ICLR

ICLR 2022 – Review

Overview

Convolutions – still a Thing!

Transformers & Variants

Interesting Concepts & Applications

ICLR

Convolutions – still a Thing!

Omni-Dimensional **Dynamic Convolution**

Learning Strides in Convolutional Neural Networks

Omni-Dimensional Dynamic Convolution

OMNI-DIMENSIONAL DYNAMIC CONVOLUTION

Chao Li[1]*, **Aojun Zhou**[2], **Anbang Yao**[1]†
[1]Intel Labs China, [2]CUHK-SenseTime Joint Lab, The Chinese University of Hong Kong
chao.li3@intel.com, aojun.zhou@gmail.com, anbang.yao@intel.com

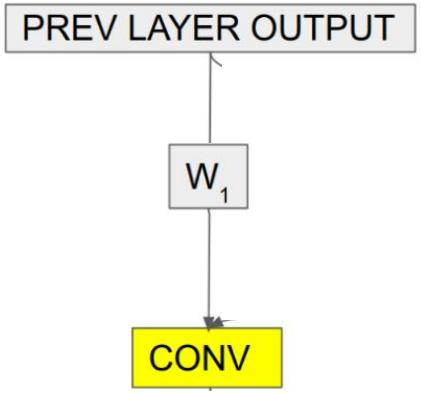
Omni-Dimensional Dynamic Convolution

OMNI-DIMENSIONAL DYNAMIC CONVOLUTION

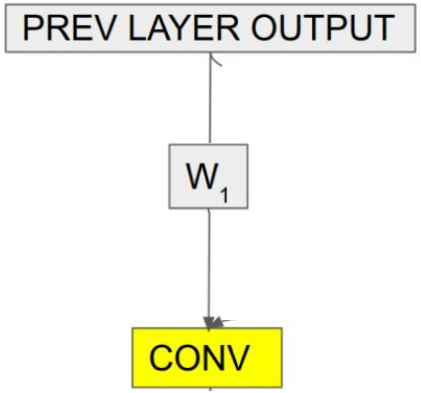
Chao Li[1]*, **Aojun Zhou**[2], **Anbang Yao**[1]†
[1]Intel Labs China, [2]CUHK-SenseTime Joint Lab, The Chinese University of Hong Kong
chao.li3@intel.com, aojun.zhou@gmail.com, anbang.yao@intel.com

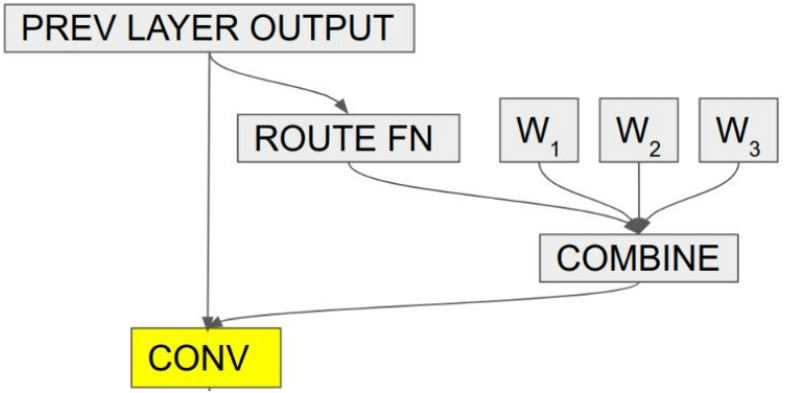
- Generalization of
  - Dynamic Weights
  - Feature Recalibration

Dynamic Weights



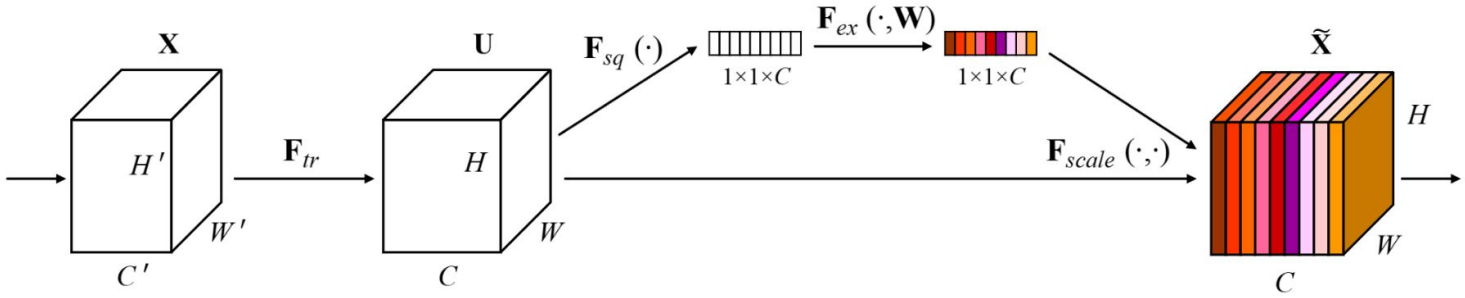
CondConv: Dynamic Weights





(a) CondConv: $(\alpha_1 W_1 + \ldots + \alpha_n W_n) * x$

Attentive Feature Recalibration: SEnet



[Submitted on 5 Sep 2017 (v1), last revised 16 May 2019 (this version, v4)]

Squeeze-and-Excitation Networks

Jie Hu, Li Shen, Samuel Albanie, Gang Sun, Enhua Wu

Attentive Feature Recalibration: SEnet

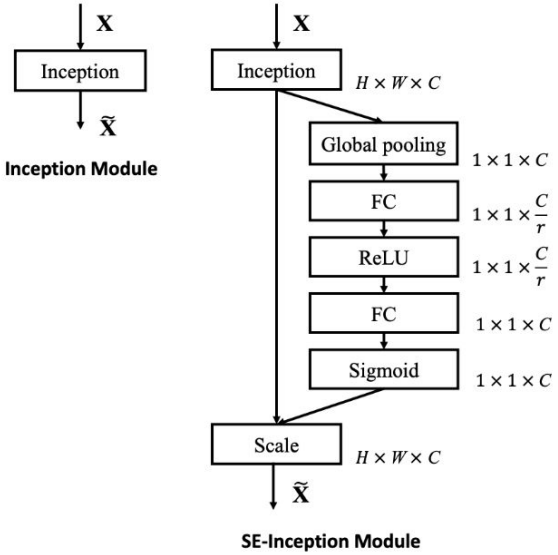


Fig. 2. The schema of the original Inception module (left) and the SE-Inception module (right).

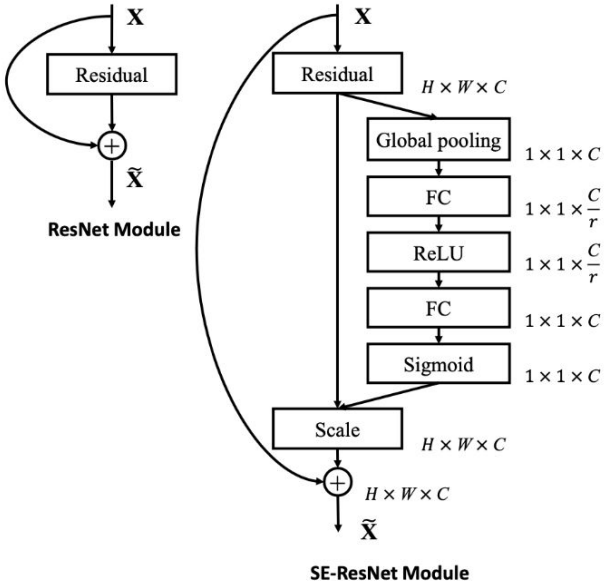


Fig. 3. The schema of the original Residual module (left) and the SE-ResNet module (right).

Omni-Dimensional Dynamic Convolution

- Feature recalibration

- Dynamic Weights:
  Making the weights of a neural network to be sample-adaptive via dynamic mechanisms

Omni-Dimensional Dynamic Convolution

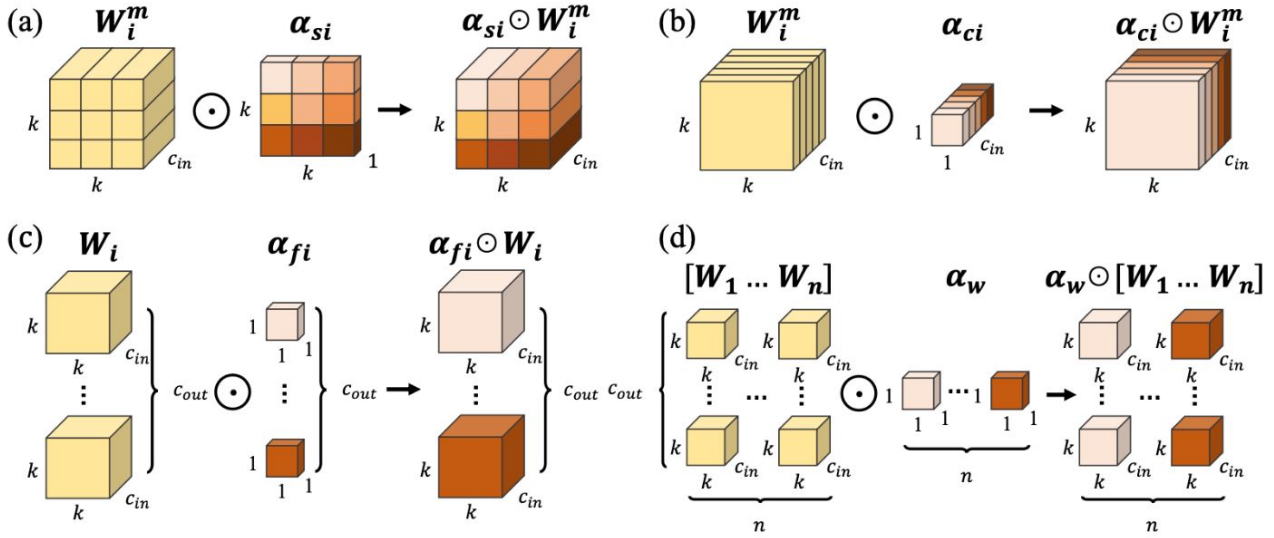


Figure 2: Illustration of multiplying four types of attentions in ODConv to convolutional kernels progressively. (a) Location-wise multiplication operations along the spatial dimension, (b) channel-wise multiplication operations along the input channel dimension, (c) filter-wise multiplication operations along the output channel dimension, and (d) kernel-wise multiplication operations along the kernel dimension of the convolutional kernel space. Notations are clarified in the Method section.

Omni-Dimensional Dynamic Convolution

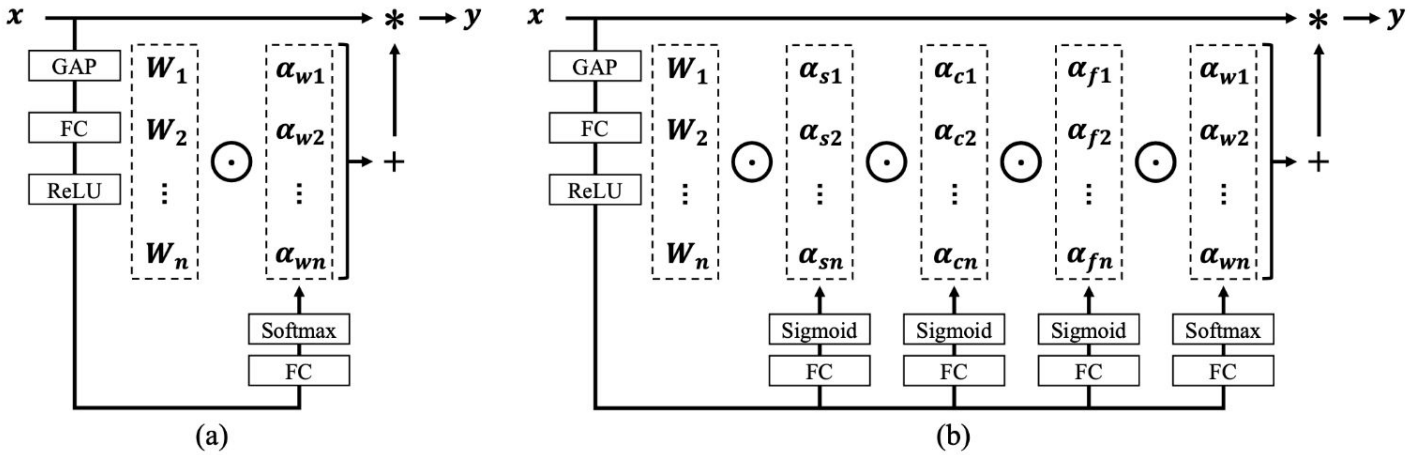


Figure 1: A schematic comparison of (a) DyConv (CondConv uses GAP+FC+Sigmoid) and (b) ODConv. Unlike CondConv and DyConv which compute a single attention scalar α_{wi} for the convolutional kernel W_i, ODConv leverages a novel multi-dimensional attention mechanism to compute four types of attentions α_{si}, α_{ci}, α_{fi} and α_{wi} for W_i along all four dimensions of the kernel space in a parallel manner. Their formulations and implementations are clarified in the Method section.

Convolutions – still a Thing!

Omni-Dimensional **Dynamic Convolution**

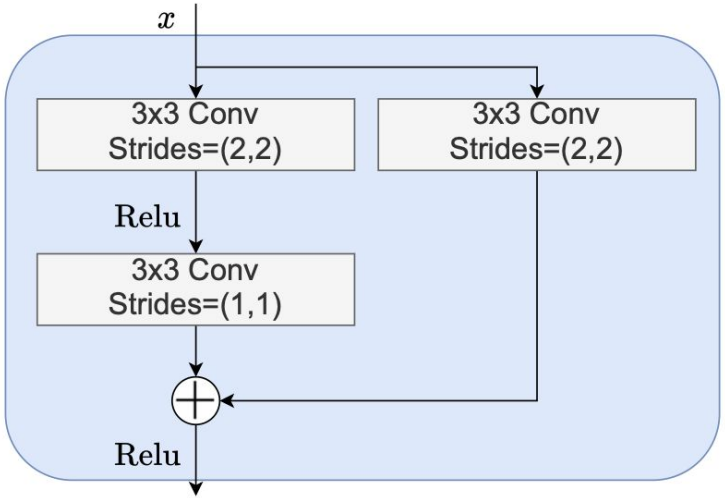
Learning Strides in Convolutional Neural Networks

Learning Strides in CNNs

LEARNING STRIDES IN CONVOLUTIONAL NEURAL NETWORKS

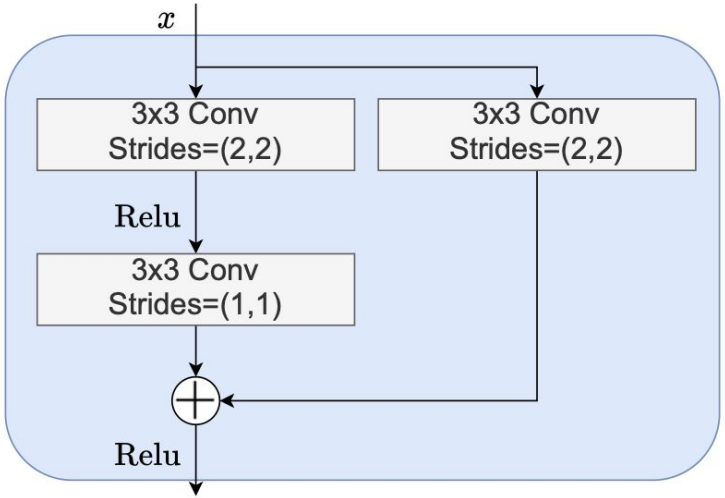
Rachid Riad[*1], **Olivier Teboul**[2], **David Grangier**[2] **& Neil Zeghidour**[2]
[1]ENS, INRIA, INSERM, UPEC, PSL Research University
[2]Google Research
rachid.riad@ens.fr, {teboul, grangier, neilz}@google.com

Learning Strides in CNNs

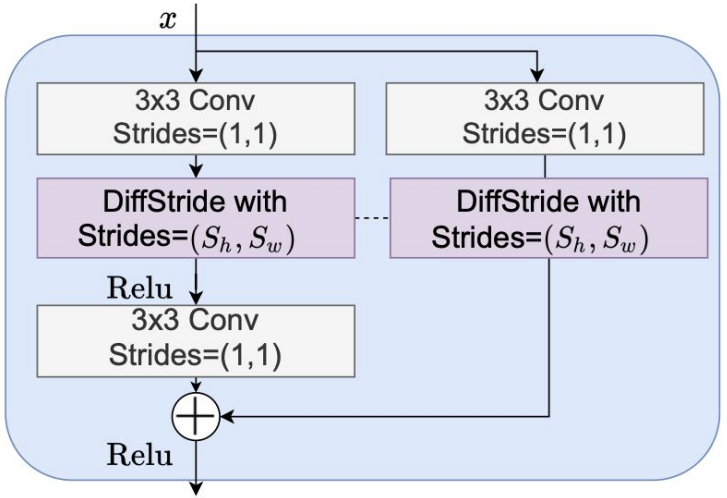


(a) Residual block with a strided convolution.

Learning Strides in CNNs



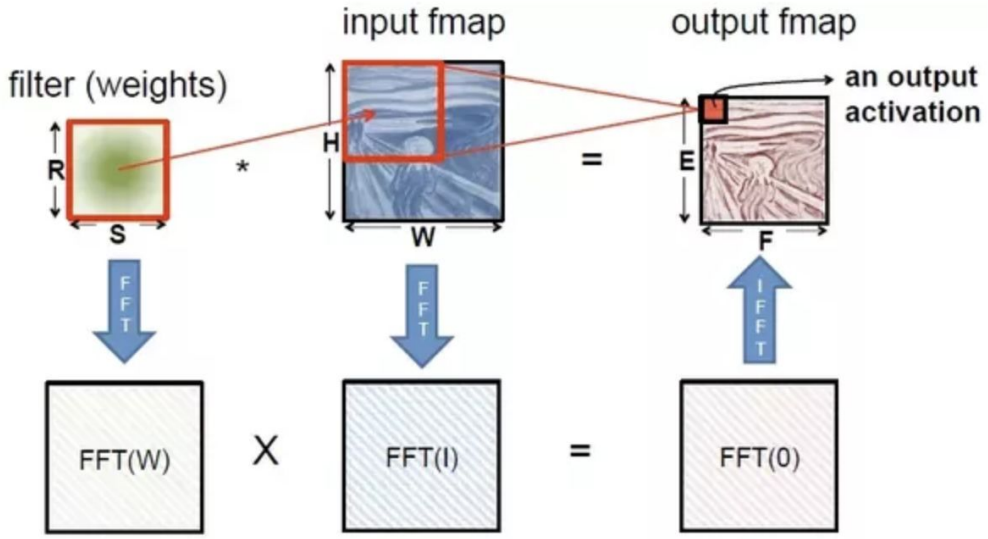
(a) Residual block with a strided convolution.



(b) Residual block with a shared DiffStride layer.

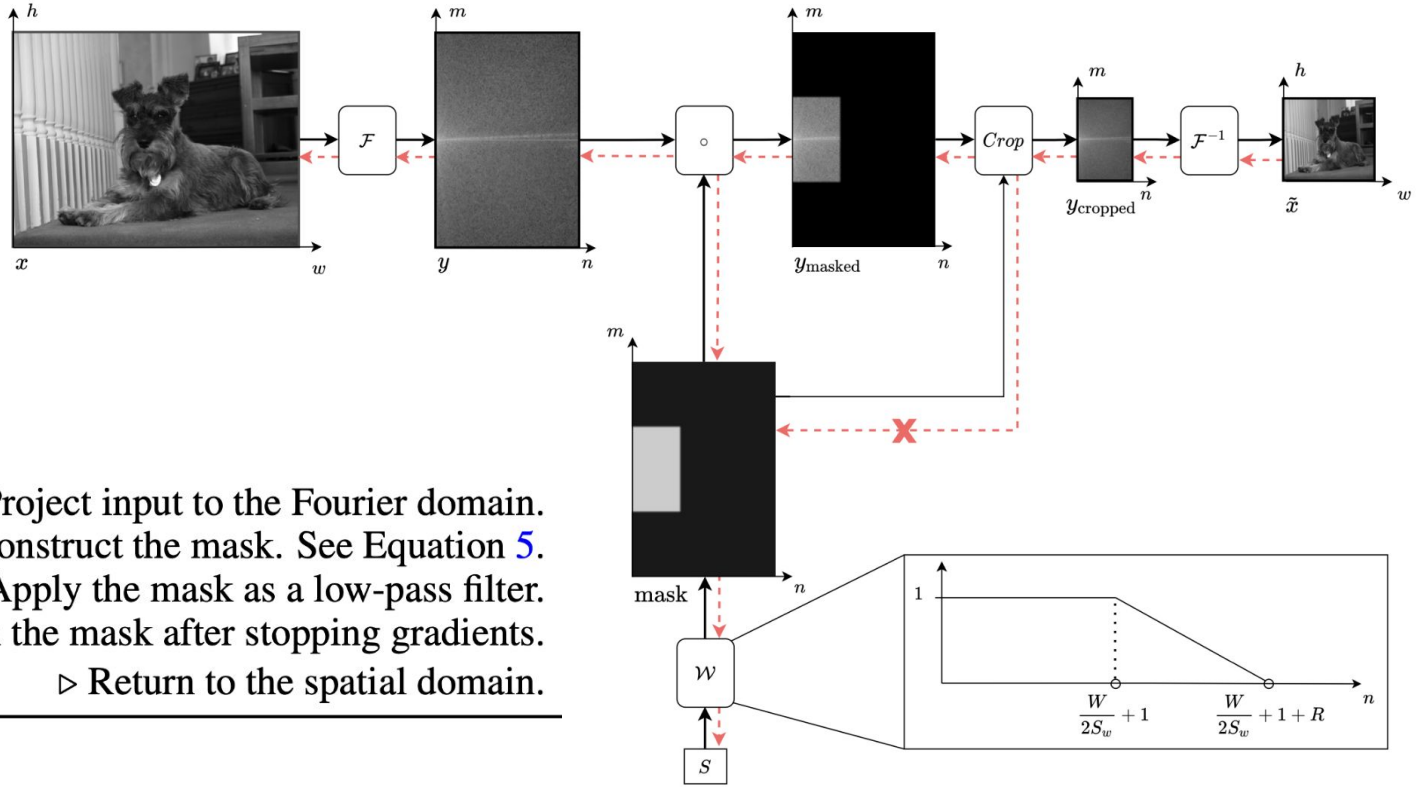
Learning Strides in CNNs

- Convolution in pixel domain becomes Multiplicationin in Fourier domain:



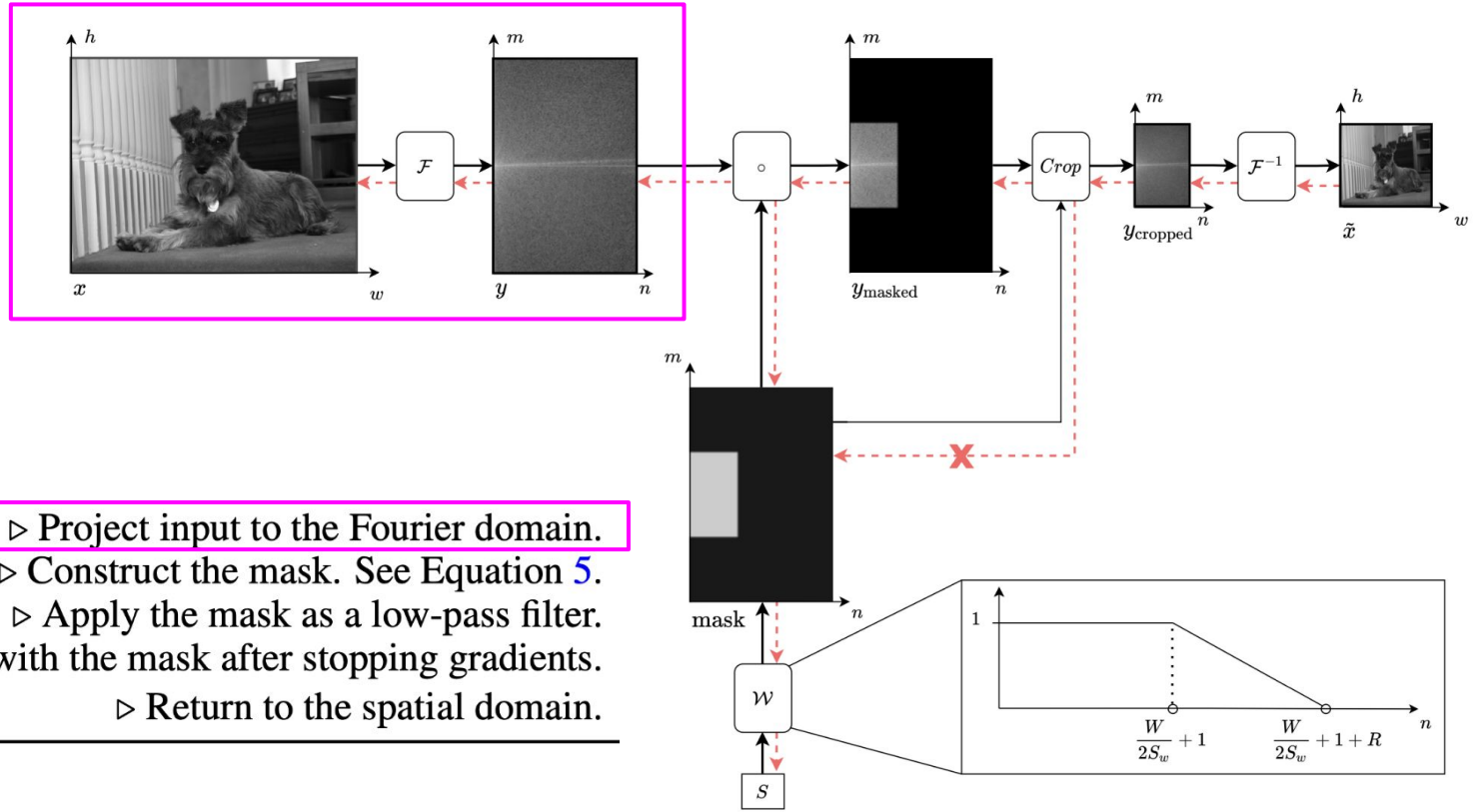
https://analyticsindiamag.com/how-fourier-transform-is-used-in-deep-learning/

Learning Strides in CNNs



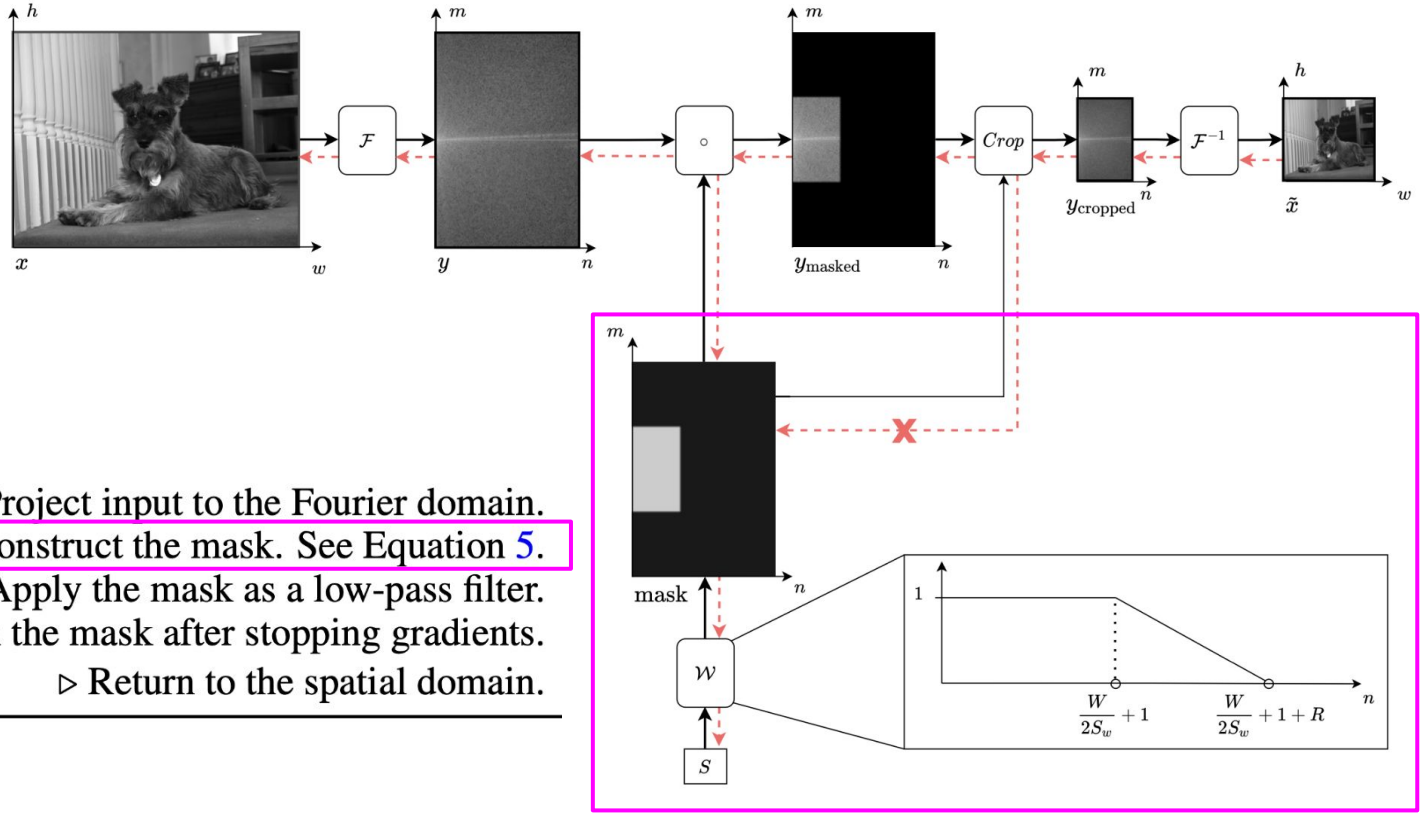
▷ Project input to the Fourier domain.
▷ Construct the mask. See Equation 5.
▷ Apply the mask as a low-pass filter.
▷ Crop the tensor with the mask after stopping gradients.
▷ Return to the spatial domain.

Learning Strides in CNNs



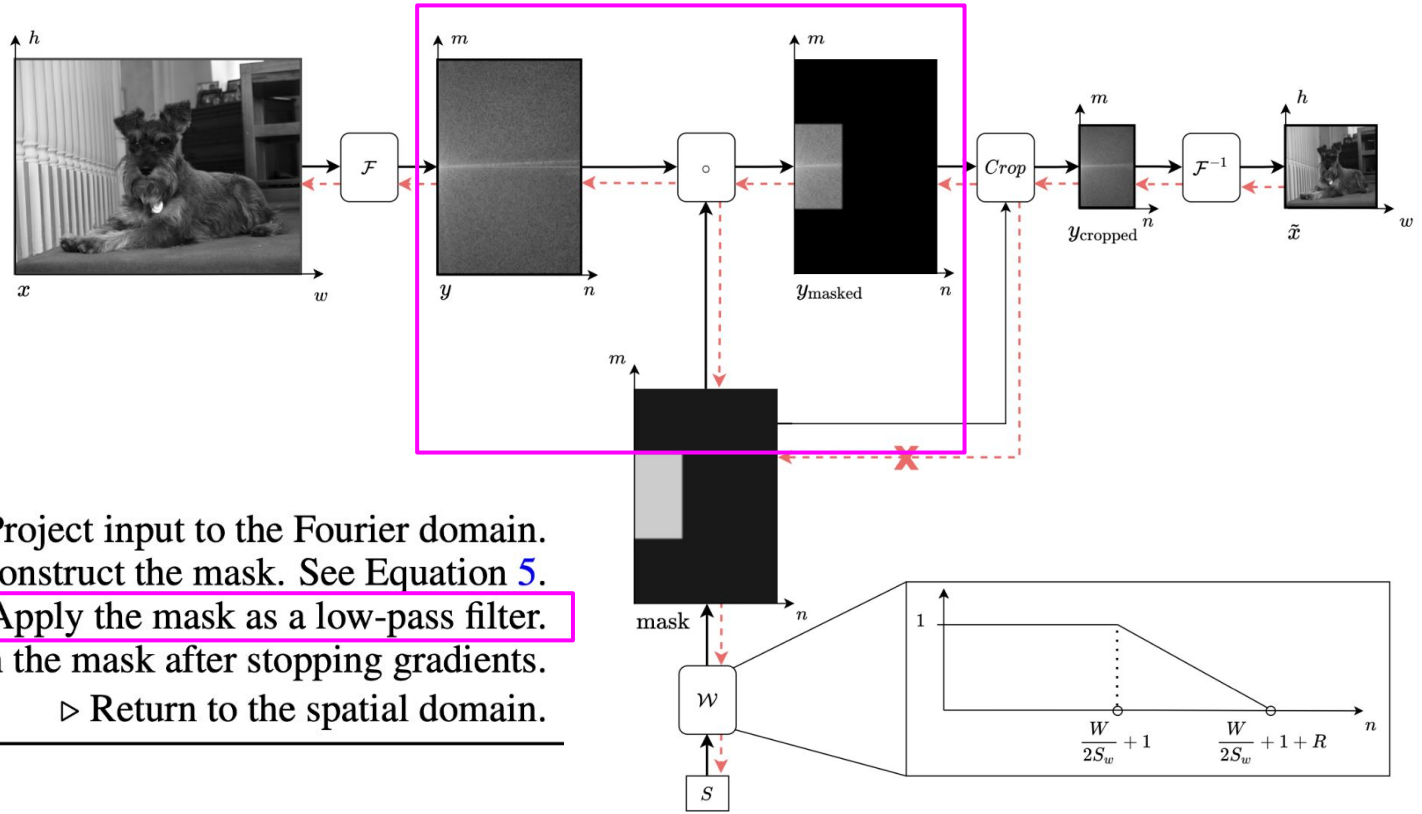
▷ Project input to the Fourier domain.
▷ Construct the mask. See Equation 5.
▷ Apply the mask as a low-pass filter.
▷ Crop the tensor with the mask after stopping gradients.
▷ Return to the spatial domain.

Learning Strides in CNNs



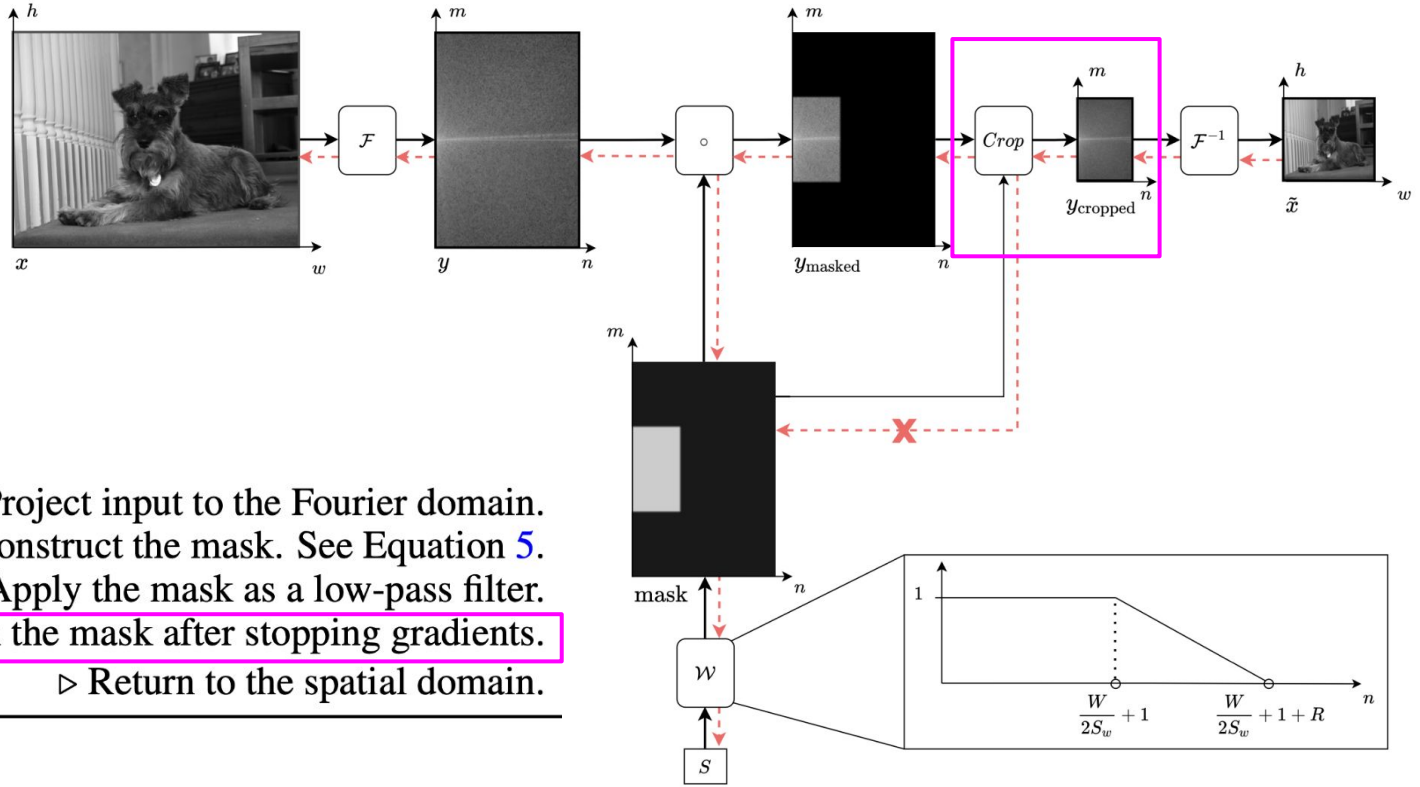
▷ Project input to the Fourier domain.
▷ Construct the mask. See Equation 5.
▷ Apply the mask as a low-pass filter.
▷ Crop the tensor with the mask after stopping gradients.
▷ Return to the spatial domain.

Learning Strides in CNNs



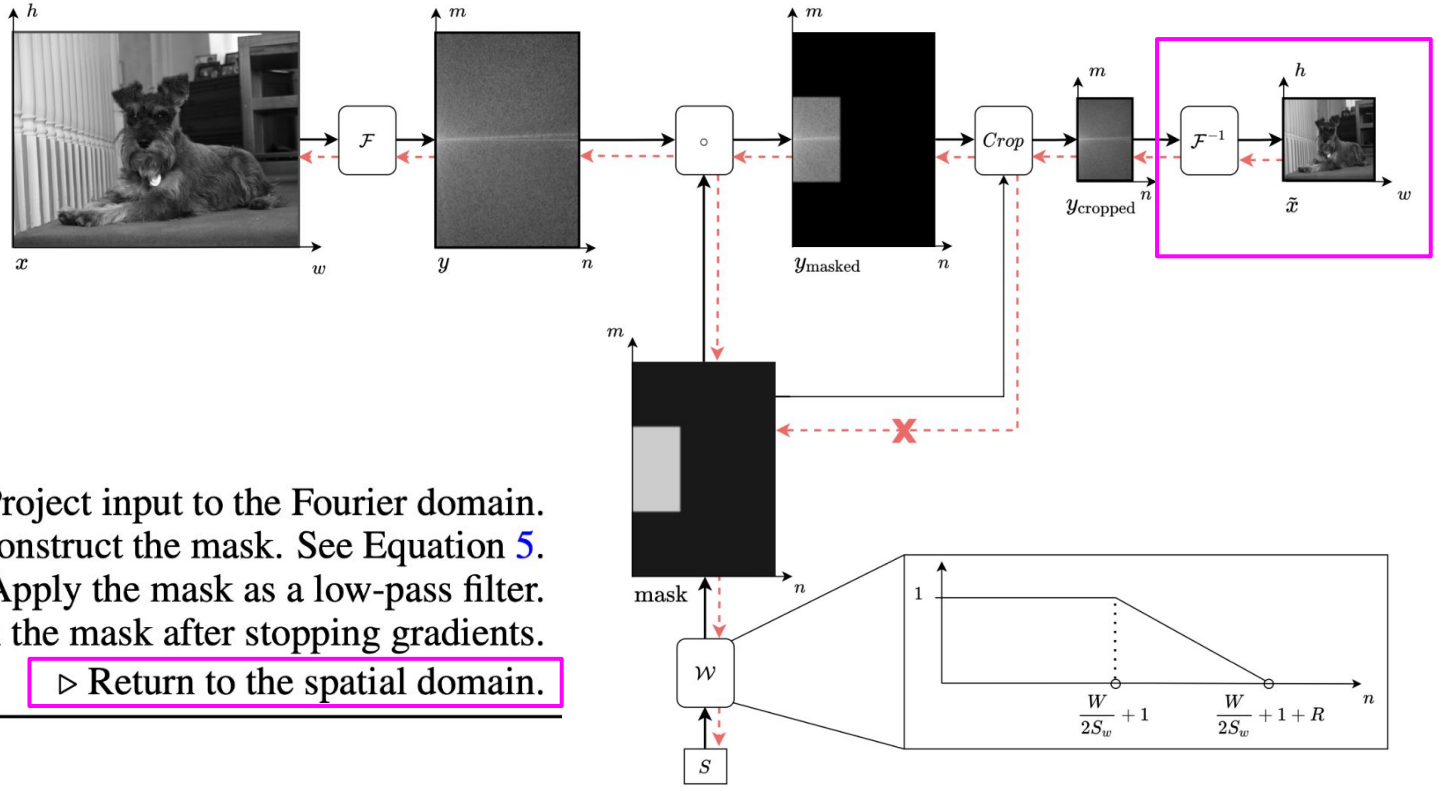
▷ Project input to the Fourier domain.
▷ Construct the mask. See Equation 5.
▷ Apply the mask as a low-pass filter.
▷ Crop the tensor with the mask after stopping gradients.
▷ Return to the spatial domain.

Learning Strides in CNNs



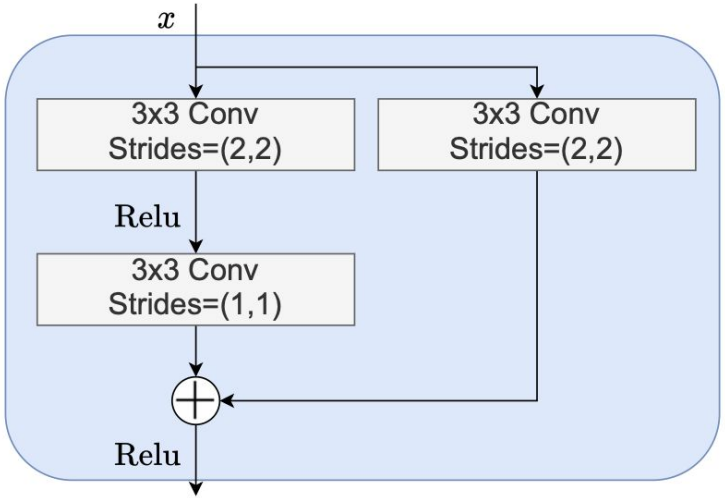
▷ Project input to the Fourier domain.
▷ Construct the mask. See Equation 5.
▷ Apply the mask as a low-pass filter.
▷ Crop the tensor with the mask after stopping gradients.
▷ Return to the spatial domain.

Learning Strides in CNNs

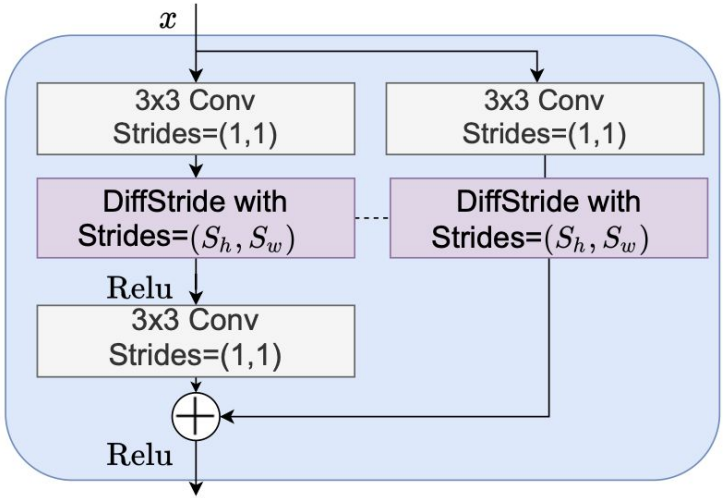


▷ Project input to the Fourier domain.
▷ Construct the mask. See Equation 5.
▷ Apply the mask as a low-pass filter.
▷ Crop the tensor with the mask after stopping gradients.
▷ Return to the spatial domain.

Learning Strides in CNNs



(a) Residual block with a strided convolution.



(b) Residual block with a shared DiffStride layer.

Learning Strides in CNNs

- Strides control “how quickly the CNNs shrinks”
- Provides fine grained control and regularization over memory and compute

following regularizer to our training loss:

$$\lambda J((S^l)_{l=1}^{l=L}) = \lambda \sum_{l=1}^{l=L} \prod_{i=1}^{l} \frac{1}{S_h^i \times S_w^i},$$

Overview

Convolutions – still a thing!

Transformers & Variants

Interesting concepts and applications

ICLR

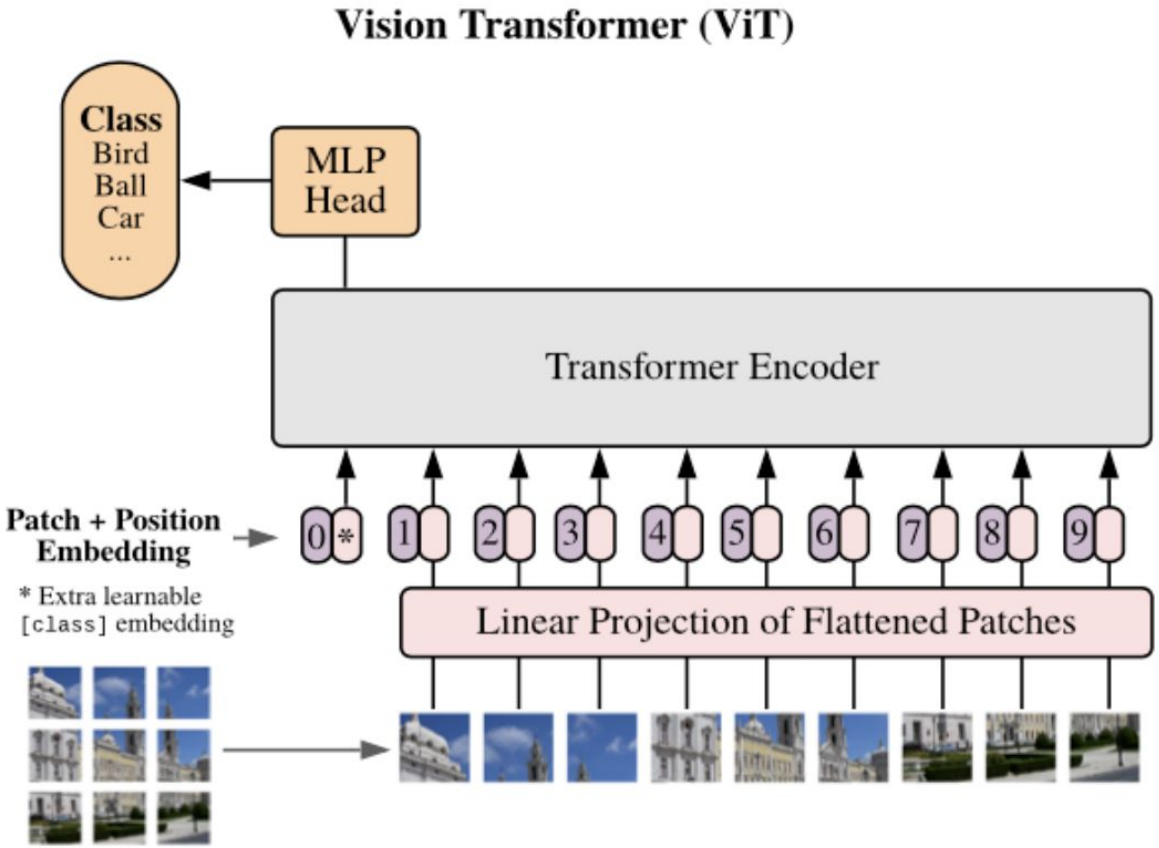
Transformers & Variants

Quick Refresher: **Transformers**

How Do Vision Transformers Work?

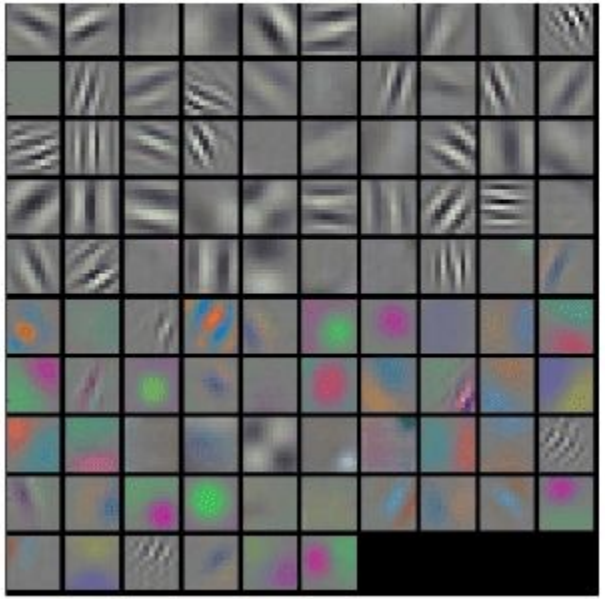
Perceiver IO:
A General Architecture For Structured Inputs & Outputs

Quick Refresher: Transformers

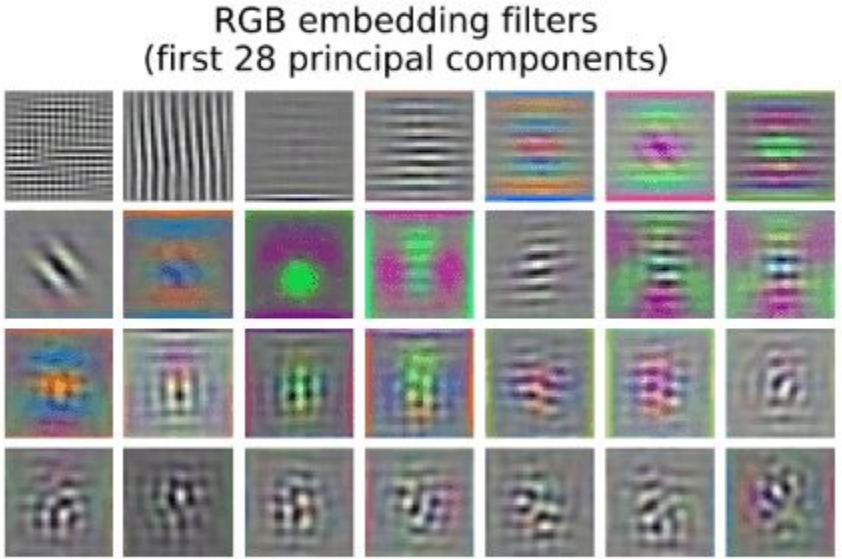


Quick Refresher: Transformers

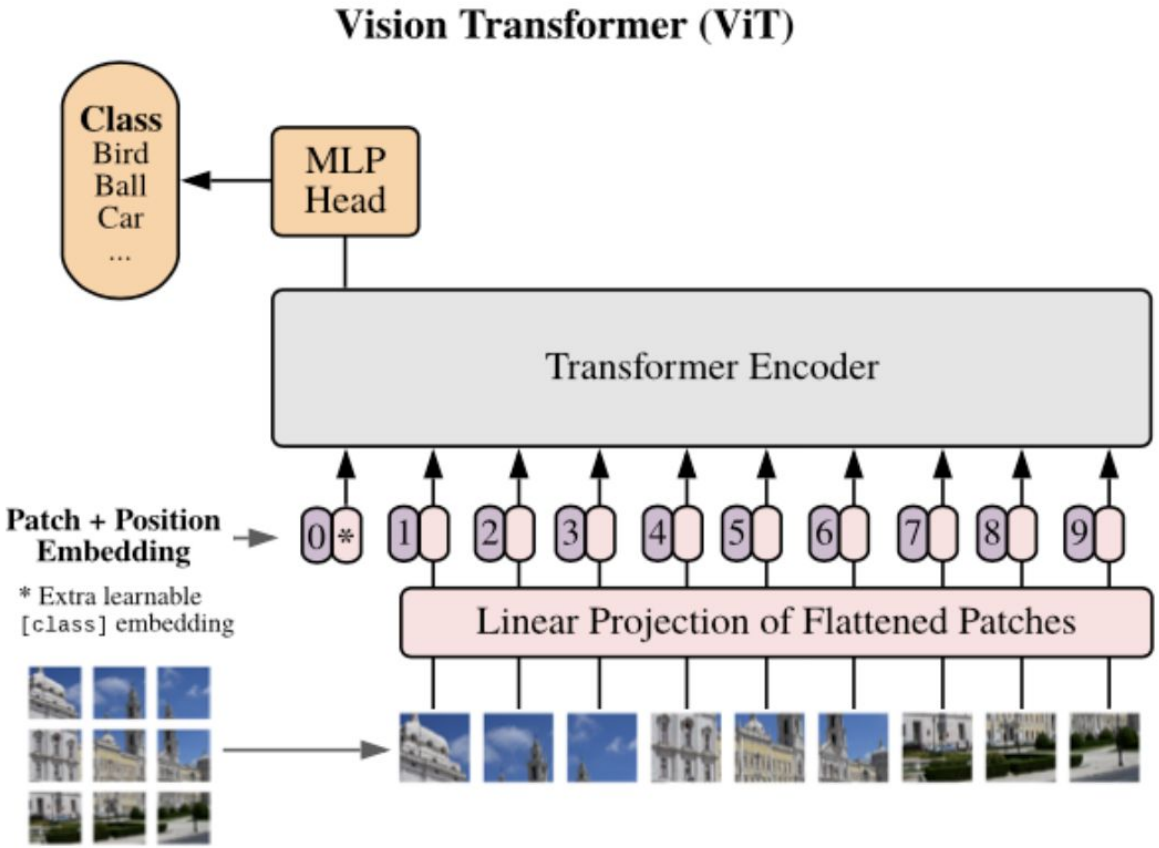
Alexnet 1st conv filters



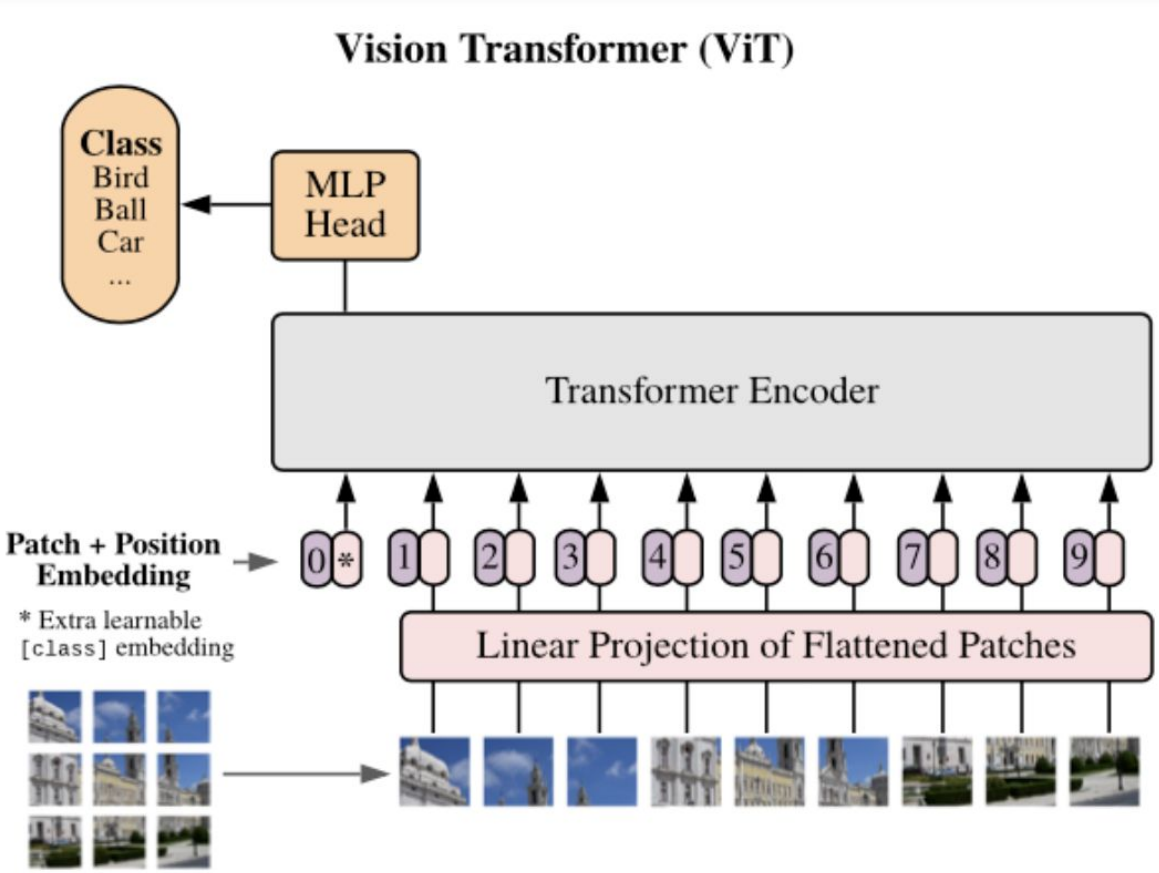
ViT 1st linear embedding filters

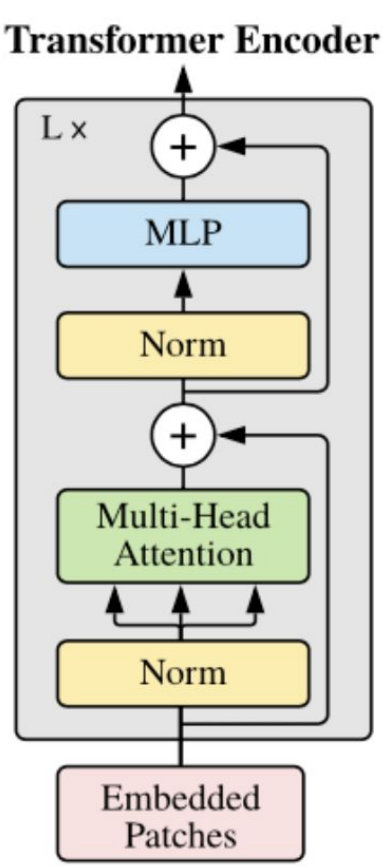


Quick Refresher: Transformers

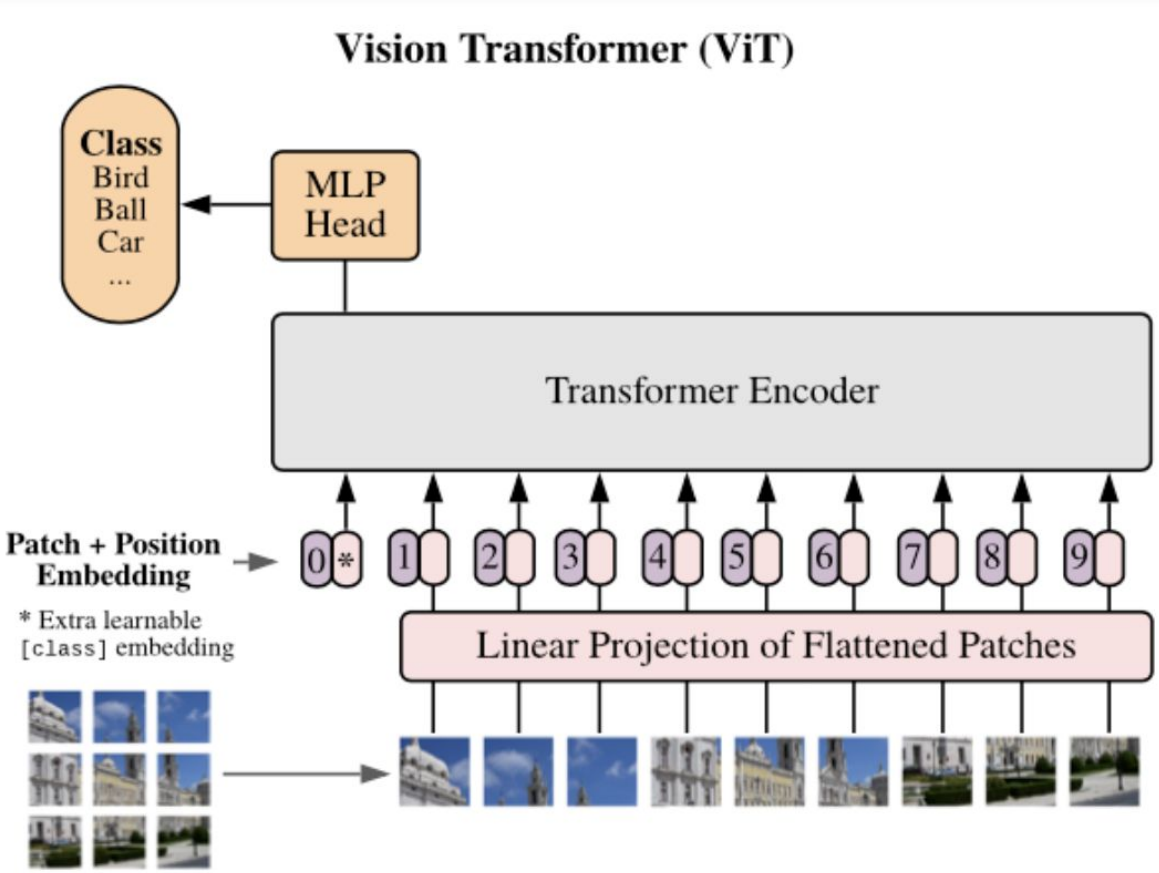


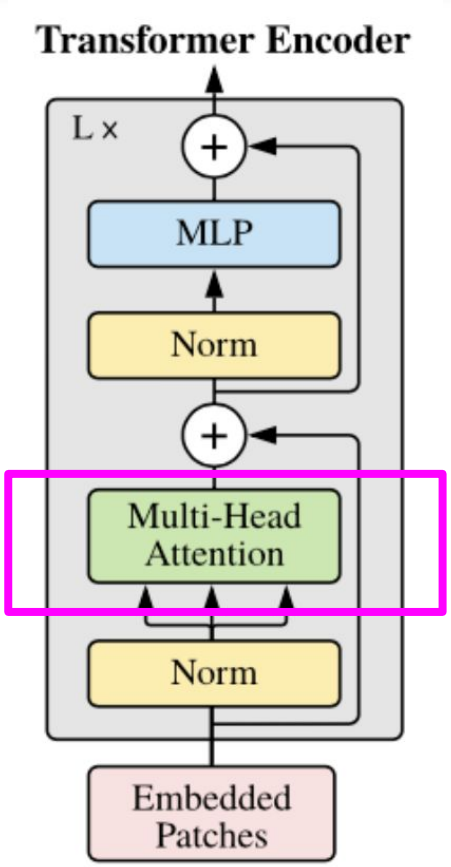
Quick Refresher: Transformers





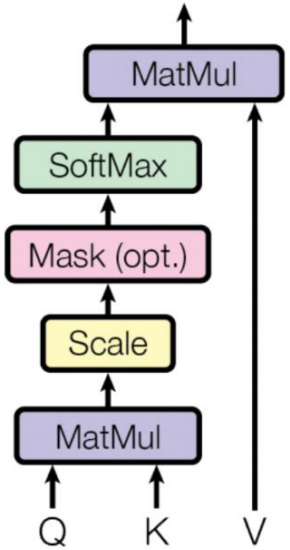
Quick Refresher: Transformers



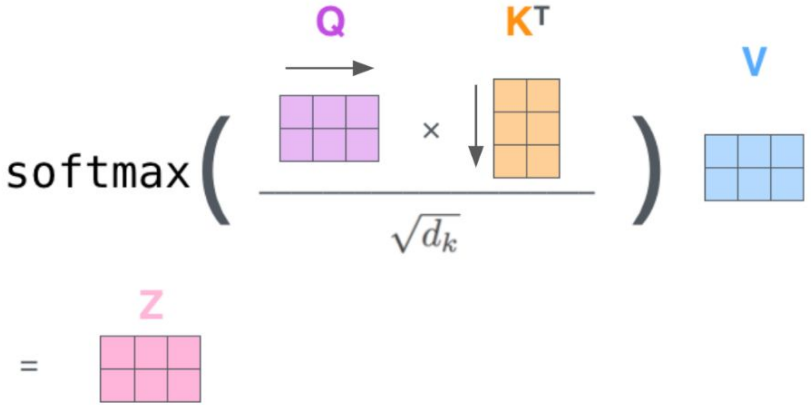


Quick Refresher: Transformers

Scaled Dot-Product Attention



$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$



Transformers & Variants

Quick Refresher: **Transformers**

How Do Vision Transformers Work? (AlterNet)

Perceiver IO:
A General Architecture For Structured Inputs & Outputs

HOW DO VISION TRANSFORMERS WORK?

Namuk Park[1,2], **Songkuk Kim**[1]
[1]Yonsei University, [2]NAVER AI Lab
{namuk.park,songkuk}@yonsei.ac.kr

How Do Vision Transformers Work?

- Common explanations for Transformers' success:

  "Have weak inductive bias"

  "They capture of long-range dependencies"

How Do Vision Transformers Work?

- Common explanations for Transformers' success:

  "Have weak inductive bias"

  "They capture of long-range dependencies"

- Commonly assumed drawback:

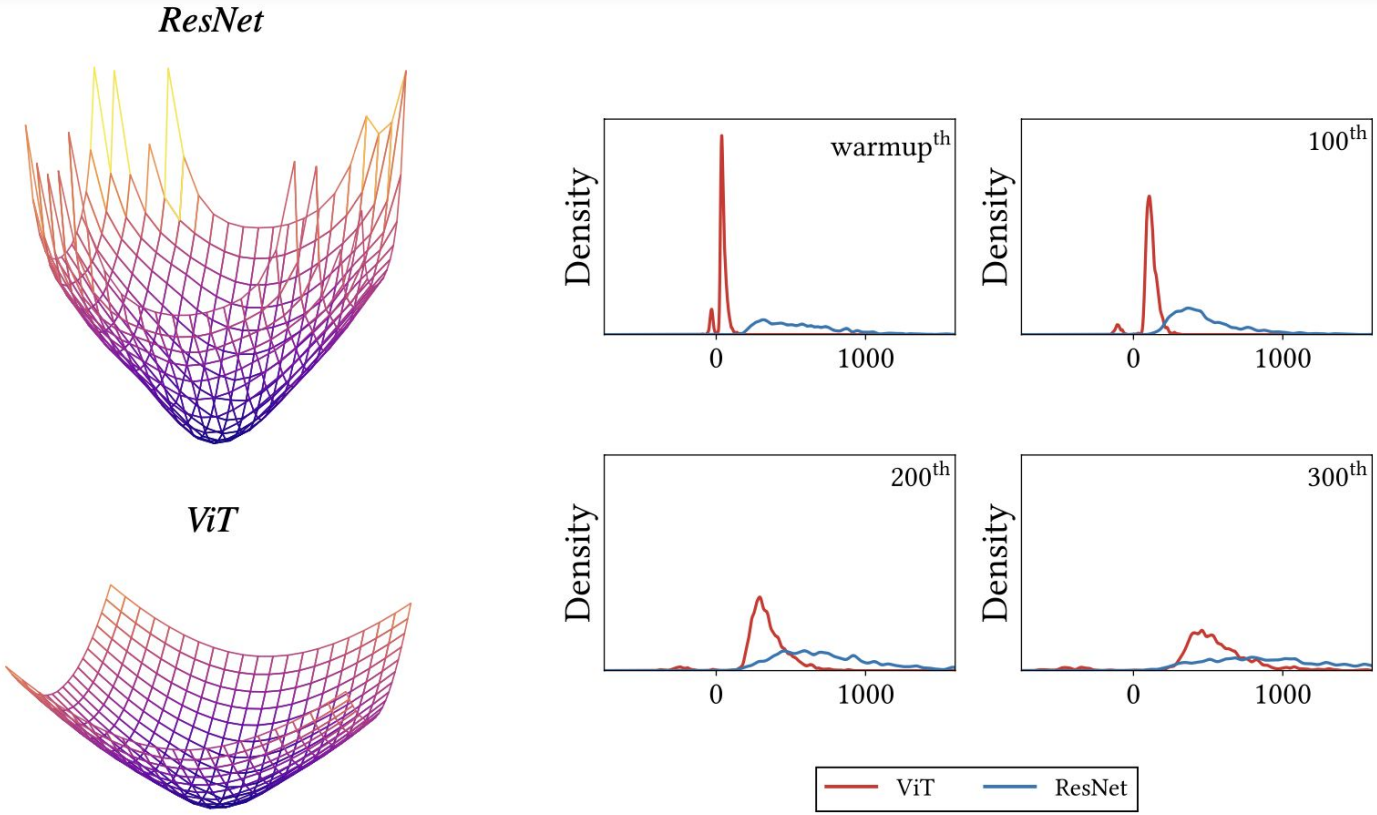
  "Tendency to overfit small datasets"

Results

multi-head self-attentions (MSAs)

- MSAs improve not only accuracy but also generalization by flattening the loss landscapes
- MSAs and Convs exhibit opposite behaviors
  MSAs are low-pass filters, but Convs are high-pass filters
- Multi-stage neural networks behave like a series connection of small individual models

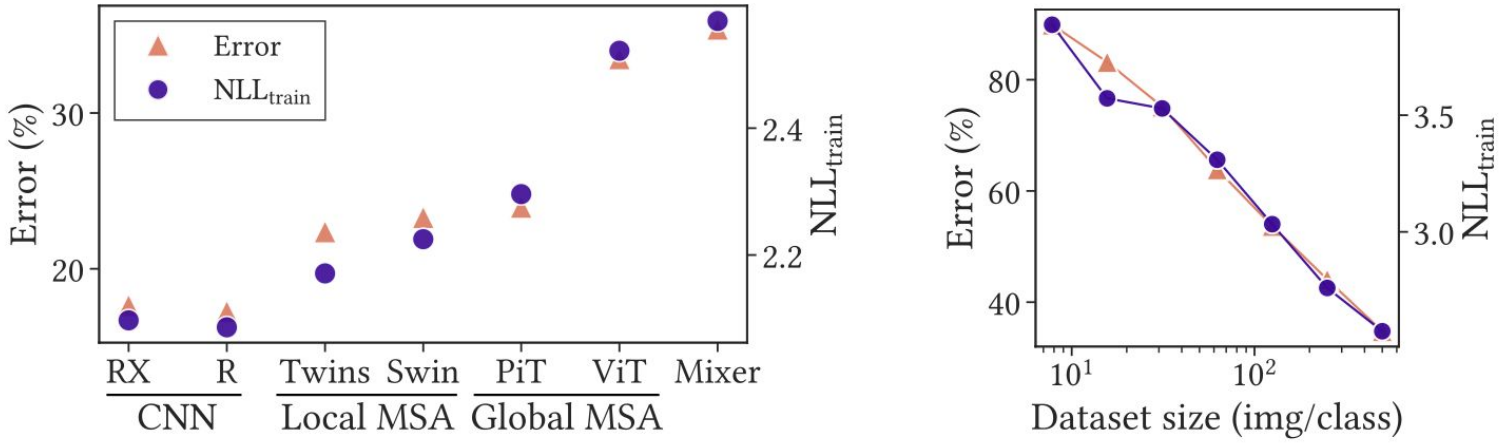
MSAs flatten the loss landscape



(a) Loss landscape visualizations

(b) Hessian max eigenvalue spectra

MSAs don't actually overfit

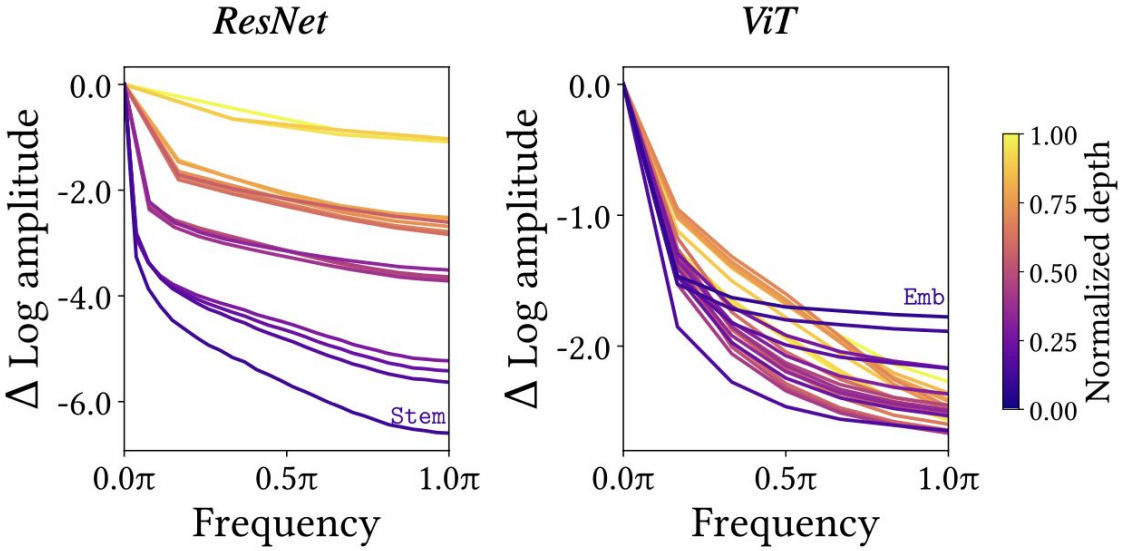


(a) Error and NLL$_{train}$ for each model.

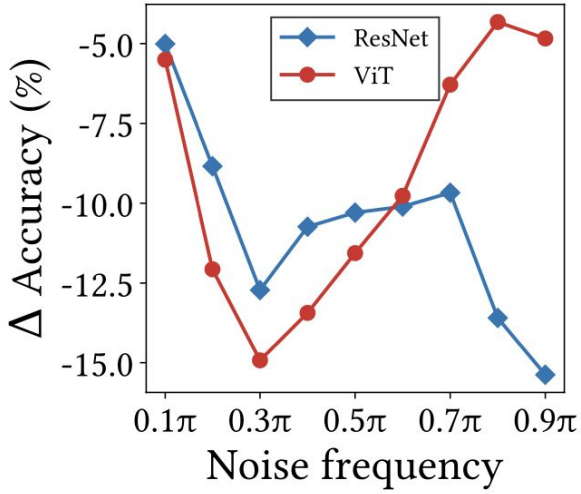
(b) Performance of ViT for dataset size.

Figure 5: **ViT does not overfit training datasets.** "R" is ResNet and "RX" is ResNeXt. *Left:* Weak inductive bias disturbs NN optimization. The lower the NLL$_{train}$, the lower the error. *Right:* The lack of dataset also disturbs NN optimization.

MSAs are low-pass filters, CNNs high-pass filters



(a) Relative log amplitudes of Fourier transformed feature maps.



(b) Robustness for noise frequency

MSAs reduce feature variance, CNNs increase it

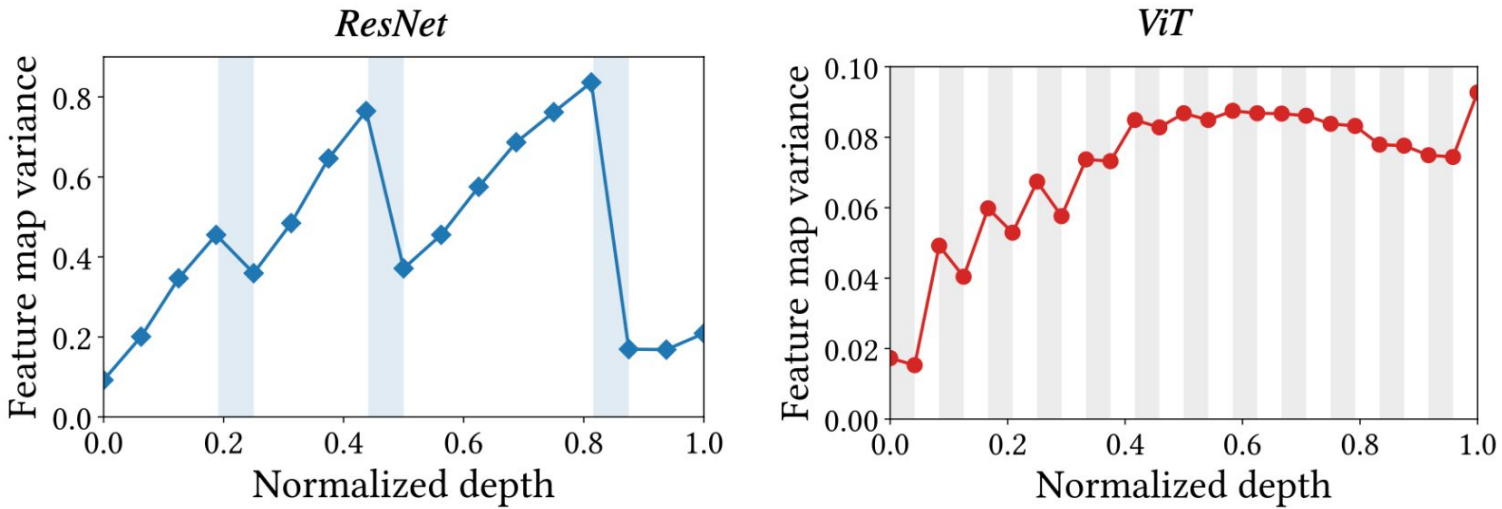
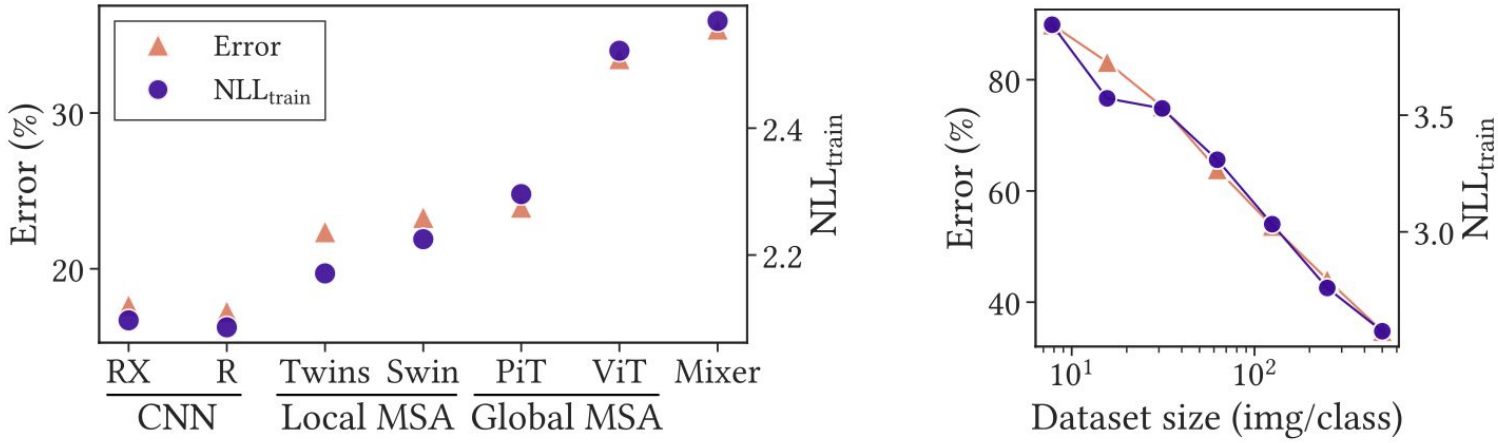


Figure 9: **MSAs (gray area) reduce the variance of feature map points, but Convs (white area) increase the variance.** The blue area is subsampling layer. This result implies that MSAs ensemble feature maps, but Convs do not.

MSAs dont actually overfit

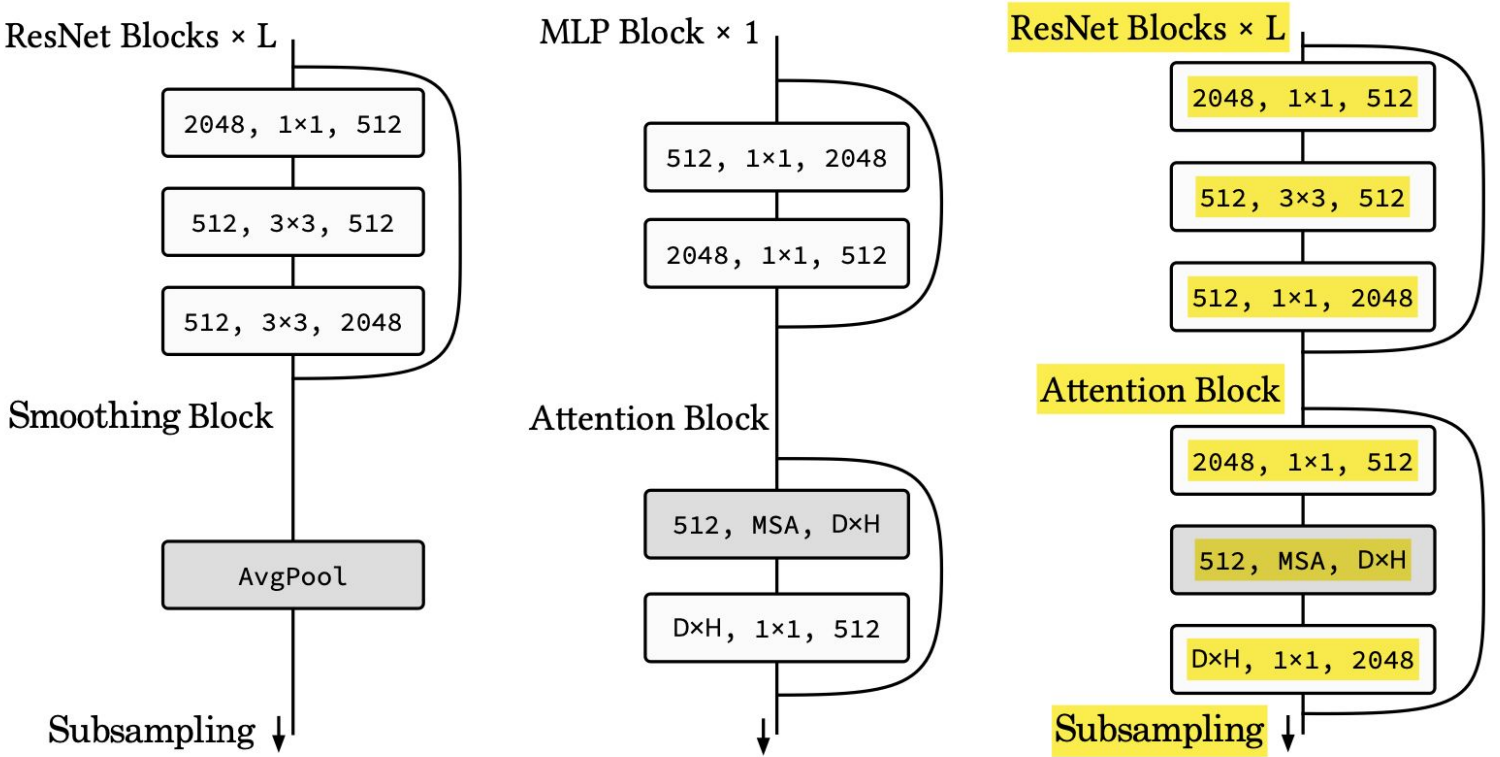


(a) Error and NLL$_{train}$ for each model.

(b) Performance of ViT for dataset size.

Figure 5: **ViT does not overfit training datasets.** "R" is ResNet and "RX" is ResNeXt. *Left:* Weak inductive bias disturbs NN optimization. The lower the NLL$_{train}$, the lower the error. *Right:* The lack of dataset also disturbs NN optimization.

AlterNet



(a) Spatial smoothing (b) Canonical Transformer (c) Alternating pattern (*ours*)

AlterNet

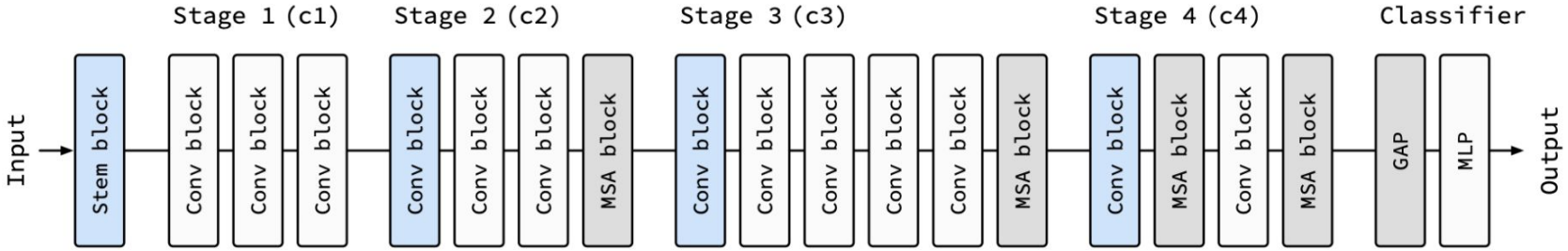
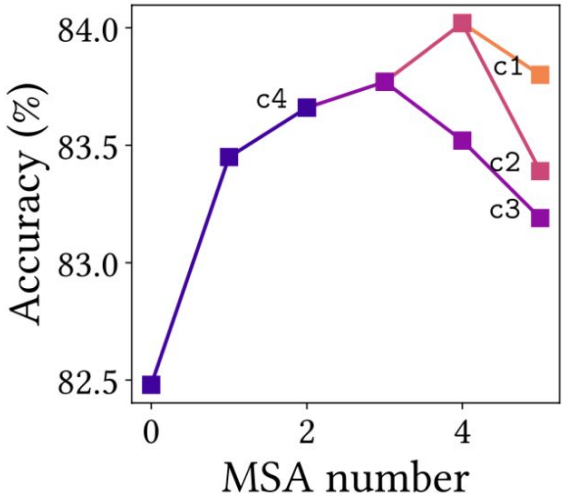
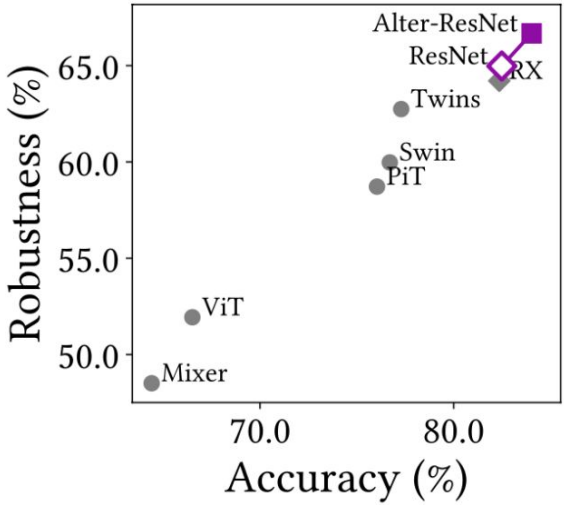


Figure 11: **Detailed architecture of Alter-ResNet-50 for CIFAR-100.** White, gray, and blue blocks mean Conv, MSA, and subsampling blocks. All stages (except stage 1) end with MSA blocks. This model is based on pre-activation ResNet-50. Following Swin, MSAs in stages 1 to 4 have 3, 6, 12, and 24 heads, respectively.

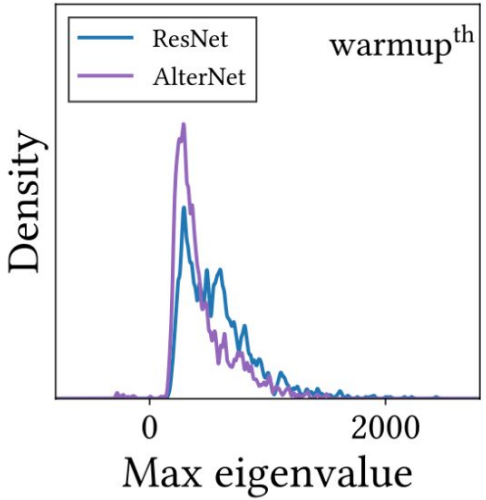
AlterNet



(a) Accuracy of AlterNet for MSA number



(b) Accuracy and robustness in a small data regime (CIFAR-100)



(c) Hessian max eigenvalue spectra in an early phase of training

Transformers & Variants

Quick Refresher: **Transformers**

How Do Vision Transformers Work?

Perceiver IO:
A General Architecture For Structured Inputs & Outputs

Perceiver IO

PERCEIVER IO: A GENERAL ARCHITECTURE FOR STRUCTURED INPUTS & OUTPUTS

Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu,

David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier Hénaff,

Matthew M. Botvinick, Andrew Zisserman, Oriol Vinyals, João Carreira

DeepMind

A single architecture for text, images, multi-modal ...

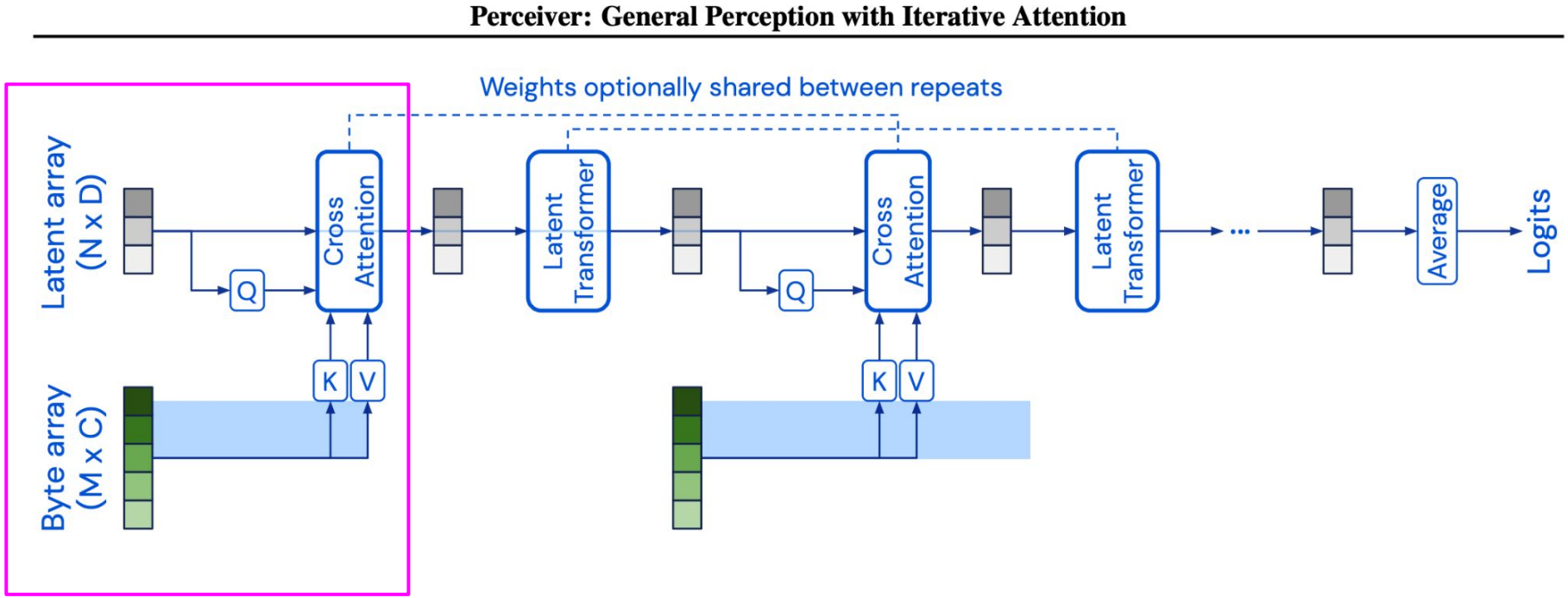
...I saw a sunset in Querétaro that seemed to reflect the colour of a rose in Bengal; I saw my empty bedroom; I saw in a closet in Alkmaar a terrestrial globe between two mirrors that multiplied it endlessly; I saw horses with flowing manes on a shore of the Caspian Sea at dawn; I saw the delicate bone structure of a hand...

Sentiment? | Grammatical?

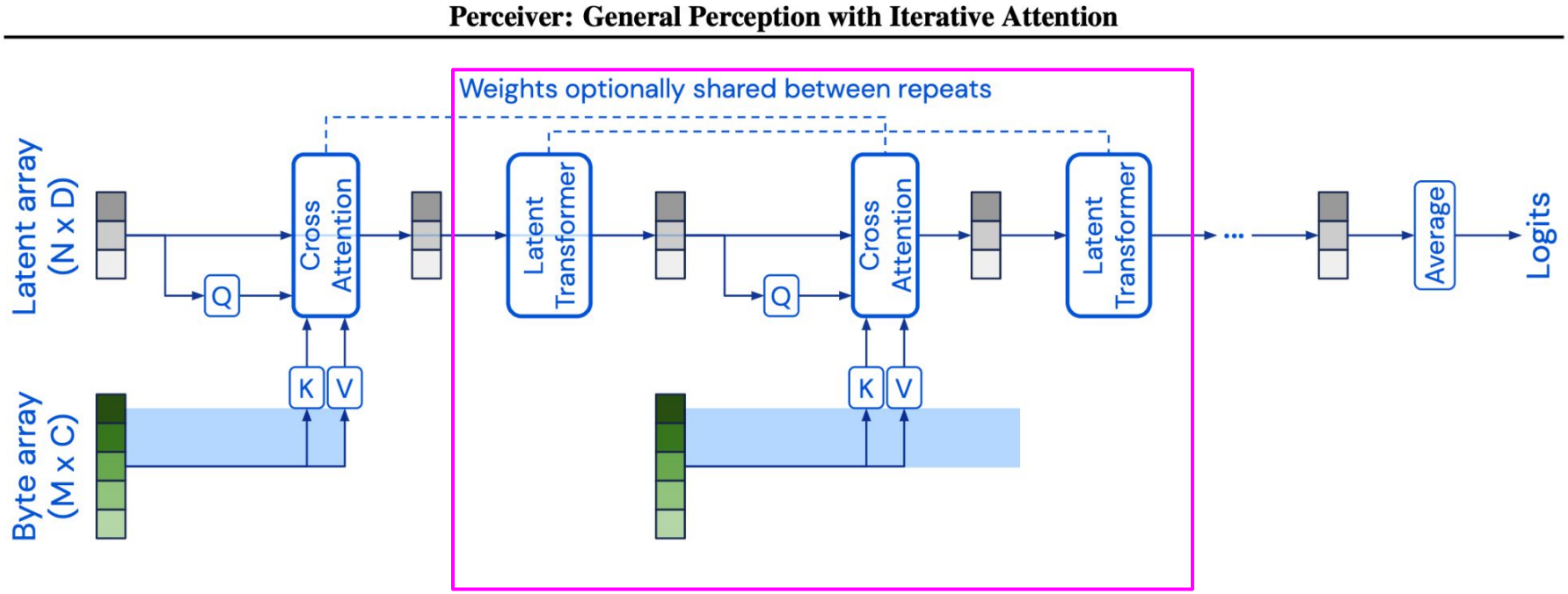
Paraphrase? | Entailment?

Label: Drumming

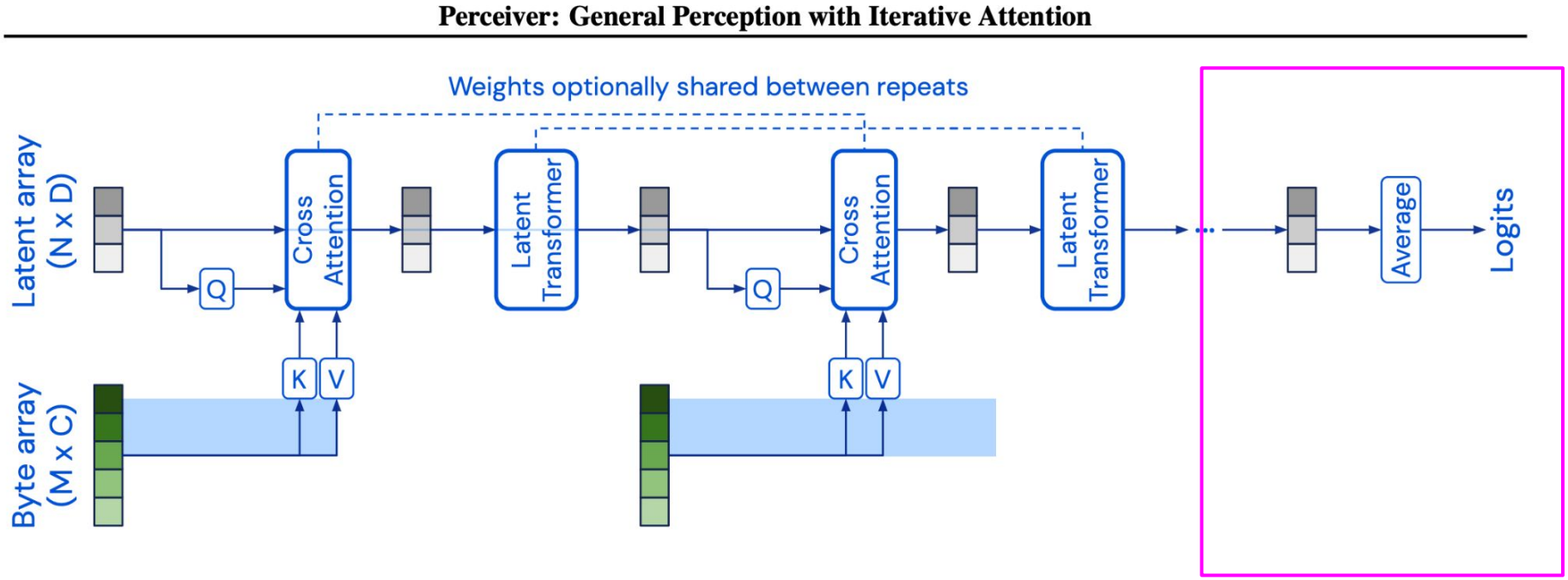
Based on Perceiver



Based on Perceiver



Based on Perceiver



Outputs: Query the latent space with QKV module!

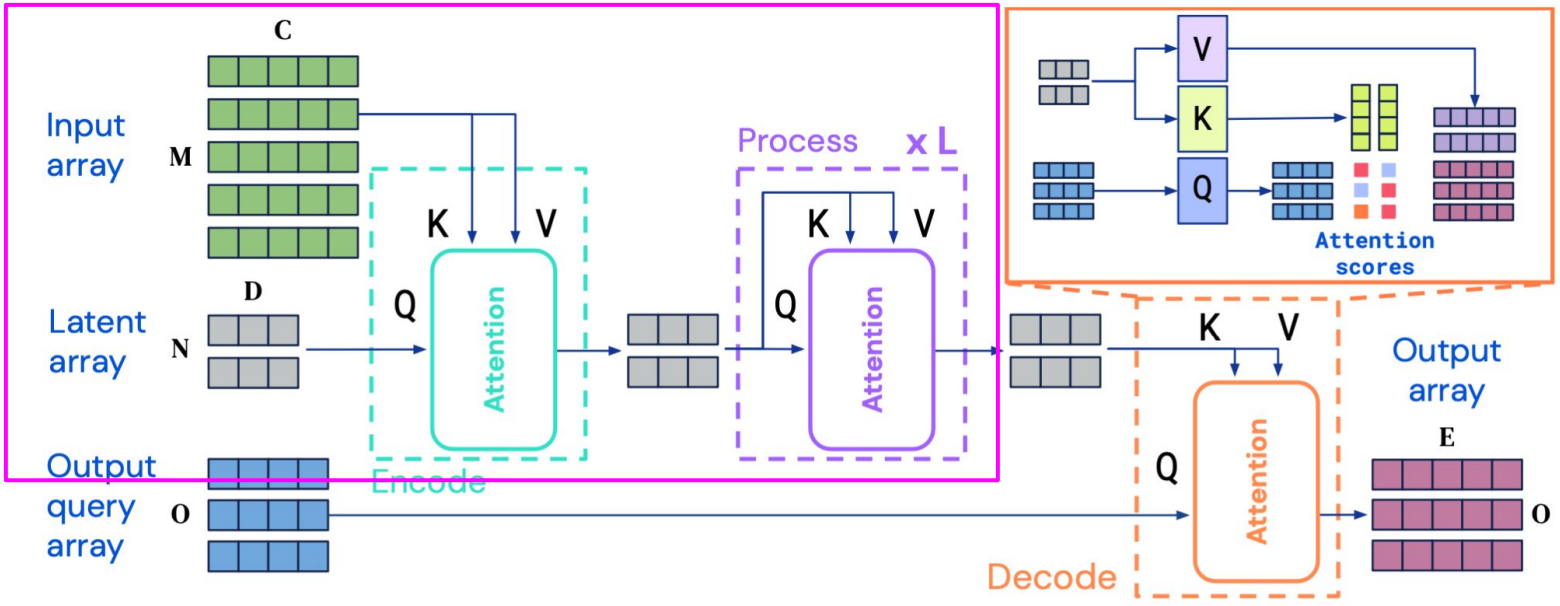


Figure 2: The Perceiver IO architecture. Perceiver IO maps arbitrary input arrays to arbitrary output arrays in a domain agnostic process. The bulk of the computation happens in a latent space whose size is typically smaller than the inputs and outputs, which makes the process computationally tractable even for very large inputs & outputs. See Fig. 5 for a more detailed look at encode, process, and decode attention.

Outputs: Query the latent space with QKV module!

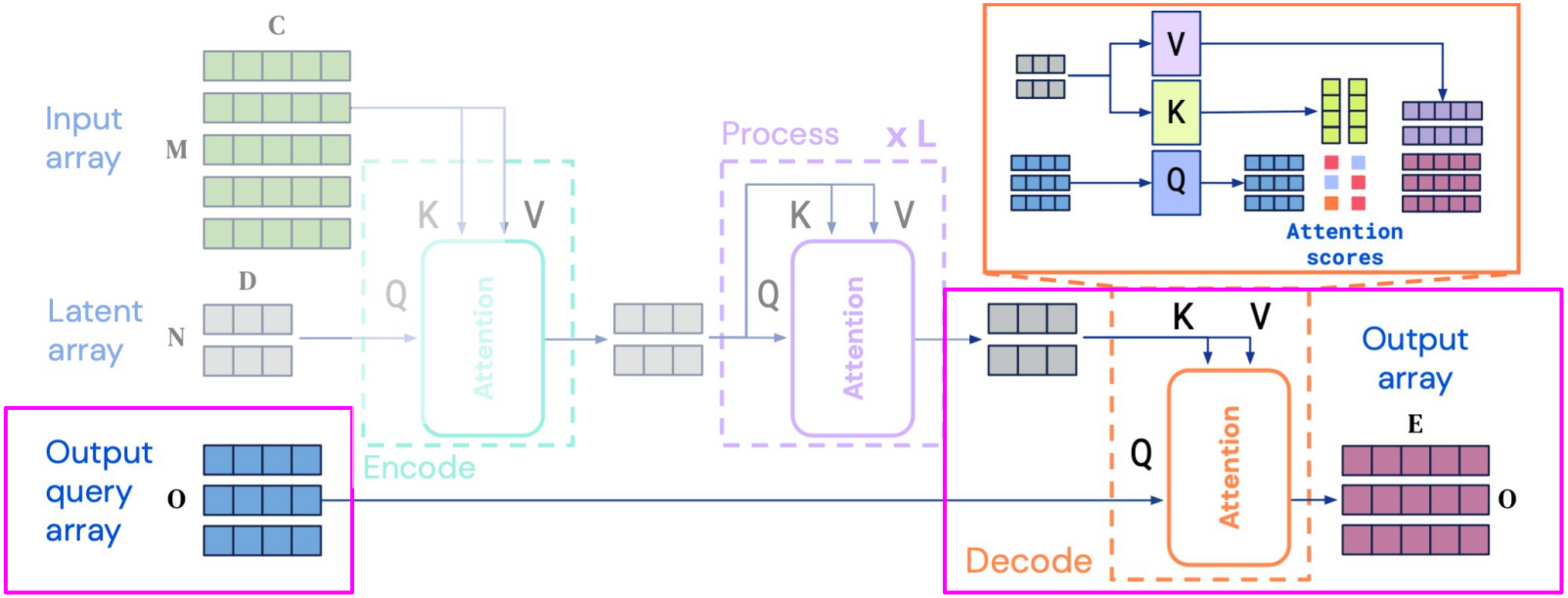
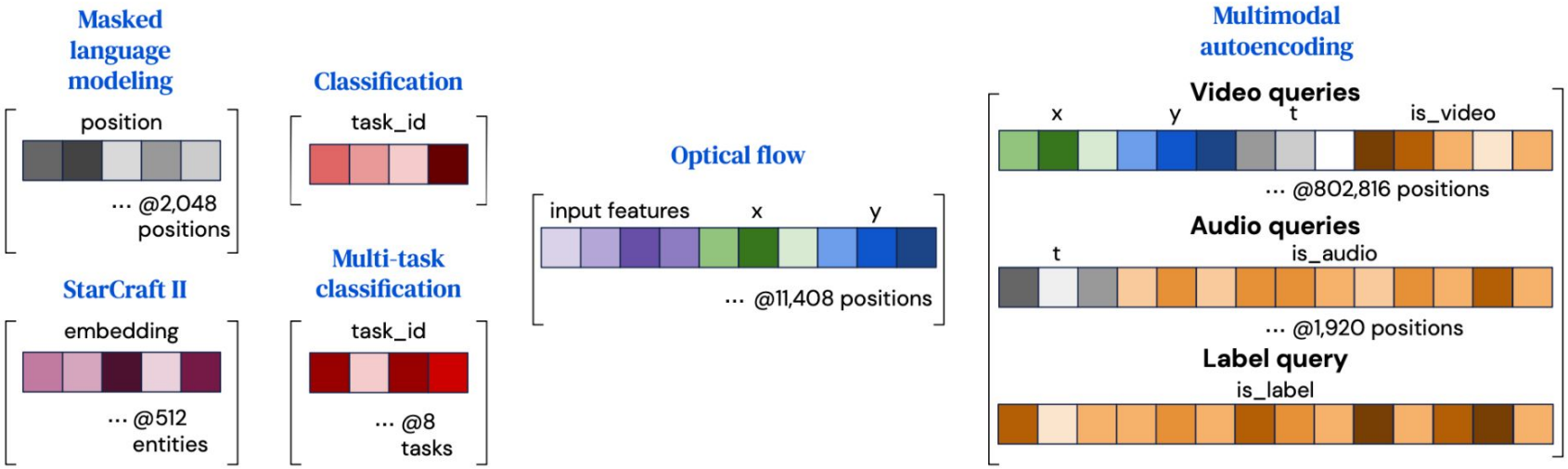


Figure 2: The Perceiver IO architecture. Perceiver IO maps arbitrary input arrays to arbitrary output arrays in a domain agnostic process. The bulk of the computation happens in a latent space whose size is typically smaller than the inputs and outputs, which makes the process computationally tractable even for very large inputs & outputs. See Fig. 5 for a more detailed look at encode, process, and decode attention.

Output Query Encodings



Performance on GLUE

Model	Tokenization	M	N	Depth	Params	FLOPs	SPS	Avg.
BERT Base (test)	SentencePiece	512	512	12	110M	109B	-	81.0
BERT Base (ours)	SentencePiece	512	512	12	110M	109B	7.3	81.1
Perceiver IO Base	SentencePiece	512	256	26	223M	119B	7.4	**81.2**
BERT (matching FLOPs)	UTF-8 bytes	2048	2048	6	20M	130B	2.9	71.5
Perceiver IO	UTF-8 bytes	2048	256	26	201M	113B	7.6	81.0
Perceiver IO++	UTF-8 bytes	2048	256	40	425M	241B	4.2	**81.8**

Overview

Convolutions – still a thing!

Transformers & Variants

Interesting concepts and applications

ICLR

Interesting Concepts & Applications

RISP - Rendering-Invariant State Predictor with **Differentiable Simulation And Rendering** for Cross-Domain Parameter Estimation

StyleAlign

Analysis And Applications Of **Aligned StyleGAN Models**

NASPY

Automated Extraction Of

Automated Machine Learning Models

RISP - Rendering-Invariant State Predictor

RISP: Rendering-Invariant State Predictor with Differentiable Simulation and Rendering for Cross-Domain Parameter Estimation

Pingchuan Ma*
MIT CSAIL
pcma@csail.mit.edu

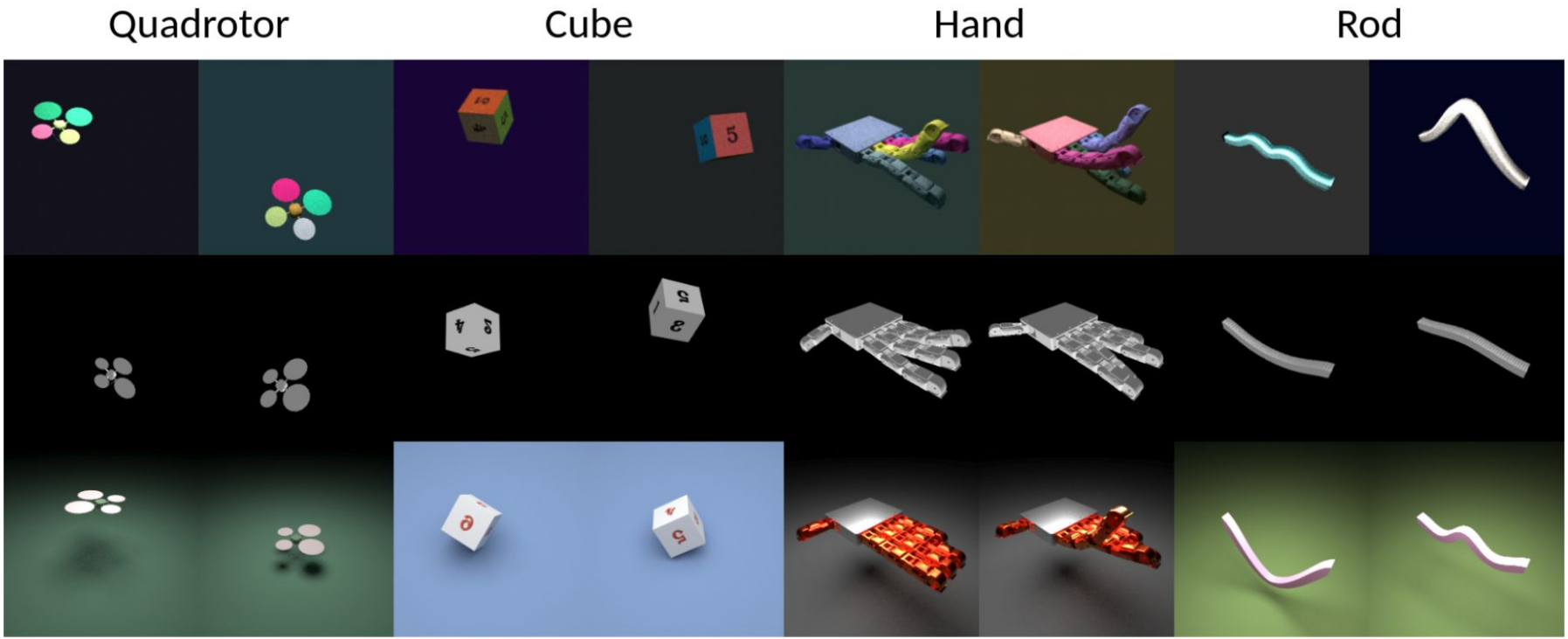
Tao Du*
MIT CSAIL
taodu@csail.mit.edu

Joshua B. Tenenbaum
MIT BCS, CBMM, CSAIL
jbt@mit.edu

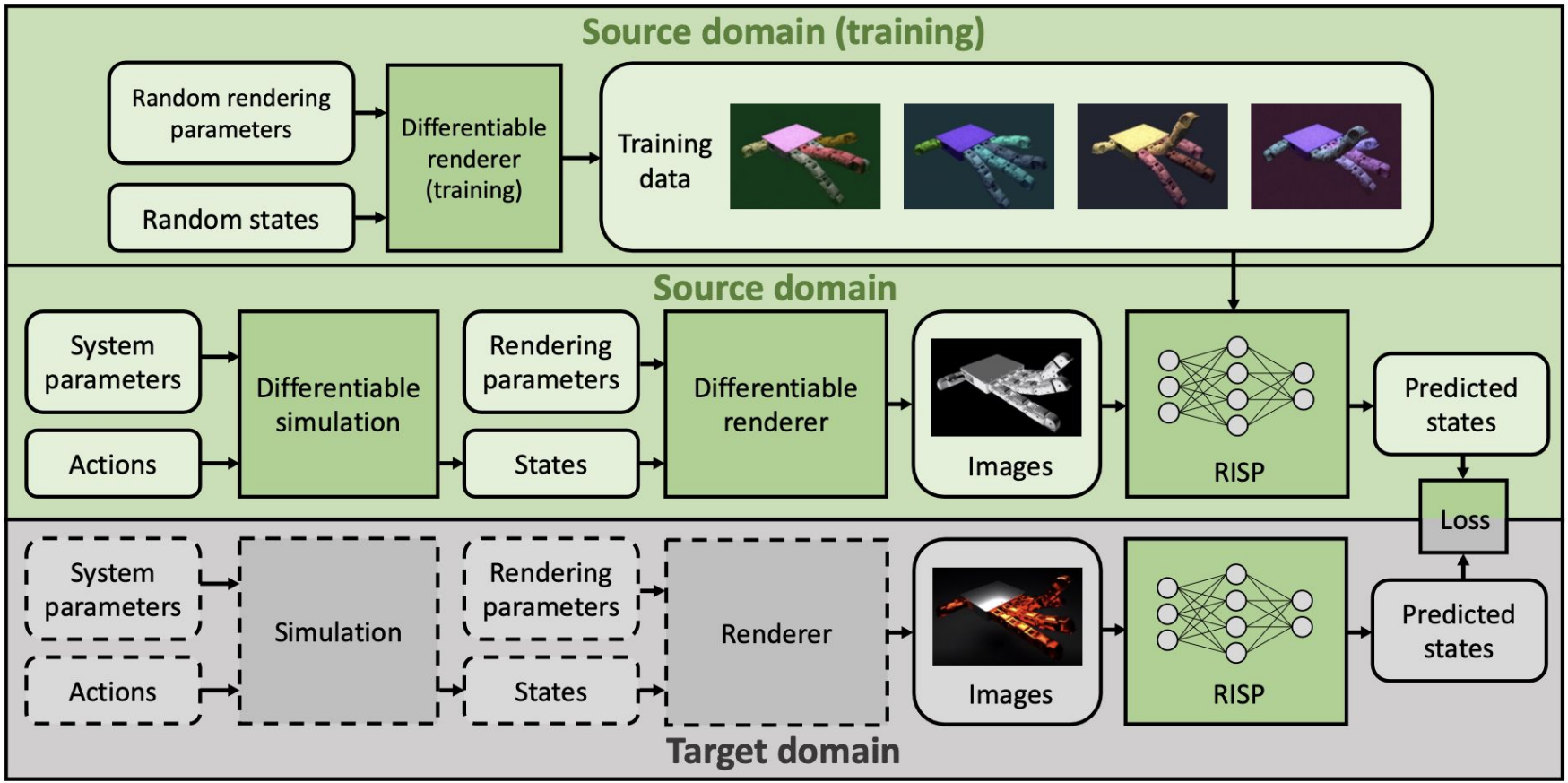
Wojciech Matusik
MIT CSAIL
wojciech@csail.mit.edu

Chuang Gan
MIT-IBM Watson AI Lab
ganchuang@csail.mit.edu

RISP - Rendering-Invariant State Predictor



RISP - Rendering-Invariant State Predictor

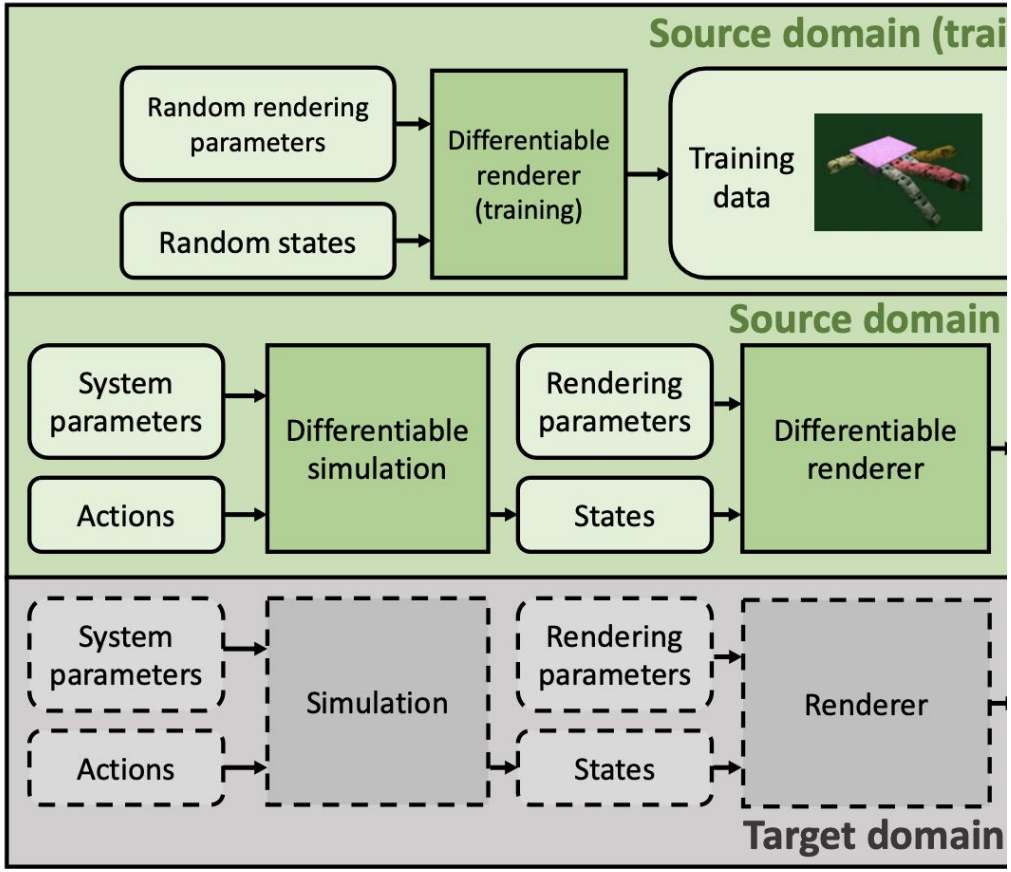


RISP - Rendering-Invariant State Predictor

Alternative approach - Adversarial Training:

- Attach a network to an intermediate representation
- Classifier shall predict e.g. rendering parameters
- Make sure the classifier performs **badly**
- This ensures there is no information about the rending in the embedding

RISP - Rendering-Invariant State Predictor



$$\mathcal{L}^{\text{error}}(\boldsymbol{\theta}, \mathcal{D}) = \sum_{(\mathbf{s}_j, \boldsymbol{\psi}_j, \mathbf{I}_j) \in \mathcal{D}} \underbrace{\|\mathbf{s}_j - \mathcal{N}_{\boldsymbol{\theta}}(\mathbf{I}_j)\|_1}_{\mathcal{L}_j^{\text{error}}}.$$

$$\frac{\partial \mathcal{N}_{\boldsymbol{\theta}}(\mathcal{R}_{\boldsymbol{\psi}}(\mathbf{s}))}{\partial \boldsymbol{\psi}} \equiv \mathbf{0}, \quad \forall \mathbf{s}, \boldsymbol{\psi},$$

RISP - Rendering-Invariant State Predictor

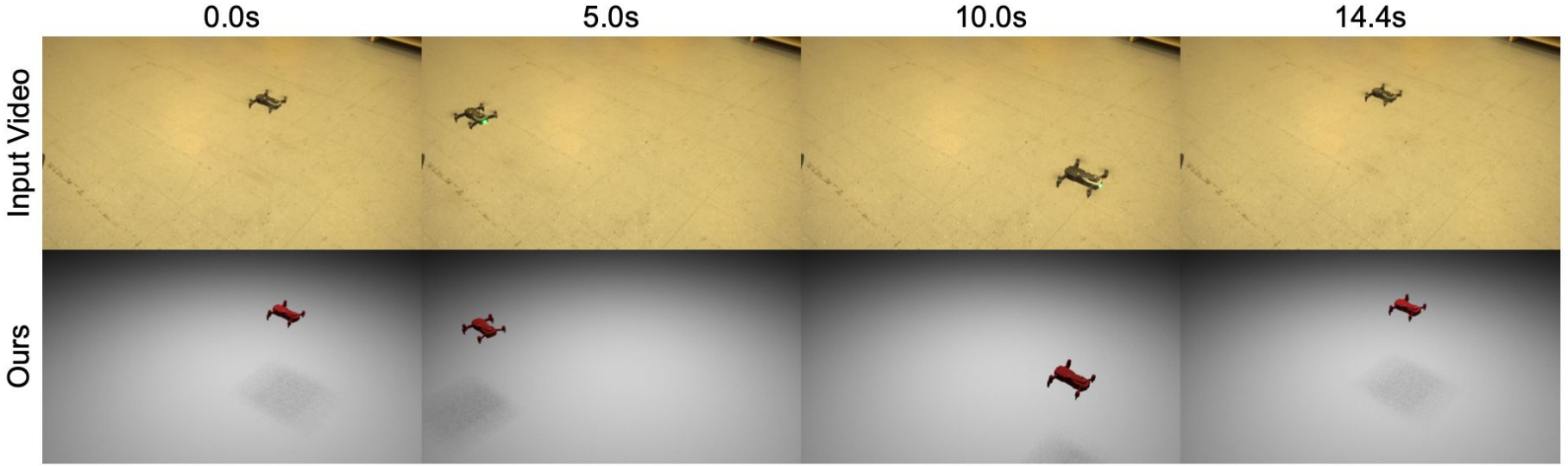


Figure 4: Imitation learning in the real-world experiment (Sec. 4.5). Given a reference video (top row), the goal is to reconstruct a sequence of actions sent to a virtual quadrotor that resembles its motion. We illustrate the motions reconstructed using our method (bottom row).

StyleAlign – Aligned StyleGAN Models

STYLEALIGN: ANALYSIS AND APPLICATIONS OF ALIGNED STYLEGAN MODELS

Zongze Wu
The Hebrew University

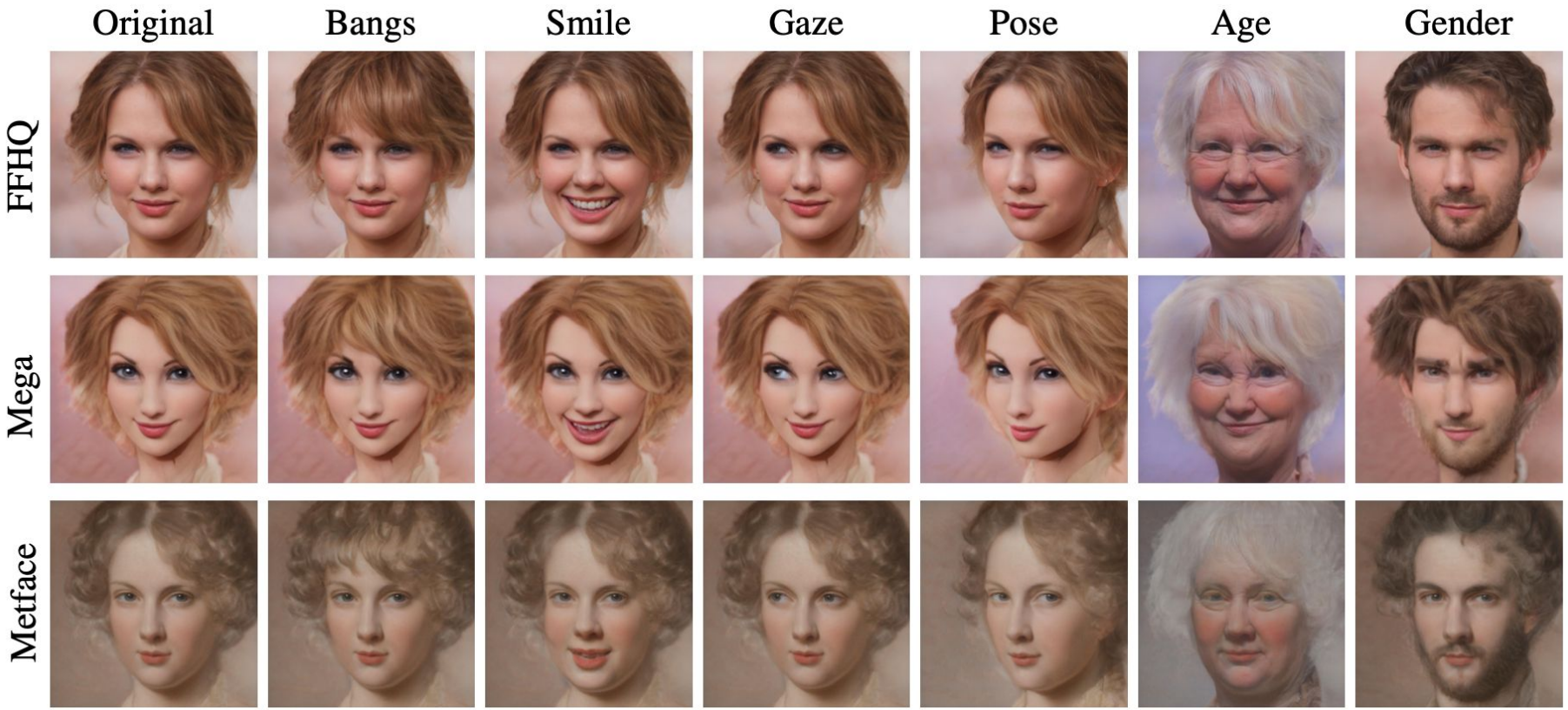
Yotam Nitzan
Tel-Aviv University

Eli Shechtman
Adobe Research

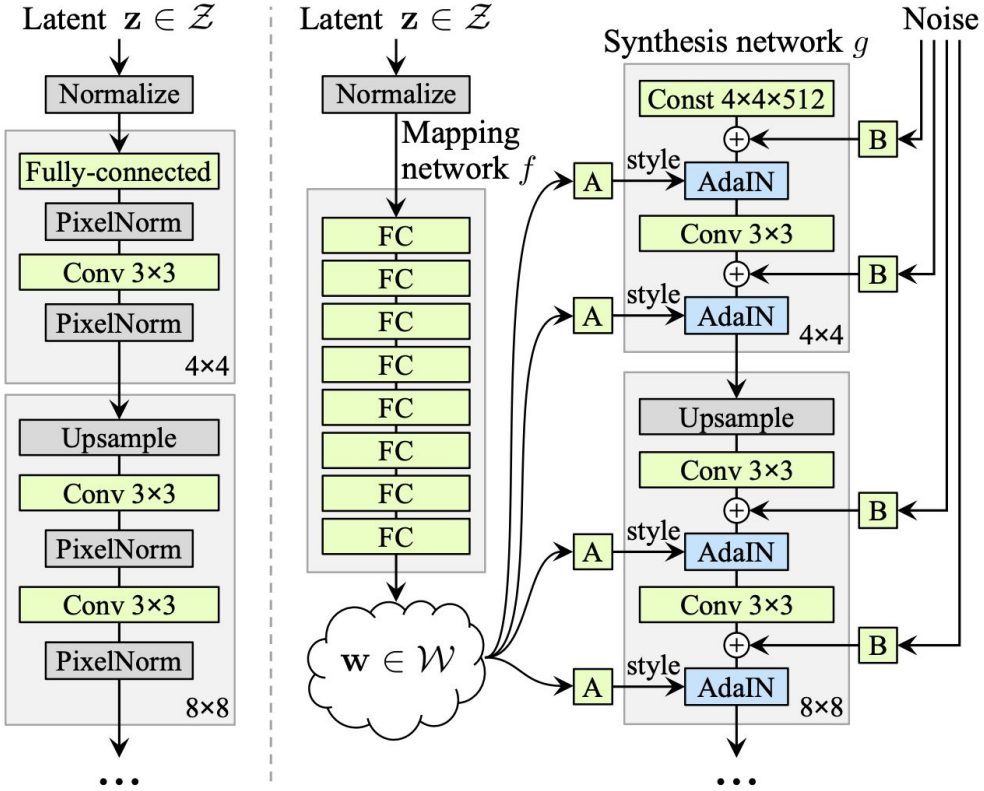
Dani Lischinski
The Hebrew University

aligned generative models. We refer to two models as aligned if they share the same architecture, and one of them (the *child*) is obtained from the other (the *parent*) via fine-tuning to another domain, a common practice in transfer learning.

StyleAlign – Aligned StyleGAN Models



StyleAlign – Aligned StyleGAN Models



(a) Traditional

(b) Style-based generator

StyleAlign – Aligned StyleGAN Models

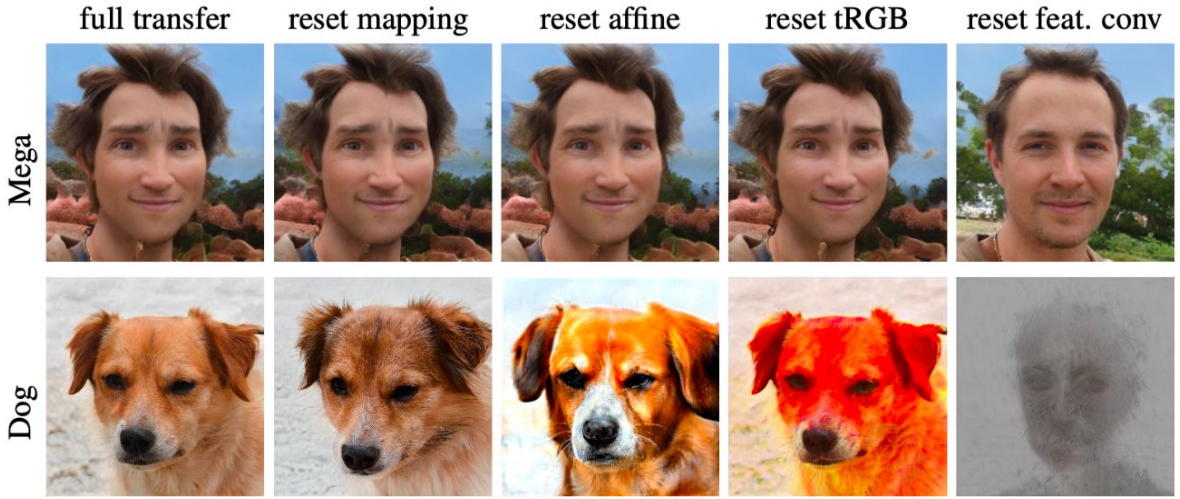
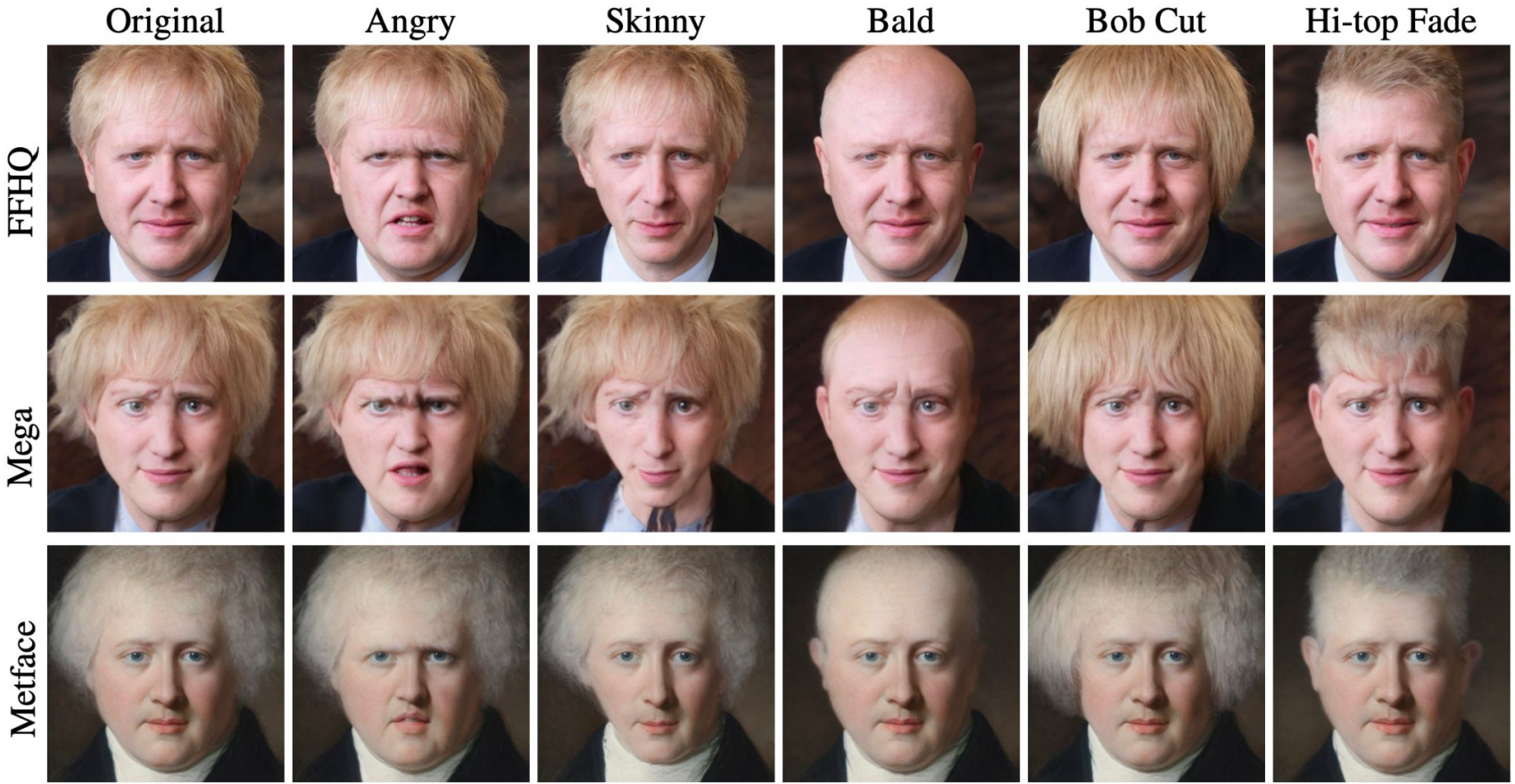


Figure 1: Effect of resetting the weights of different components in child models (Mega, Dog) to their initial values, which come from the parent model (FFHQ). Resetting the feature convolution weights causes the most drastic changes. Also see Figure 8.

StyleAlign – Aligned StyleGAN Models



NASPY – Extraction Of Machine Learning Models

NASPY: AUTOMATED EXTRACTION OF AUTOMATED MACHINE LEARNING MODELS

Xiaoxuan Lou[1], **Shangwei Guo**[2], **Jiwei Li**[3,4], **Yaoxin Wu**[1] **and Tianwei Zhang**[1]
[1]Nanyang Technological University, [2]Chongqing University, [3]Zhejiang University, [4]Shannon.AI
{xiaoxuan001, wuyaoxin, tianwei.zhang}@e.ntu.edu.sg, swguo@cqu.edu.cn, jiwei_li@shannonai.com

NASPY – Extraction Of Machine Learning Models

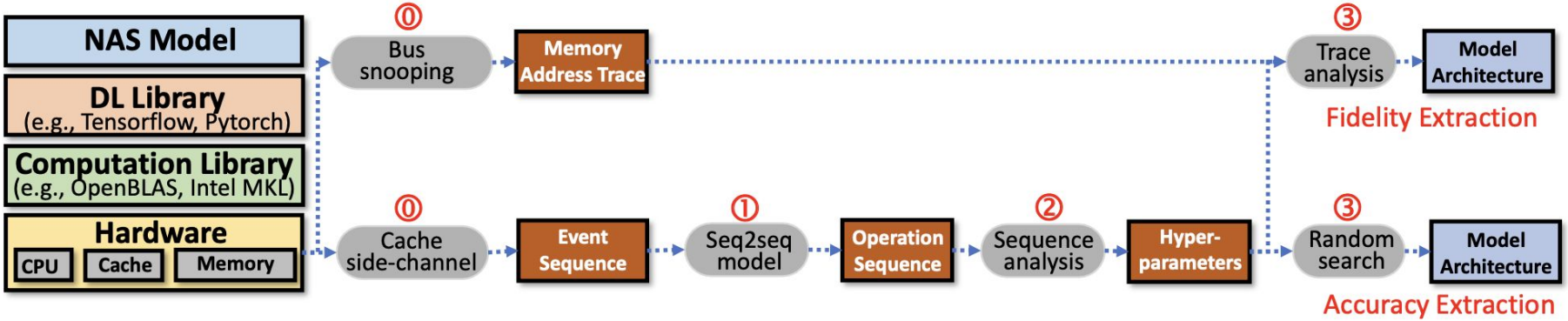


Figure 1: Workflow of our model extraction framework.

NASPY – Extraction Of Machine Learning Models

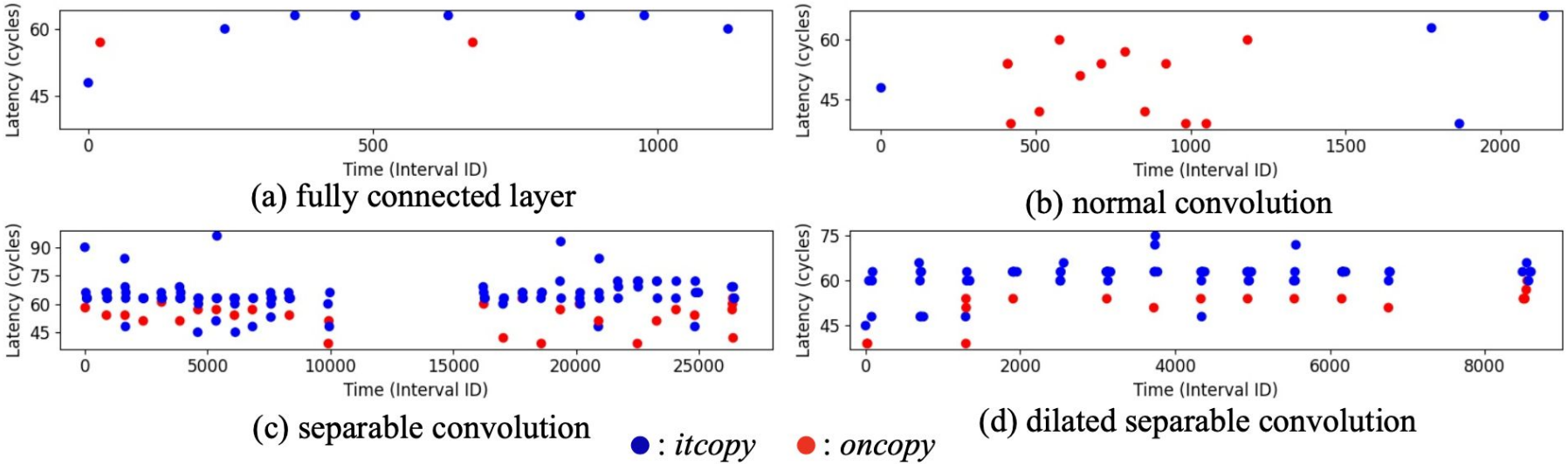


Figure 2: Event sequences of four representative operations in NAS models.

NASPY – Extraction Of Machine Learning Models

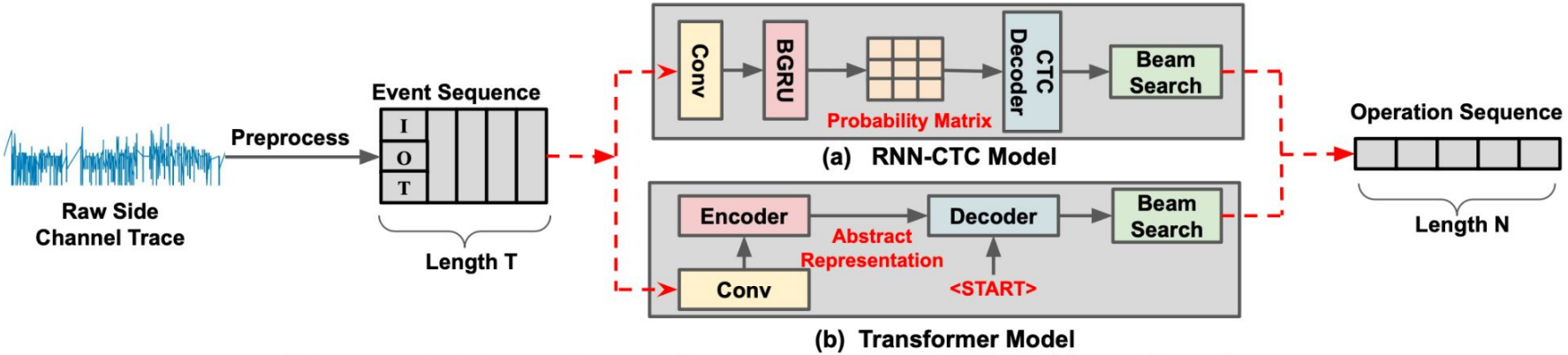


Figure 3: Procedure of operation sequence identification.

NASPY – Extraction Of Machine Learning Models

Dataset	Original Model	Random Model with Same Operations				
		#1	#2	#3	#4	#5
CIFAR 10	96.82	96.53	96.44	96.60	96.57	96.77
CIFAR 100	81.07	80.95	80.16	80.33	80.90	80.56

Table 2: Accuracy (%) of random models on two datasets.

NASPY – Extraction Of Machine Learning Models

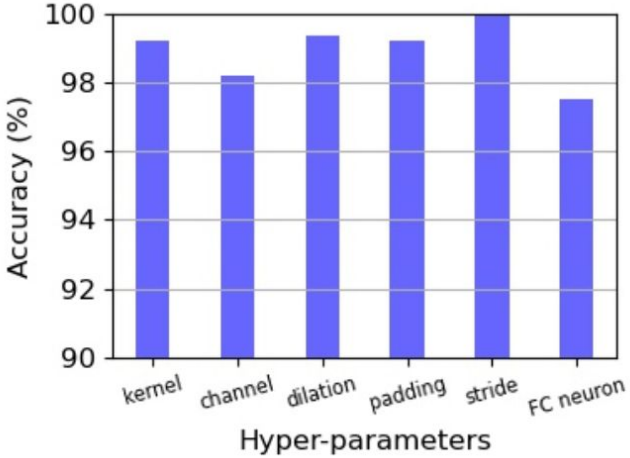


Figure 7: Recovery accuracy.

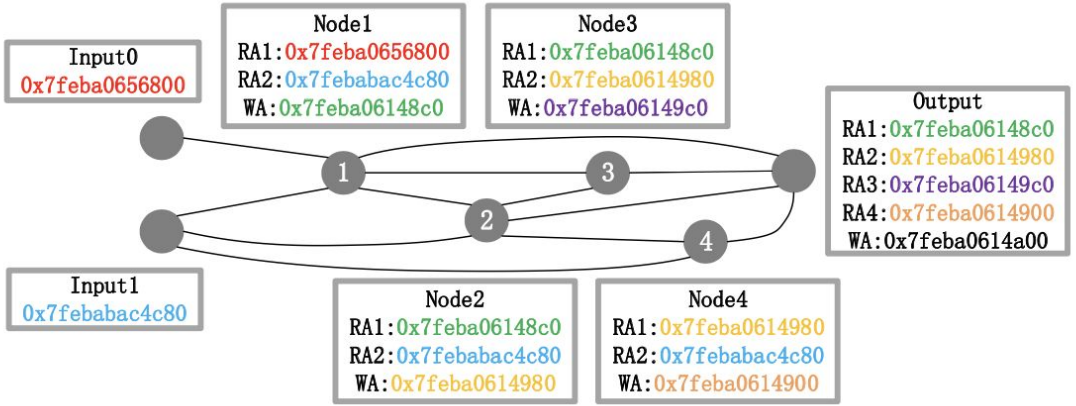


Figure 8: Memory address trace of a NAS cell.

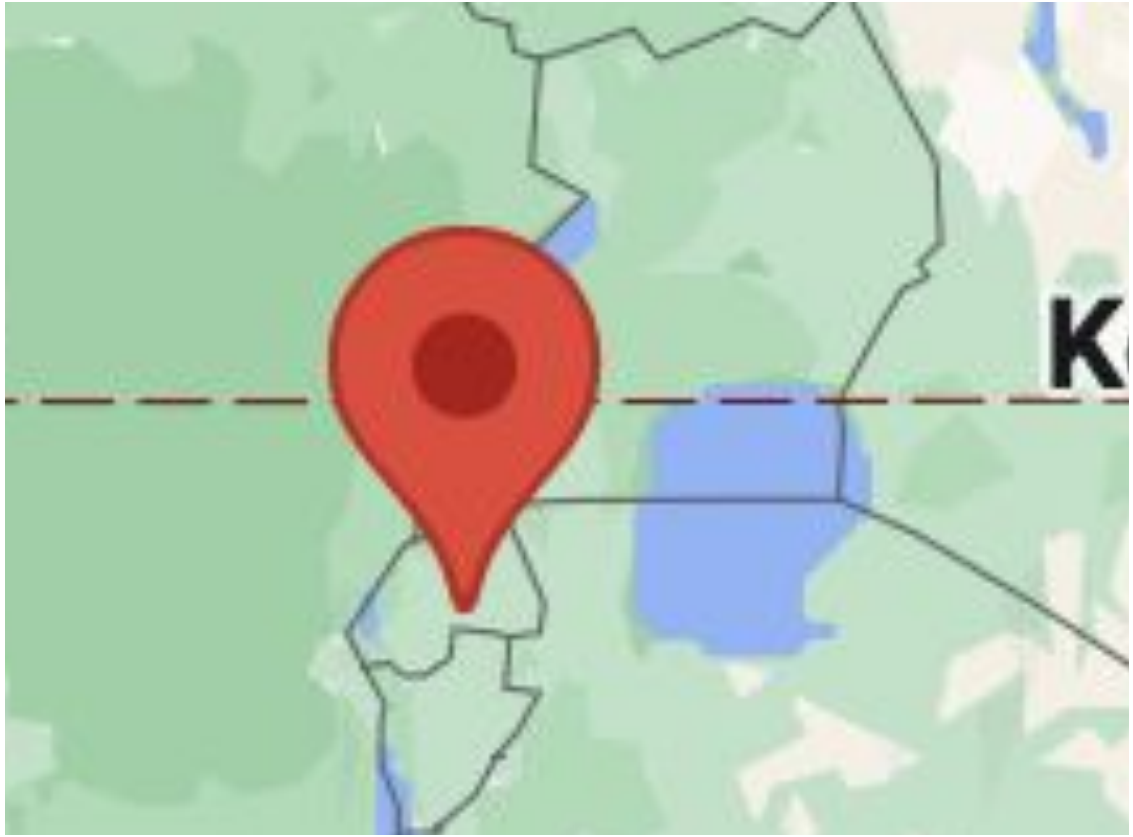
The End

Convolutions – still a Thing!

Transformers & Variants

Interesting Concepts & Applications

ICLR


K

ICLR 2023 – Kigali, Rwanda!

ICLR

ICLR 2022 – Review