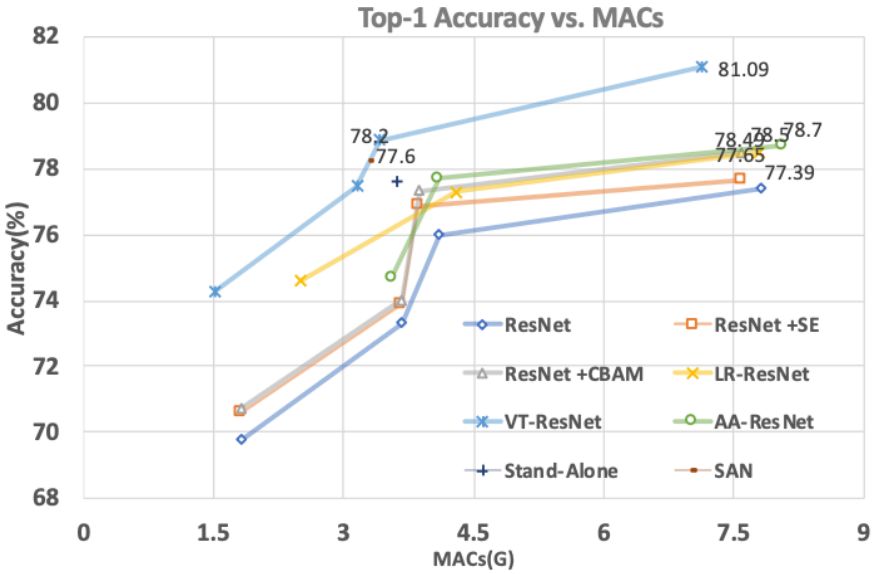


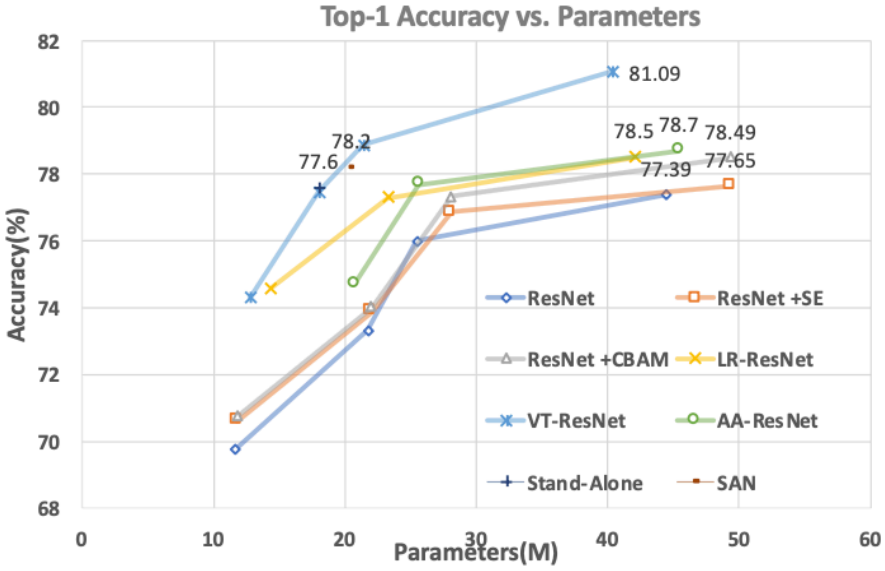
$$\bar{X}_{out} = \bar{X}_{in} + \text{softmax}\left[\left(\bar{X}_{in} W_{K\to X}\right)\left(T_{out} W_{T\to X}\right)^{T}\left(T_{out} W_{V\to X}\right)\right]$$

$\bar{X}_{out}$: $HW \times C$ — $\bar{X}_{in}$: $HW \times C$ — $\bar{X}_{in} W_{K\to X}$: $HW \times C$ — $T_{out} W_{T\to X}$: $C \times L$ — $T_{out} W_{V\to X}$: $L \times C$

Results: ImageNet Classification

Models	Top-1 Acc (%)	Δ Top-1 Acc (%)	MACs (G)	Δ MACs (M)	Params (M)	Δ Params (M)
R-101 [5]	77.39	-	7.802	-	44.44	-
R-101 + SE [7, 9]	77.65	+0.26	7.575*	+5*	49.33	+4.89
R-101 + CBAM [9]	78.49	+1.10	7.581*	+11*	49.33	+4.89
LR-R-101 [15]	78.5	+1.11	7.79	-12	42.0	-2.44
AA-R-101 [16]	78.7	+1.31	8.05	+248	45.4	+0.96
VT-R-101 (ours)	**81.09**	**+3.70**	**7.127**	**-675**	**40.33**	**-4.11**

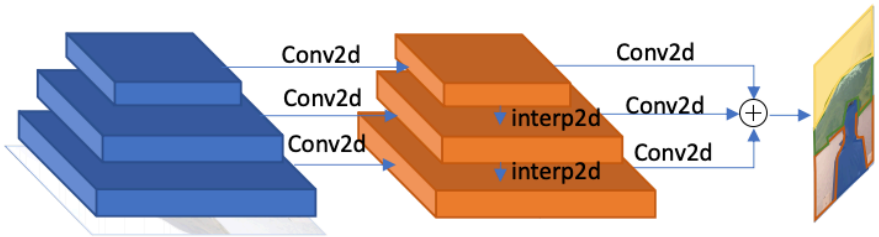




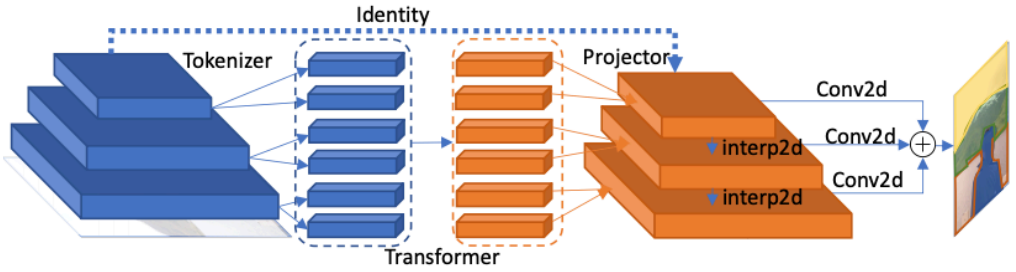
MAC = Multiply-Accumulate Operation

Results: Image Segmentation

Feature Pyramid Nets (FPN) - Lin et al. (2017)



FPN ala Visual Transformer (VT-FPN)

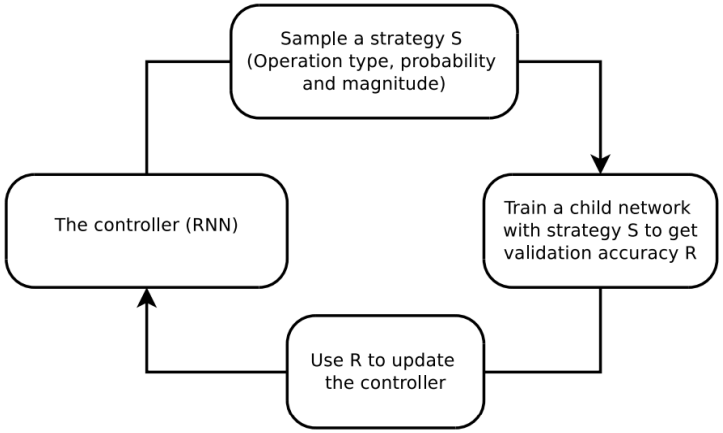


		mIoU (%)	Total MACs (G)	FPN MACs (G)	Dataset
R-50	FPN	40.78	159	55.1	COCO-stuff
	VT-FPN	41.00	113 (1.41x)	8.5 (6.48x)	
	FPN	47.04	37.1	12.8	LIP
	VT-FPN	47.39	26.4 (1.41x)	2.0 (6.40x)	
R-101	FPN	41.51	231	55.1	COCO-stuff
	VT-FPN	41.50	185 (1.25x)	8.5 (6.48x)	
	FPN	47.35	54.4	12.8	LIP
	VT-FPN	47.58	43.6 (1.25x)	2.0 (6.40x)	

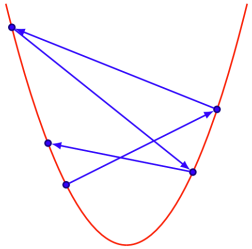
$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

Some cool training details I didn't know about

- AutoAugment - Cubuk et al. (2019)

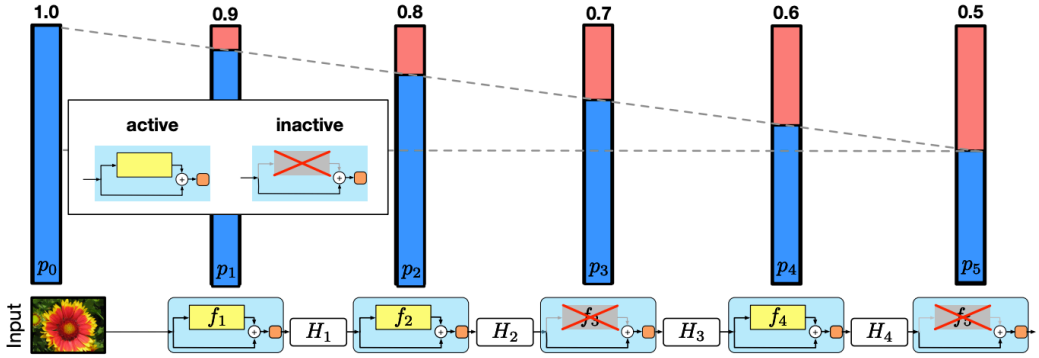


- Polyak/EM-average on weights



$$W_t^{EMA} = \alpha W_t + (1-\alpha) W_{t-1}^{EMA}$$

- Stochastic depth survival prob. - Huang et al. (2016)



$$H_\ell = \texttt{ReLU}(b_\ell f_\ell(H_{\ell-1}) + \mathrm{id}(H_{\ell-1})).$$

$$b_l \sim Bernouilli(p_l)$$

$$p_\ell = 1 - \frac{\ell}{L}(1 - p_L).$$