Demystifying Vision Transformers:

From Theory to Industry Insights

Abhishek Aich
Researcher,
NEC Laboratories, America

NEC
NEC Laboratories America

Hello SONOMA STATE UNIVERSITY ! Some context ..

Bachelors,
Biju Patnaik University
of Technology, Odisha, India

2011-15

Masters,
National Institute of
Technology, Trichy, India

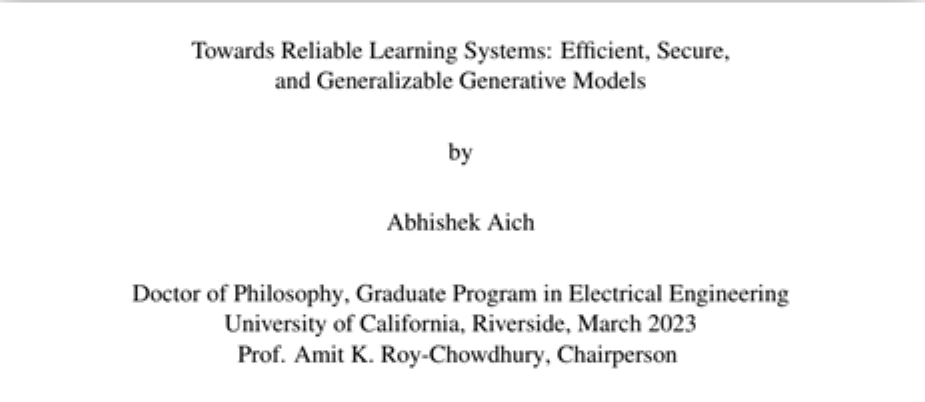
2016-18

Ph.D.,
University of California,
Riverside, USA

2018-23

Researcher,
NEC Labs, America
San Jose, USA

2023-



Towards Reliable Learning Systems: Efficient, Secure,
and Generalizable Generative Models

by

Abhishek Aich

Doctor of Philosophy, Graduate Program in Electrical Engineering
University of California, Riverside, March 2023
Prof. Amit K. Roy-Chowdhury, Chairperson

Hello SONOMA STATE UNIVERSITY! Some context ..

NEC

- Nippon Electric Company is a Japanese multinational information technology company
- HQ: Tokyo, Japan
- Est. 1899, approx. 125 years old

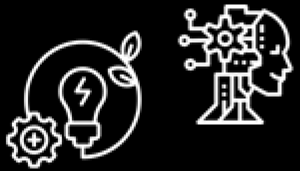
NEC
NEC Laboratories America

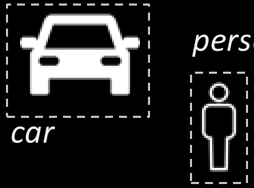
- NEC Corporation's global network of corporate research laboratories.
- HQ: Princeton, NJ
  - Our office: San Jose, CA
- Est. 1988

My Current Research Focus

Efficient Vision Transformers

Make AI Models Computationally Efficient





Detect/Segment Any Object in the Wild

Open-Vocabulary Perception

Vision-Language Models

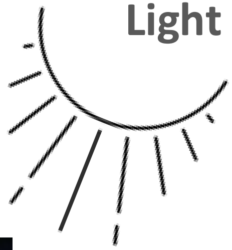
Models Integrating Vision & Language

Talk Outline

- **Zero-to-Vision Transformers (ViT)**
  - **Capturing Images**
  - **Images to Neural Networks**
  - **Neural Networks to Convolutional Neural Networks**
  - **Convolutional Neural Networks to ViT**
- **ViT and their Computational Costs**
- **From Paper to Deployment**
- **(unrelated) Pursuing Ph.D.**

From capturing an image to passing it through a ViT (from *efficiency* perspective)

Light source

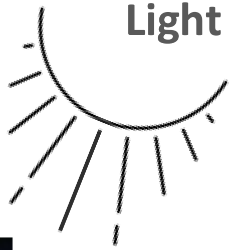


- Light is reflected from physical objects

Real world

Light capturing device

- Focuses (reflected) light on to image sensor and converts light into electrical signals.
- Each pixel records intensity and color information.



Light source

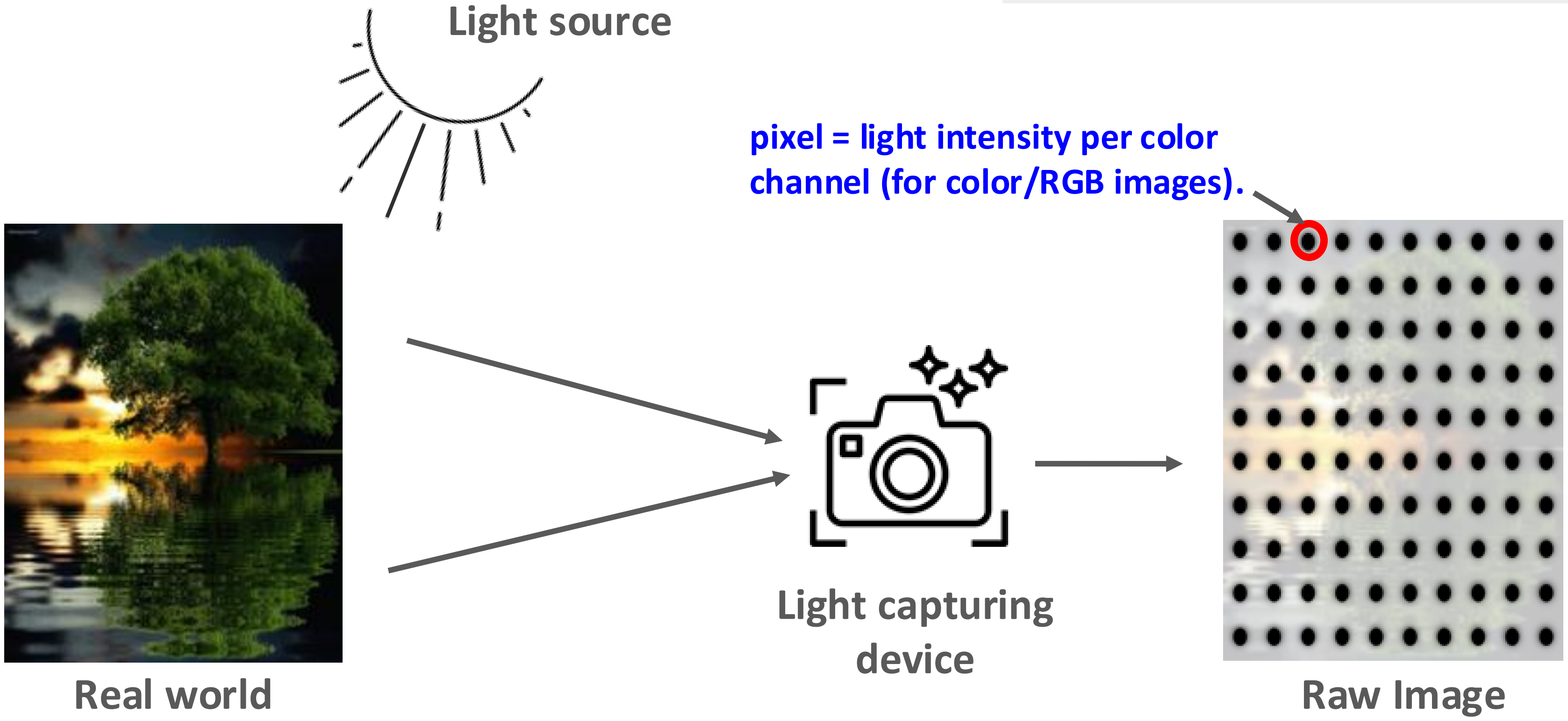
- Light is reflected from physical objects

Real world

Light capturing device

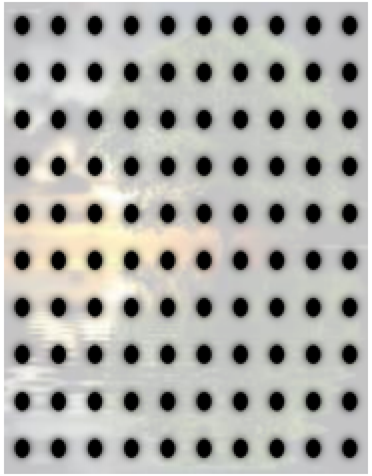
- Focuses (reflected) light on to image sensor and converts light into electrical signals.

- Each pixel records intensity and color information.


Light source
pixel = light intensity per color channel (for color/RGB images).
Light capturing device
Real world
Raw Image

- Sensors record the intensity values for
  - 3 channels [Red, Green, Blue]
  - 8-bit standard

    ∴ we get 2^8 =256 values $\Rightarrow$ [0, 255]



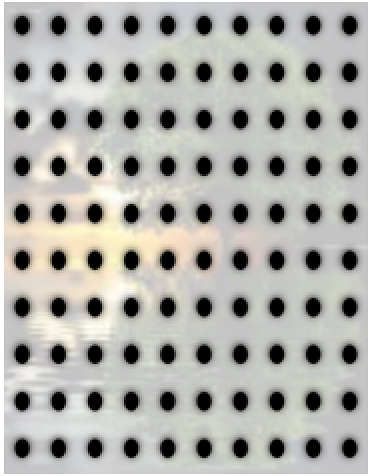
Raw Image

→ Bunch of pre-processing →

Digital Image

- **Sensors record the intensity values for**
  - **3 channels [Red, Green, Blue]**
  - **8-bit standard**

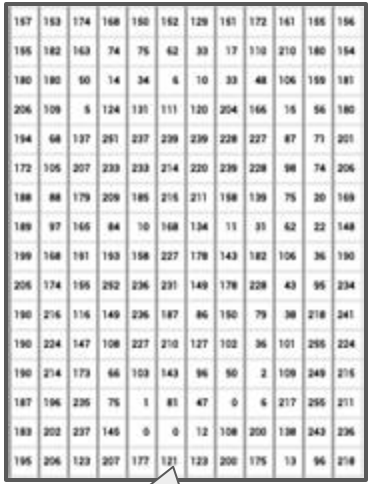
    ∴ we get 2^8 =256 values ⇒ [0, 255]



Raw Image

Bunch of pre-processing

width



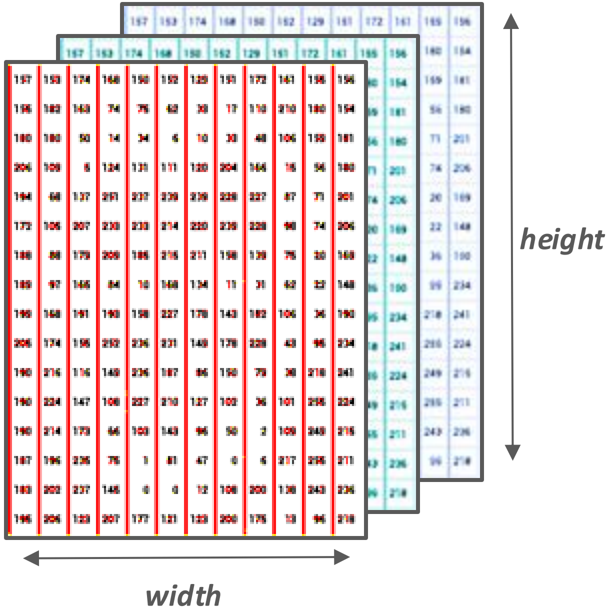
height

These values range from [0, 255].

Digital Image

Let's call it "image" for simplicity!

R ⇒ [0, 255], G ⇒ [0, 255], B ⇒ [0, 255]



Digital image = $\begin{bmatrix} R \\ G \\ B \end{bmatrix}$

Traditionally, ….

Image

- **To perform any "analysis" on this "image", we would require some multiple steps, broadly divided into three steps.**

Traditionally, ….

Image

Low level → (Image-to-Image) Image level manipulation e.g. normalization, edge detection

Mid level → (Image-to-Features) Extract features e.g. regions of homogeneous colors

High level → (Features-to-Analysis) Use features for downstream analysis e.g. object detection

[Ref.1] CSE576: Computer Vision (Spring 2006), *Basics1.pdf by Linda Shapiro, University of Washington*

Recent times, ….

Image

Low level

Mid level

High level

Image

⇩

Neural Networks

⇩

Analysis

Why? This is because they allow to learn much high-dimensional functions!

Recent times, ….

Image

- **Remember that with current available hardware (GPU/CPU):**
  - **we are memory constrained**
  - **we are time constrained**

  &

  - **we want the best performance**

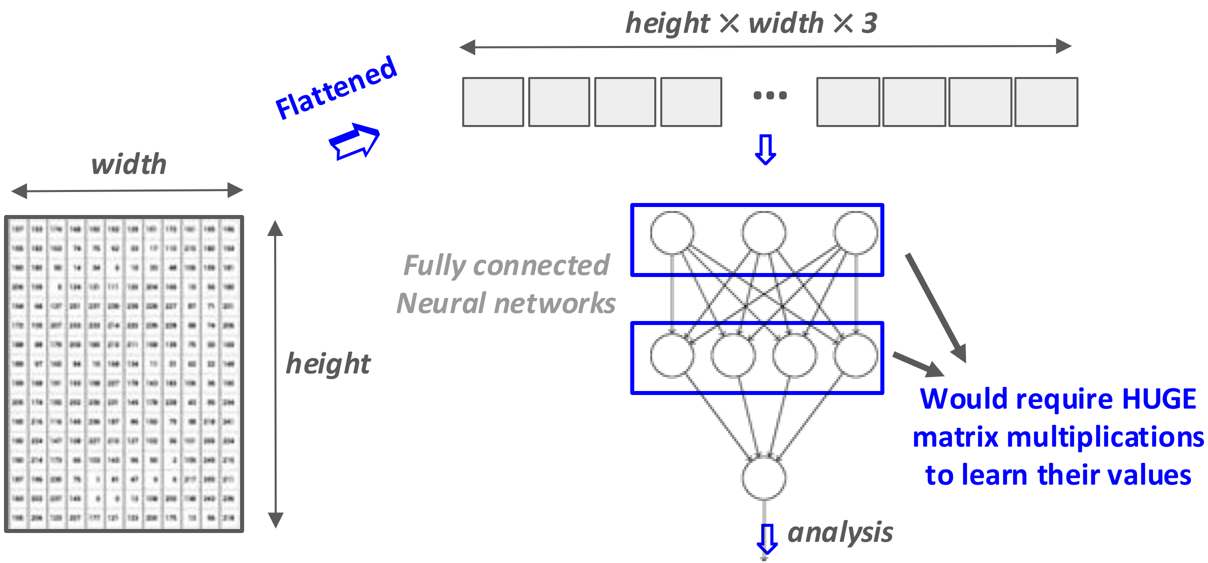
Image

⇩

Neural Networks

⇩

Analysis



NOTE:

Remember that with current available hardware (GPU/CPU):

- we are <u>memory</u> constrained
- we are also <u>time</u> constrained
- we want the <u>best</u> performance

- **Number of input parameters = Number of pixels**
  - **E.g. Image of size (224, 224, 3) would result in size of 150528 or 0.15M parameters!**
  - **So if the first layer is built with 1000 parameters, the matrix is of size [150528, 1000]**

Image
⇩

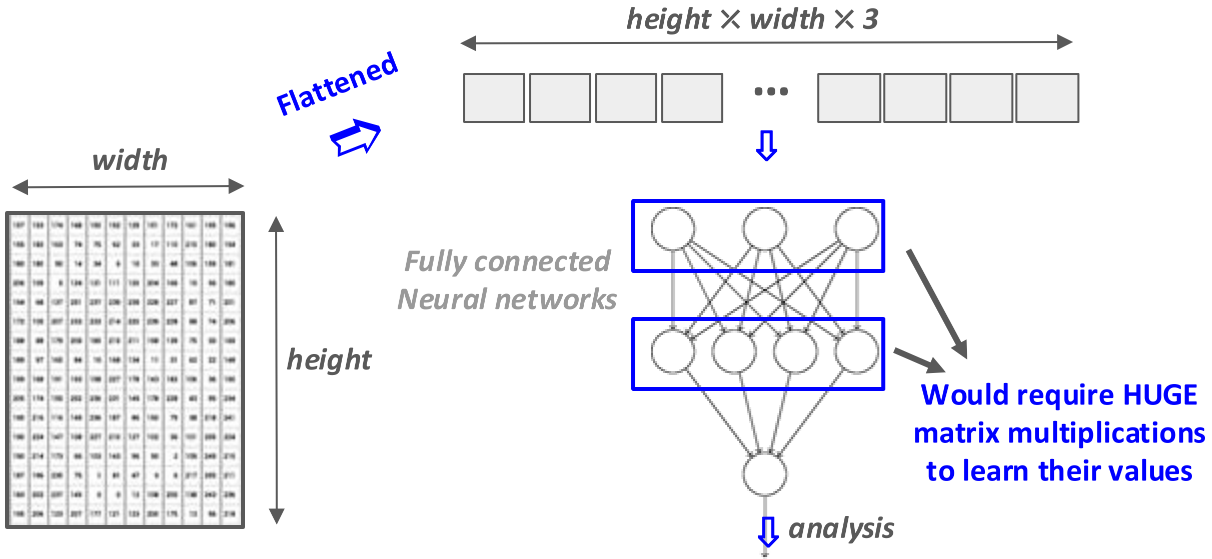
Neural Networks

⇩
Analysis

NOTE:

Remember that with current available hardware (GPU/CPU):

- we are <u>memory</u> constrained
- we are also <u>time</u> constrained
- we want the <u>best</u> performance



- Slow!
- Computationally expensive (lots of energy required!)
- Also, (extremely?) poor feature learning

Image

⇩

Neural Networks

⇩

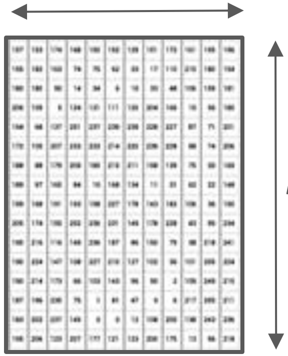
Analysis

NOTE:

Remember that with current available hardware (GPU/CPU):

- **we are memory constrained**
- **we are also time constrained**
- **we want the best performance**

width



height

- **Fully connected NN required dense interaction at every layer.**
  - **Also, no parameter sharing among the pixels.**

- **So, "convolutional" NNs were designed with the idea of sparse interaction and sharing parameters**

We will see this in the next slide.

Image

⇩

Neural Networks

⇩

Analysis

NOTE:

Remember that with current available hardware (GPU/CPU):

- we are <u>memory</u> constrained
- we are also <u>time</u> constrained
- we want the <u>best</u> performance

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

called “kernels”

Create a <u>network parameter</u> that only looks at <u>small regions</u> of the image.

size of “small” region = size of a “kernel”

Why? ⇒ “local pixels are more strongly related than distant ones”

Image

⇩

Neural Networks

⇩

Analysis

NOTE:

Remember that with current available hardware (GPU/CPU):

- **we are memory constrained**
- **we are also time constrained**
- **we want the best performance**

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

a (3, 3) convolutional kernel

Empirical value

Conv $(P, K) = \sum p_{i,j}\, k_{i,j}$

"kernel" = "filter" = a matrix that modifies input data in a structured way

Image

⇩

Neural Networks

⇩

Analysis

NOTE:

Remember that with current available hardware (GPU/CPU):

- **we are <u>memory</u> constrained**
- **we are also <u>time</u> constrained**
- **we want the <u>best</u> performance**

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

- **At multiple NN layers, these kernels are made to travel across the image.**

Why? ⇒ “patterns may appear anywhere in the image”

Image

⇩

Neural Networks

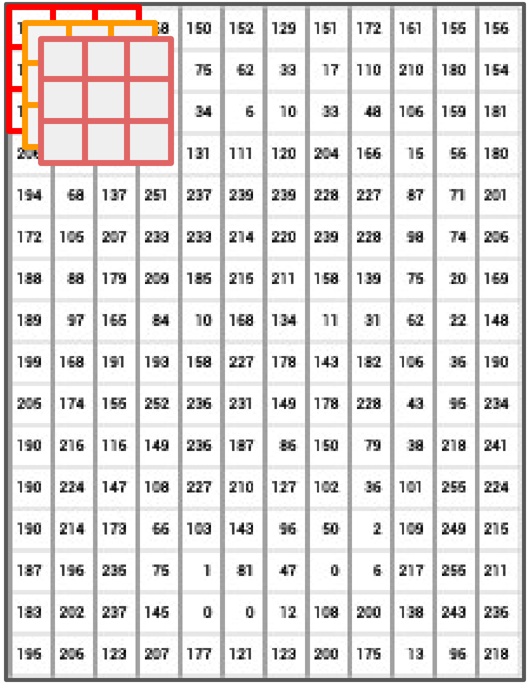
⇩

Analysis

NOTE:

Remember that with current available hardware (GPU/CPU):

- we are <u>memory</u> constrained
- we are also <u>time</u> constrained
- we want the <u>best</u> performance



- To allow the network to learn better features, we stack multiple kernels together.

But is this good enough?

Image

⇩

Neural Networks

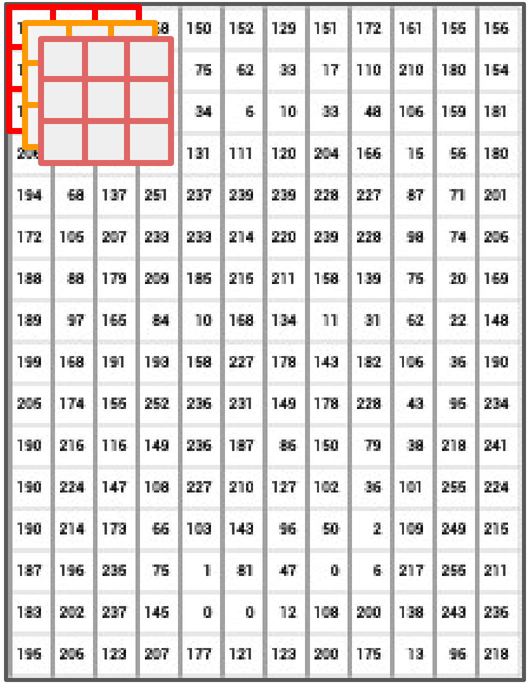
⇩

Analysis

NOTE:

Remember that with current available hardware (GPU/CPU):

- we are memory constrained
- we are also time constrained
- we want the best performance



But is this good enough?

↓

No. Kernels look at small regions and miss global information

Well, didn't we make them "small" by choice?

Image
⇩
Neural Networks
⇩
Analysis

NOTE:

Remember that with current available hardware (GPU/CPU):
- we are <u>memory</u> constrained
- we are also <u>time</u> constrained
- we want the <u>best</u> performance

Yes!, and it still works great.

BUT, with increase in available compute, performance gains are minimal.

Enter Transformers!

Image
⇩

Neural Networks

⇩
Analysis

NOTE:

Remember that with current available hardware (GPU/CPU):
- ~~we are memory constrained~~
- ~~we are also time constrained~~
- we want the best performance

Enter Transformers!

For sometime, let's keep the efficiency aside.

Quick history ..

arXiv:1706.03762v1 [cs.CL] 12 Jun 2017

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez*† University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin* illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks in an encoder-decoder configuration. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.0 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

1 Introduction

Recurrent neural networks, long short-term memory [12] and gated recurrent [7] neural networks in particular, have been firmly established as state of the art approaches in sequence modeling and transduction problems such as language modeling and machine translation [31, 2, 5]. Numerous efforts have since continued to push the boundaries of recurrent language models and encoder-decoder architectures [34, 22, 14].

- **Transformers for *language.***

- **Proposed for language translation task in 2017.**

- **Take away message: *Self-attention based (language) sequence modeling is powerful!***

[Ref.2] https://samuelalbanie.com/files/digest-slides/2023-11-vision-transformer-basics.pdf *by Samuel Albanie, University of Cambridge , UK*

Quick history ..

2010.11929v1 [cs.CV] 22 Oct 2020

AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE

Alexey Dosovitskiy*,†, Lucas Beyer*, Alexander Kolesnikov*, Dirk Weissenborn*, Xiaohua Zhai*, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby*,†
*equal technical contribution, †equal advising
Google Research, Brain Team
{adosovitskiy, neilhoulsby}@google.com

ABSTRACT

While the Transformer architecture has become the de-facto standard for natural language processing tasks, its applications to computer vision remain limited. In vision, attention is either applied in conjunction with convolutional networks, or used to replace certain components of convolutional networks while keeping their overall structure in place. We show that this reliance on CNNs is not necessary and a pure transformer applied directly to sequences of image patches can perform very well on image classification tasks. When pre-trained on large amounts of data and transferred to multiple mid-sized or small image recognition benchmarks (ImageNet, CIFAR-100, VTAB, etc.), Vision Transformer (ViT) attains excellent results compared to state-of-the-art convolutional networks while requiring substantially fewer computational resources to train.[1]

1 INTRODUCTION

Self-attention-based architectures, in particular Transformers (Vaswani et al., 2017), have become the model of choice in natural language processing (NLP). The dominant approach is to pre-train on a large text corpus and then fine-tune on a smaller task-specific dataset (Devlin et al., 2019). Thanks to Transformers' computational efficiency and scalability, it has become possible to train models of unprecedented size, with over 100B parameters. With the models and datasets growing, there is still no sign of saturating performance.

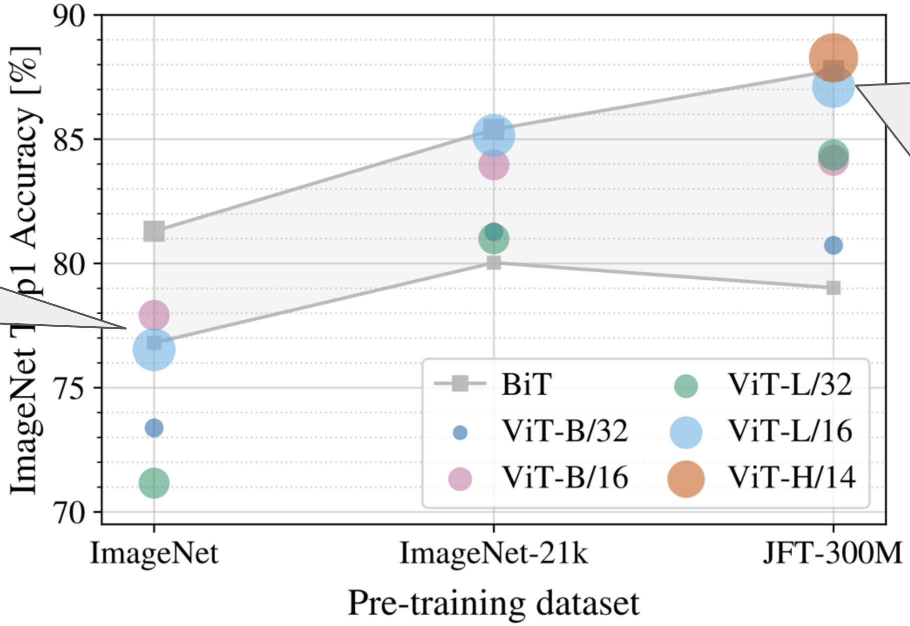
- *Vision* **Transformers (ViT)**
- **Adapted for image/vision task in 2020.**
- **Take away message: *Self-attention based sequence modeling is powerful for images as well!***

[Ref.2] https://samuelalbanie.com/files/digest-slides/2023-11-vision-transformer-basics.pdf *by Samuel Albanie, University of Cambridge , UK*

Why are they everywhere?

- Vision Transformers show better results when the dataset scales up.

ViT models perform worse than CNNs (BiT) when pre-trained on small datasets.



Large ViT models perform much better when pre-trained on larger datasets. Not the case in smaller datasets.

[Ref.2] https://samuelalbanie.com/files/digest-slides/2023-11-vision-transformer-basics.pdf *by Samuel Albanie, University of Cambridge , UK*

Why are they everywhere?

- Vision Transformers show better results when the dataset scales up.
- And less inductive bias!

Built-in assumptions that guide learning from limited data

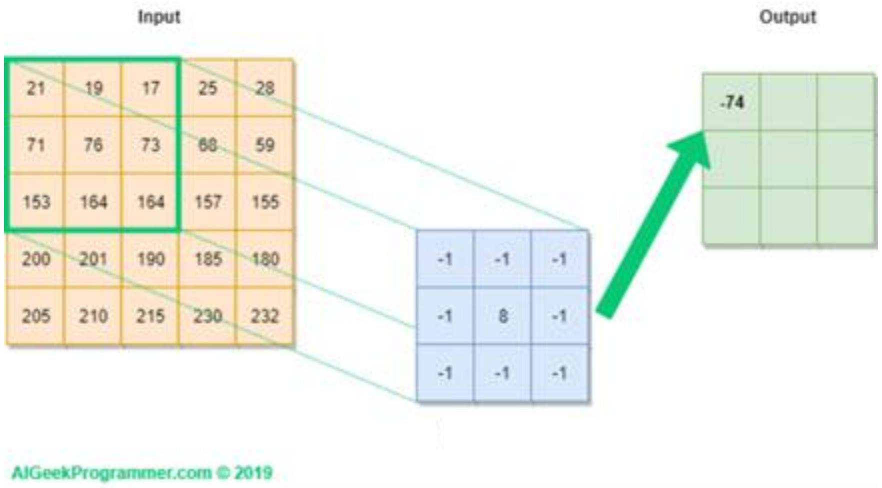
More **Inductive Bias** *Less*

Fully Connected NN ConvNN ViTs

[Ref.2] https://samuelalbanie.com/files/digest-slides/2023-11-vision-transformer-basics.pdf *by Samuel Albanie, University of Cambridge , UK*

Self-attention?

At any single layer of CNNs ...



- Each output location for next layer is computed by convolving the kernel over a local patch of pixels.

↓

- Neighborhood patches do not provide any context.

[Ref.3] Image source: https://commons.wikimedia.org/wiki/File:CNN-filter-animation-1.gif

Self-attention?

At any single layer of Transformers …

- Each patch directly participates or attends in computation of all other output patches in the same layer.

↓

Let’s see how.

Image
⇩
Neural Networks
⇩
Analysis

NOTE:

Remember that with current available hardware (GPU/CPU):
- ~~we are memory constrained~~
- ~~we are also time constrained~~
- we want the best performance

Vision Transformers

- "Transformer" ∵ 

∵ They *transform* a set of vectors in some representation space into a corresponding set of vectors in some new space, having the same dimensionality.

Image

⇩

Neural Networks

⇩

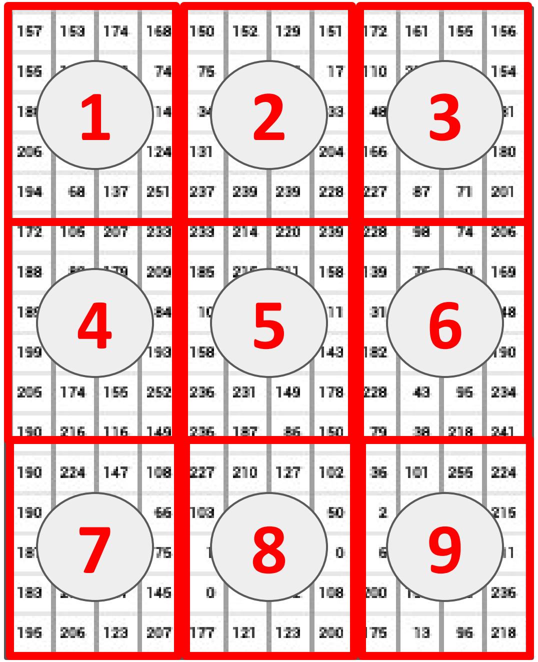
Analysis

NOTE:

Remember that with current available hardware (GPU/CPU):

- we are <u>memory</u> constrained
- we are also <u>time</u> constrained
- we want the <u>best</u> performance

Vision Transformers



Patches = tokens

- ***Step 1:***
  Divide the image into tokens.

  Why? Because Transformers process sequences.

Image

⇩

Neural Networks

⇩

Analysis

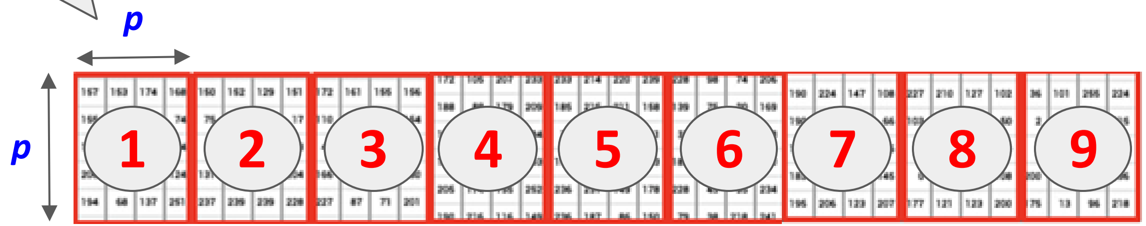
NOTE:

Remember that with current available hardware (GPU/CPU):

- ~~we are memory constrained~~
- ~~we are also time constrained~~
- we want the best performance

Vision Transformers

size = (p, p, 3)



Flatten each token to $3p^2$ ⇩

Linear layer to reduce dimensionality to D

1 2 3 4 5 6 7 8 9

Each token is now of length $D < p^2$

Image

⇩

Neural Networks

⇩

Analysis

NOTE:

Remember that with current available hardware (GPU/CPU):

- ~~we are memory constrained~~
- ~~we are also time constrained~~
- we want the best performance

Vision Transformers

1 2 3 4 5 6 7 8 9

We can see the position, transformers cannot.

- ***Step 2:***
  Add positional encodings to the patches.

  Why? Because we need to point to the Transformer the position of the patch in the image

Image

⇩

Neural Networks

⇩

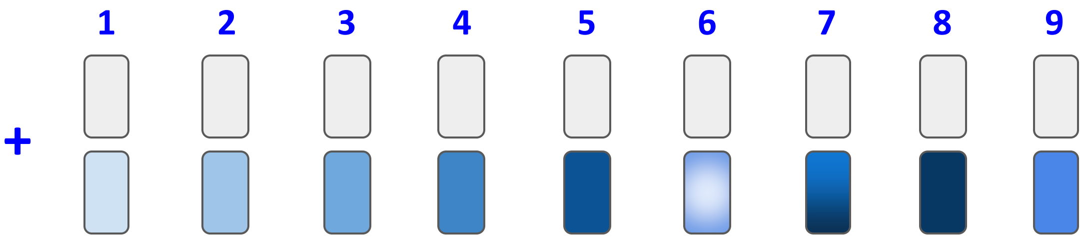
Analysis

NOTE:

Remember that with current available hardware (GPU/CPU):

- ~~we are memory constrained~~
- ~~we are also time constrained~~
- we want the best performance

Vision Transformers



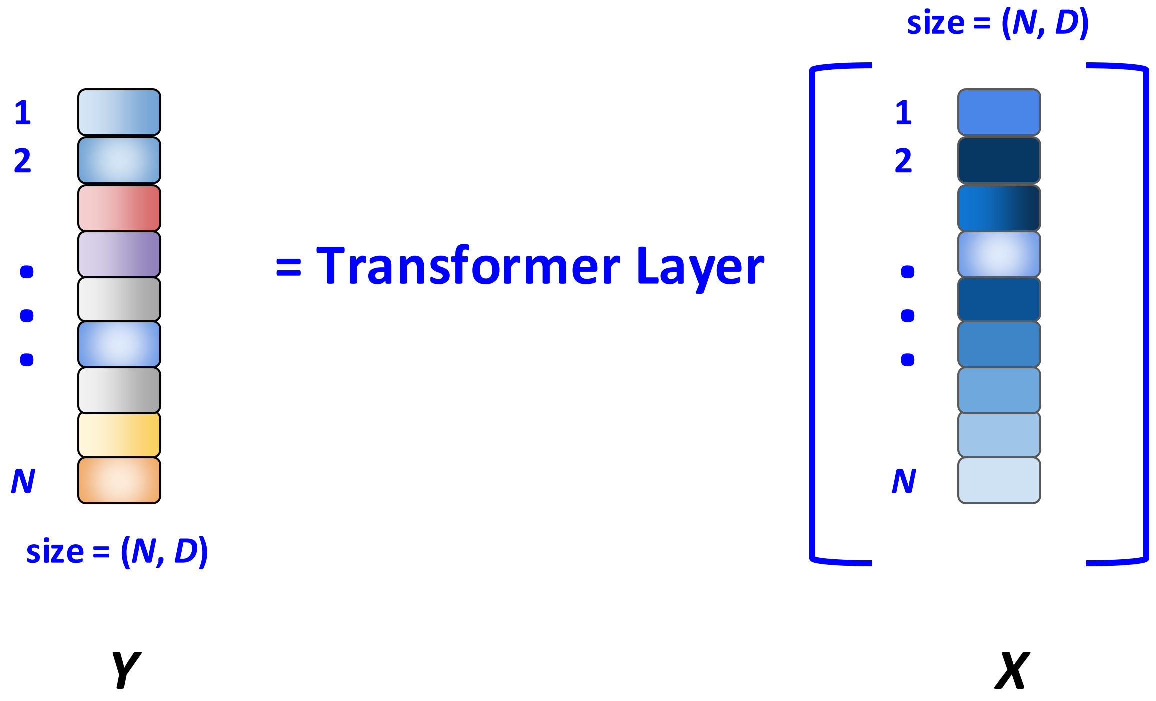
⇩

L Transformer Layers

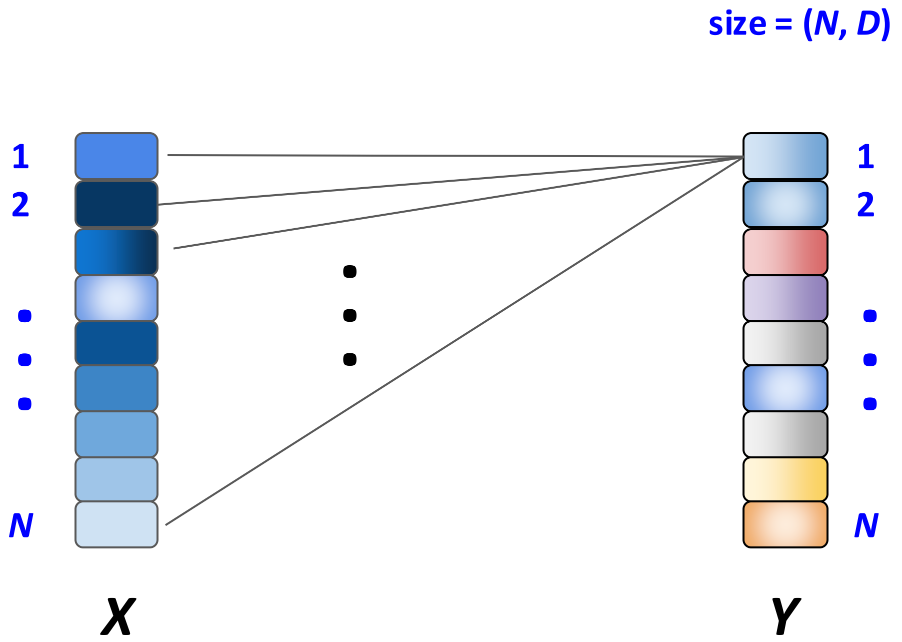
⇩

Analysis

Vision Transformers



Vision Transformers



$$y_n = \sum a_{nm} x_m$$

$0 \le a_{nm} \le 1$, $\sum a_{nm} = 1$

Make sure:

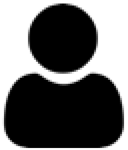
- **contributions of *certains* x_n to *certain* y_n would be higher (and lowers for others) and**
- **not cancel out each other**

[Ref.5] Deep Learning: Foundations and Concepts: Christopher M. Bishop, Hugh Bishop

Vision Transformers

Dairy Aisle

“A” “B” . . . “C”



Customer

“C *milk*” ?

We computed this similarity.

Query	Key	Value →	Output
“C *milk*” ?	“A”, “B”, “C”	“C”	

- We can think of the customer ‘attending’ to the particular brand (value) of milk (output) whose key most closely matches their query.

[Ref.5] Deep Learning: Foundations and Concepts: Christopher M. Bishop, Hugh Bishop

Vision Transformers

Query | **Key** | **Value** → **Output**

x_n | x_m | $\sum_m a_{nm} x_m$ | y_n

"self"-attention

$$y_n = \sum_m a_{nm} x_m$$

$$0 \le a_{nm} \le 1, \ \sum_m a_{nm} = 1$$

$$a_{nm} = x^T{}_n x_m$$

Computing element wise similarity

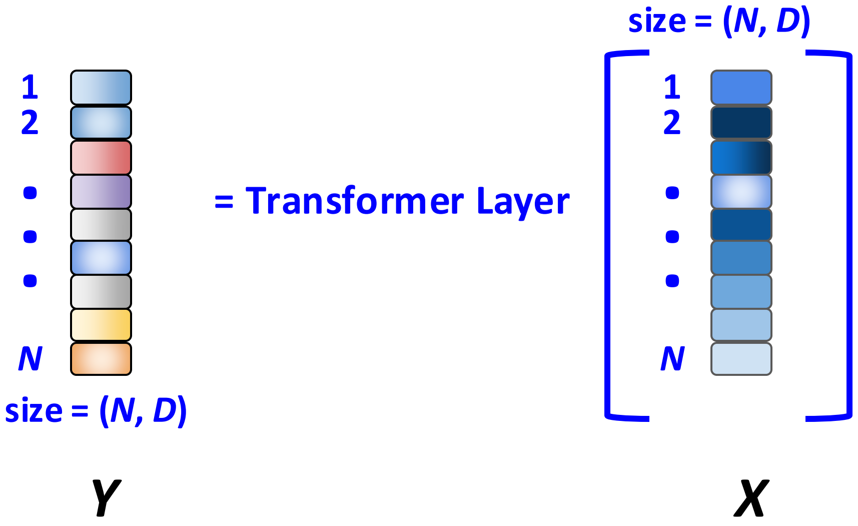
with constraints →

$$a_{nm} = exp(x^T{}_n x_m) / \sum_m exp(x^T{}_n x_m)$$

Softmax operation!

[Ref.5] Deep Learning: Foundations and Concepts: Christopher M. Bishop, Hugh Bishop

Vision Transformers



$$y_n = \sum_m a_{nm} x_m$$

$$a_{nm} = \exp(x^T{}_n x_m) / \sum_m \exp(x^T{}_n x_m)$$

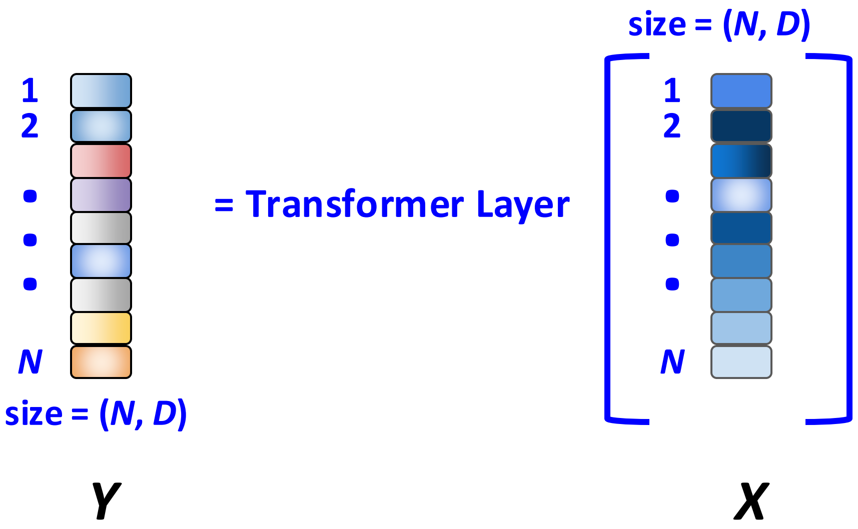
⇩**Matrix form**

$$Y = \mathrm{Softmax}(XX^T)X$$

No learnable/trainable parameter

[Ref.5] Deep Learning: Foundations and Concepts: Christopher M. Bishop, Hugh Bishop

Vision Transformers



$$Y = \text{Softmax}(XX^{\mathrm{T}})X$$

$$\Downarrow$$

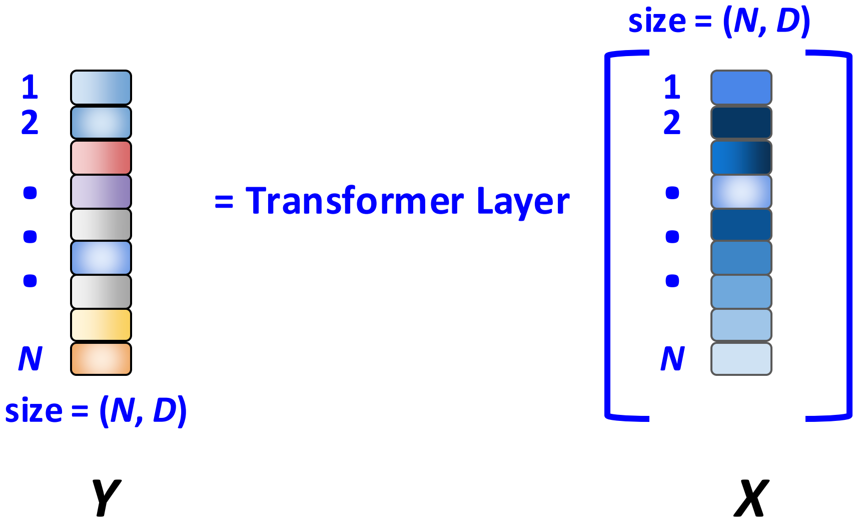
$$X \cong XU$$

Allow a learnable parameter U

$$Y = \text{Softmax}(XUU^{\mathrm{T}}X^{\mathrm{T}})XU$$

[Ref.5] Deep Learning: Foundations and Concepts: Christopher M. Bishop, Hugh Bishop

Vision Transformers



$$Y = \mathrm{Softmax}(XUU^{\mathrm{T}}X^{\mathrm{T}})XU$$

$$\Downarrow$$

$$Q = XU^{(\mathrm{q})}$$

$$K = XU^{(\mathrm{k})}$$

$$V = XU^{(\mathrm{v})}$$

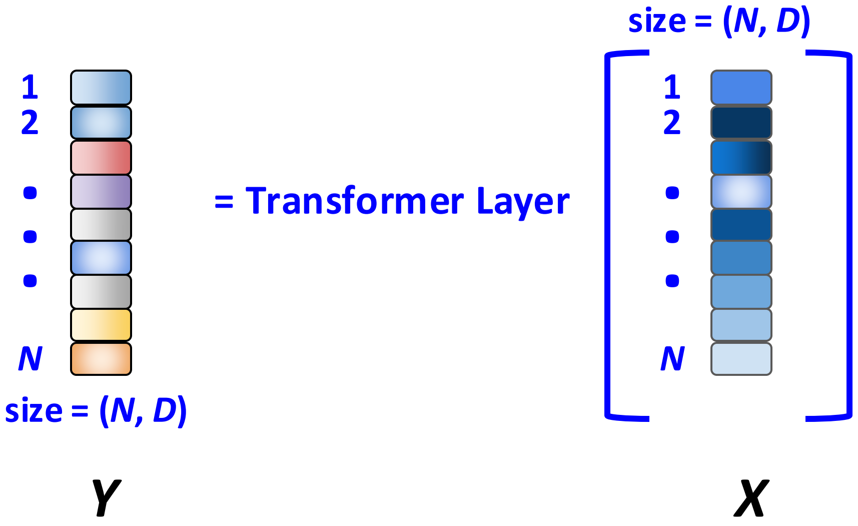
Allow independent learning

$$\Downarrow$$

$$Y = \mathrm{Softmax}(QK^{\mathrm{T}})V$$

[Ref.5] Deep Learning: Foundations and Concepts: Christopher M. Bishop, Hugh Bishop

Vision Transformers



$$Y = \text{Softmax}(XUU^{T}X^{T})XU$$

$$\Downarrow$$

$$Q = XU^{(q)}$$

$$K = XU^{(k)}$$

Allow independent learning

$$V = XU^{(v)}$$

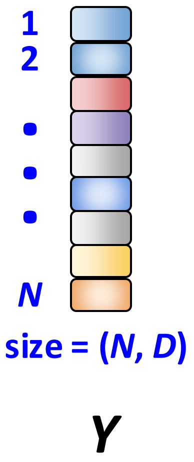
$$\Downarrow$$

$$Y = \text{Softmax}(QK^{T})V$$

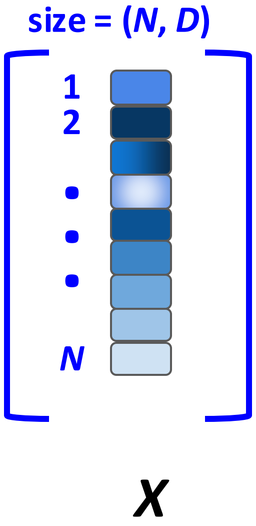
This is self-attention (with some scaling to stabilize the training)

[Ref.5] Deep Learning: Foundations and Concepts: Christopher M. Bishop, Hugh Bishop

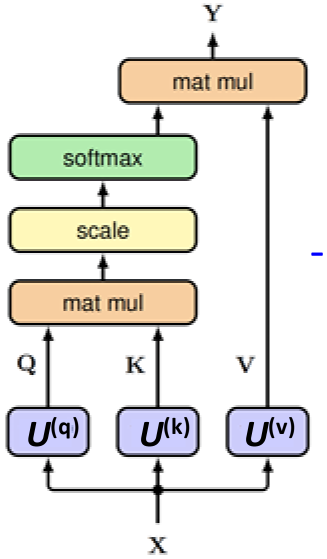
Vision Transformers



= Transformer Layer



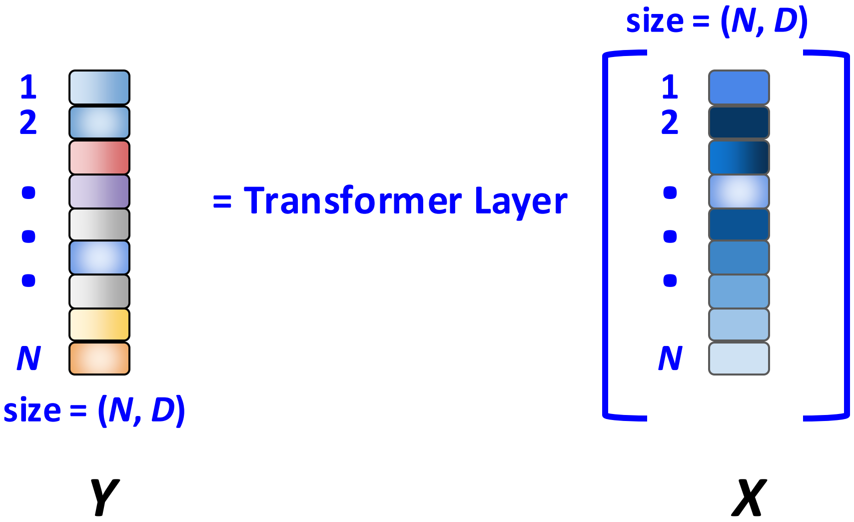
$$Y = \text{Softmax}(QK^{T})V$$



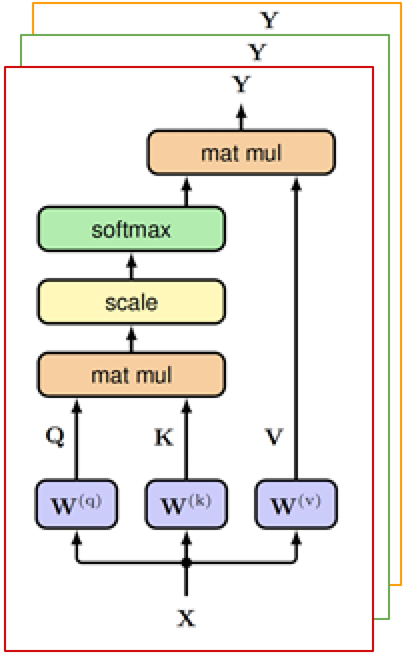
- This is single-head self-attention.

[Ref.5] Deep Learning: Foundations and Concepts: Christopher M. Bishop, Hugh Bishop

Vision Transformers



$$Y = \text{Softmax}(QK^T)V$$

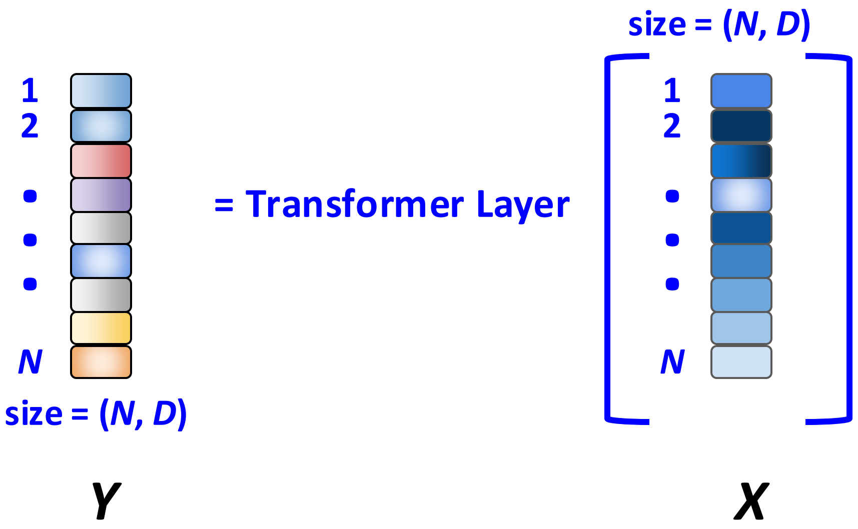


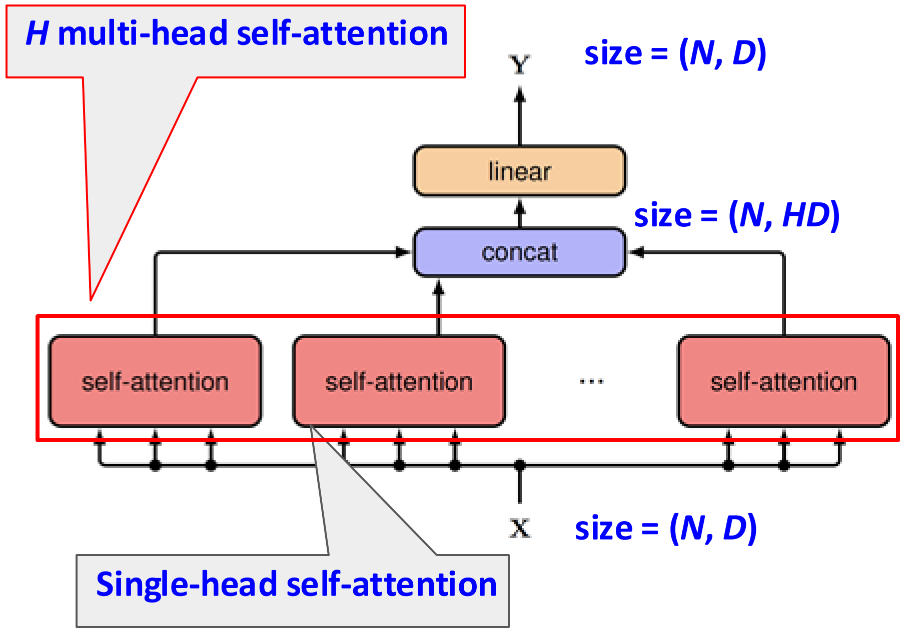
- This is multi-head self-attention.

Remember we did something similar for CNNs?

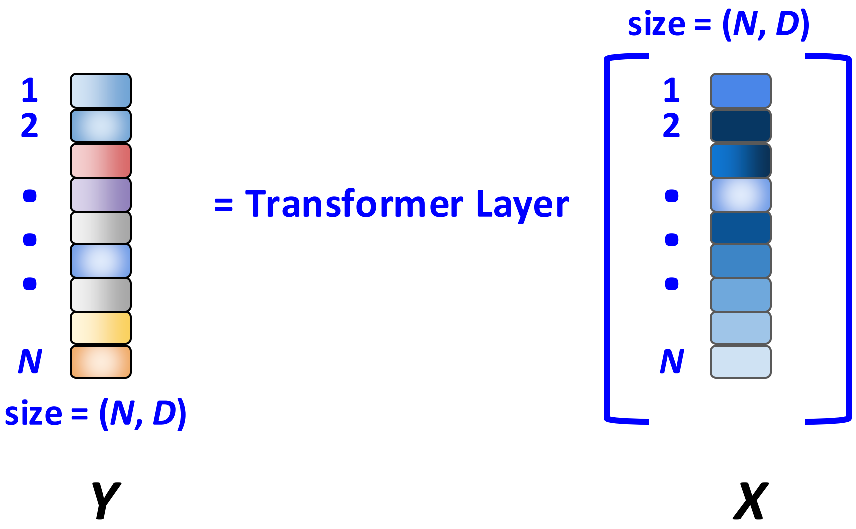
[Ref.5] Deep Learning: Foundations and Concepts: Christopher M. Bishop, Hugh Bishop

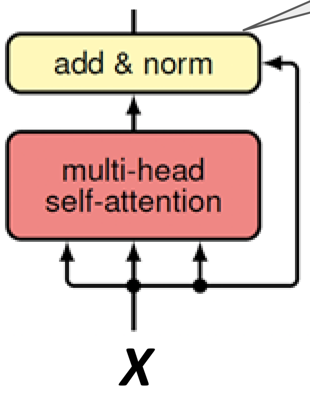
Vision Transformers





Vision Transformers



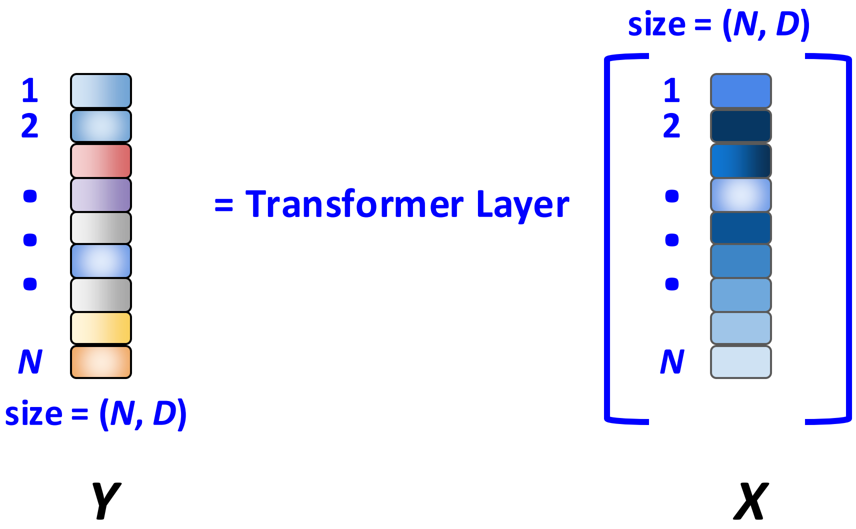


Normalize the output to stabilize training. Standard practice!

Add "residual" connection to stabilize training. The RESNET paper!

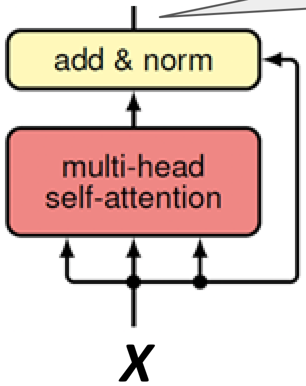
[Ref.5] Deep Learning: Foundations and Concepts: Christopher M. Bishop, Hugh Bishop

Vision Transformers

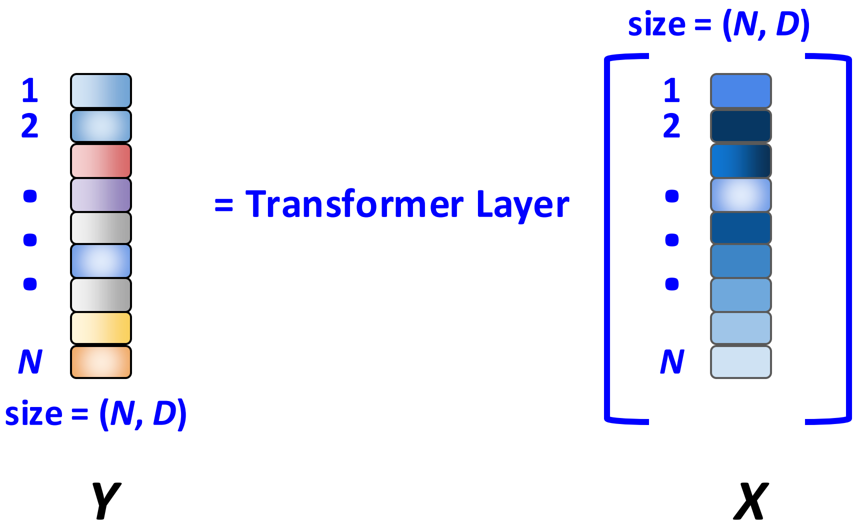


- **Till this point, the outputs are linear combinations of vectors in *X* (some non-linearity from the softmax function, but still restrictive).**

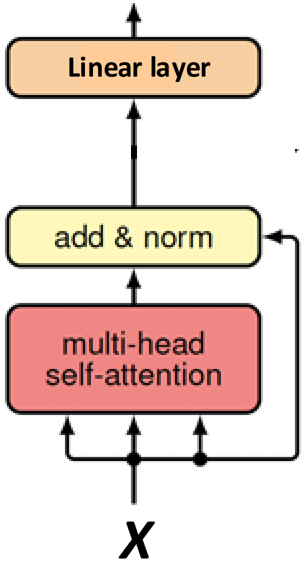
Restricted scope of learning till this point.



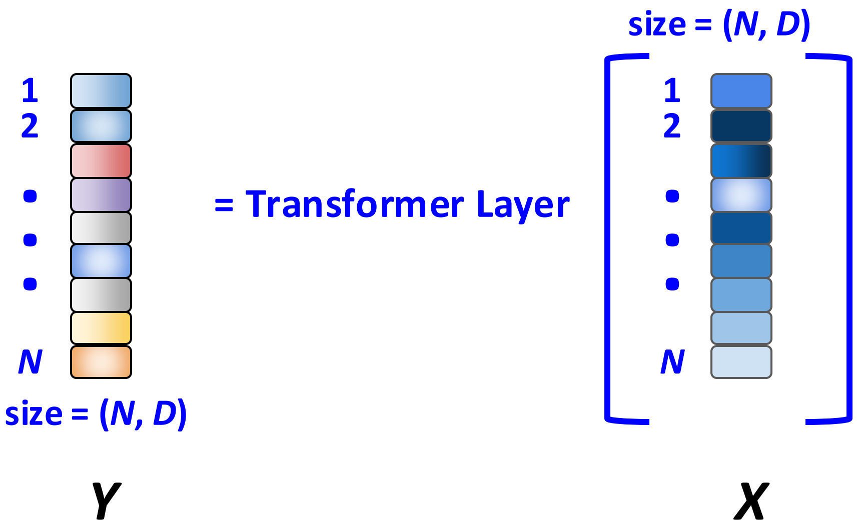
Vision Transformers

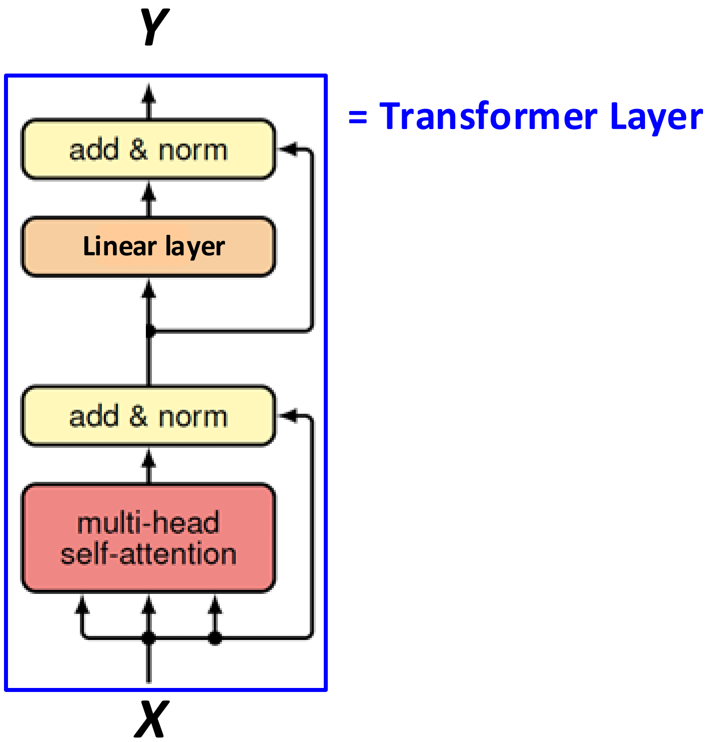


- Introduce an additional linear layer!



Vision Transformers





[Ref.5] Deep Learning: Foundations and Concepts: Christopher M. Bishop, Hugh Bishop

ViTs and their Computational Costs

Yay! My research.

To understand the computational cost & connection to my research

Image

⇩

Neural Networks

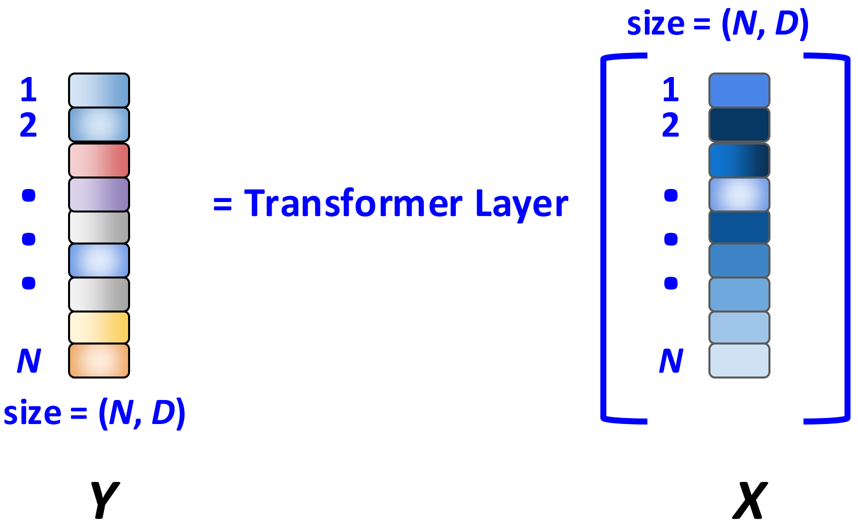
⇩

Analysis

NOTE:

Remember that with current available hardware (GPU/CPU):

- we are <u>memory</u> constrained
- we are also <u>time</u> constrained
- we want the <u>best</u> performance



- To complete this operation, the complexity is of the order N^2 !

Due to self-attention mechanism in $Y = \text{Softmax}(QK^T)V$

Image

⇩

Neural Networks

⇩

Analysis

NOTE:

Remember that with current available hardware (GPU/CPU):

- we are <u>memory</u> constrained
- we are also <u>time</u> constrained
- we want the <u>best</u> performance

My analysis is for dense perception!

Image

⇩

Transformer Architecture

⇩

Dense perception

We have great performance from ViTs, can we make them efficient?

Image

⇩

Transformer Architecture

⇩

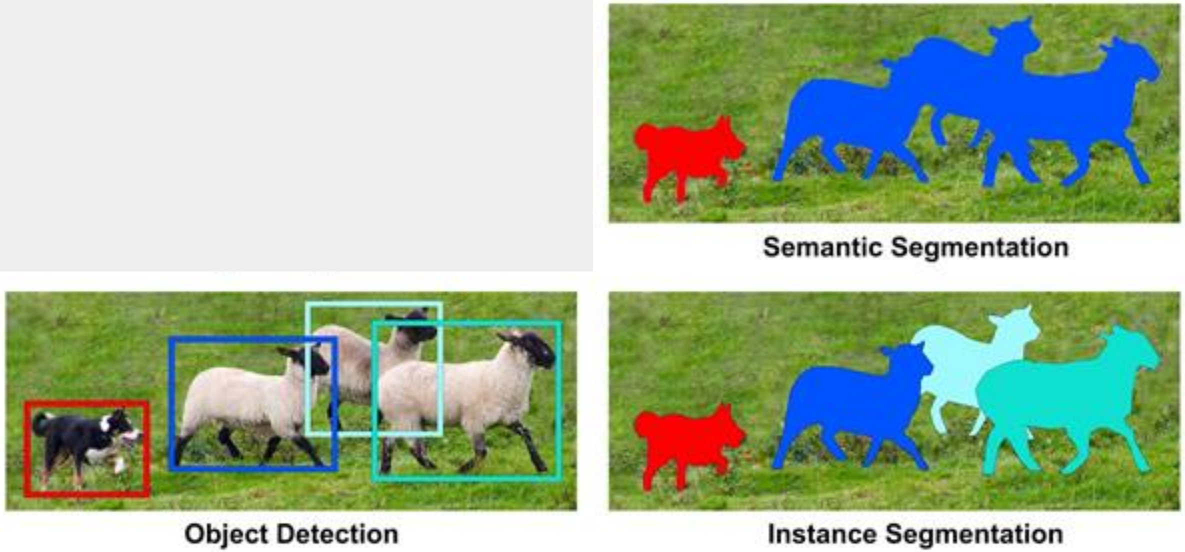
Dense perception

NOTE:

Remember that with current available hardware (GPU/CPU):

- we are memory constrained
- we are also time constrained
- we want the best performance

Dense prediction = Object detection, Image segmentation, etc.



[Ref.6] Image source: https://manipulation.csail.mit.edu/segmentation.html

Image

⇩

Transformer Architecture

⇩

Dense perception

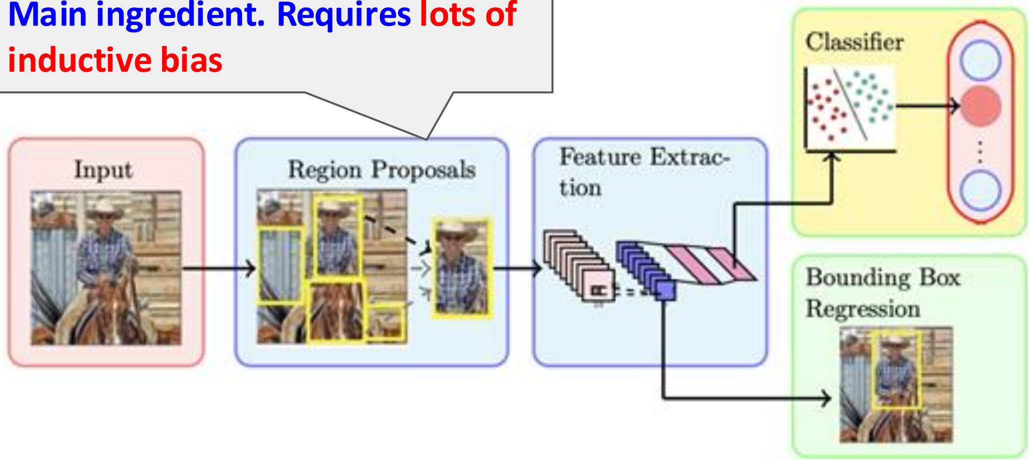
NOTE:

Remember that with current available hardware (GPU/CPU):

- we are memory constrained
- we are also time constrained
- we want the best performance

Traditionally …

Main ingredient. Requires lots of inductive bias



- Region Proposals : Needs carefully chosen
  - anchor box sizes (the yellow boxes) and aspect ratios
  - Non-Maximum Suppression (NMS) to remove duplicates

[Ref.6] Image source: https://www.cse.iitm.ac.in/~miteshk/CS7015/Slides/Handout/Lecture12.pdf

Image

⇩

Transformer Architecture

⇩

Dense perception

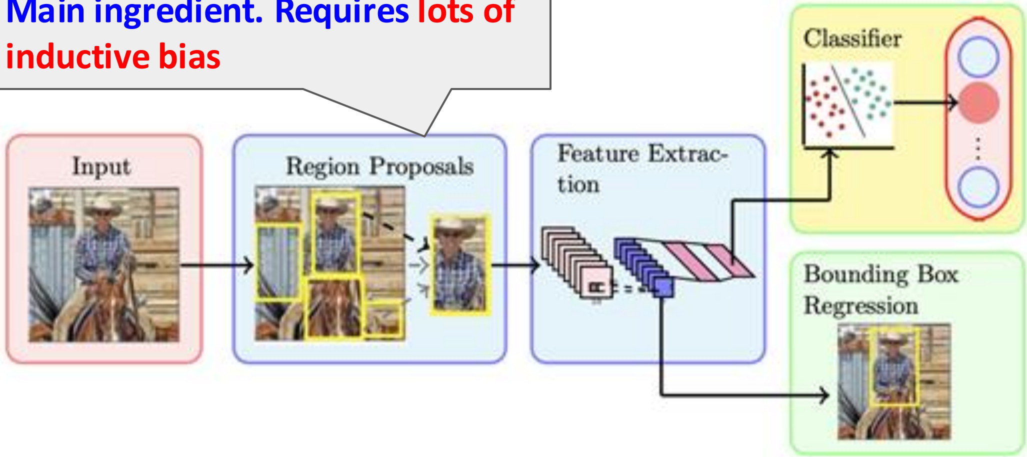
NOTE:

Remember that with current available hardware (GPU/CPU):

- we are memory constrained
- we are also time constrained
- we want the best performance

Traditionally …

Main ingredient. Requires lots of inductive bias



For segmentation, instead of bounding boxes, we predict masks.

[Ref.6] Image source: https://www.cse.iitm.ac.in/~miteshk/CS7015/Slides/Handout/Lecture12.pdf

Image

⇩

Transformer Architecture

⇩

Dense perception

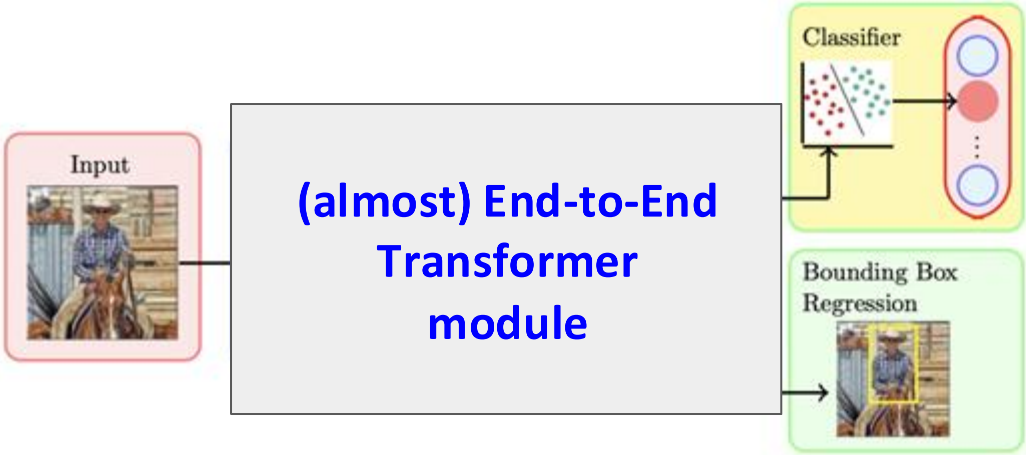
NOTE:

Remember that with current available hardware (GPU/CPU):

- we are memory constrained
- we are also time constrained
- we want the best performance

DEtection TRansformer [Ref. 7]

ViTs to the rescue. Enter DETR!



[Ref.7] End-to-End Object Detection with Transformers, ECCV 2020

Image
⇩
Transformer Architecture
⇩
Dense perception

NOTE:

Remember that with current available hardware (GPU/CPU):
- we are <u>memory</u> constrained
- we are also <u>time</u> constrained
- we want the <u>best</u> performance

DETR but for segmentation

ViTs to the rescue. Enter Mask2Former [REF. 8]!

Image → (almost) End-to-End Transformer module → (Mask, Label)

[Ref.8] Masked-attention Mask Transformer for Universal Image Segmentation, CVPR 2022

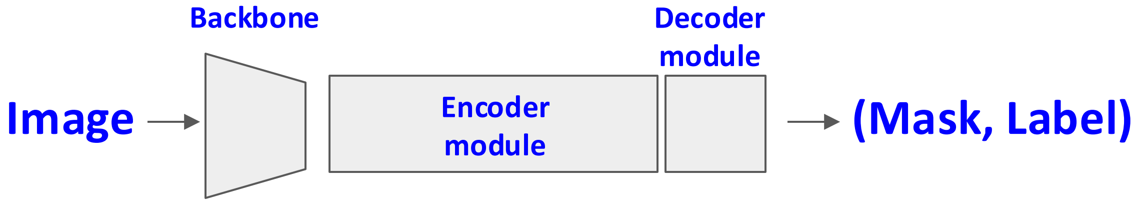
Image
⇩
Transformer Architecture
⇩
Dense perception

NOTE:

Remember that with current available hardware (GPU/CPU):

- we are <u>memory</u> constrained
- we are also <u>time</u> constrained
- we want the <u>best</u> performance

Mask2Former:



- Backbone = it's a simple feature extractor from images. For example, any CNN that provides difference scales of features.

"Backbone" cause the rest can't work without it.

[Ref.8] Masked-attention Mask Transformer for Universal Image Segmentation, CVPR 2022

Image

⇩

Transformer Architecture

⇩

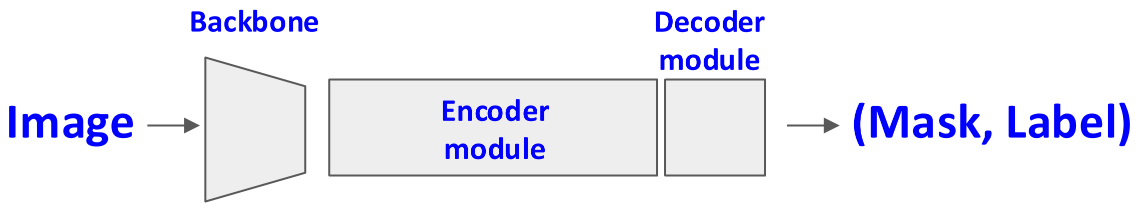
Dense perception

NOTE:

Remember that with current available hardware (GPU/CPU):

- we are memory constrained
- we are also time constrained
- we want the best performance

Mask2Former:



- **Encoder module = Make all the multi-scale features "attend" to each other and enhance their representations.**

[Ref.8] Masked-attention Mask Transformer for Universal Image Segmentation, CVPR 2022

Image

⇩

Transformer Architecture

⇩

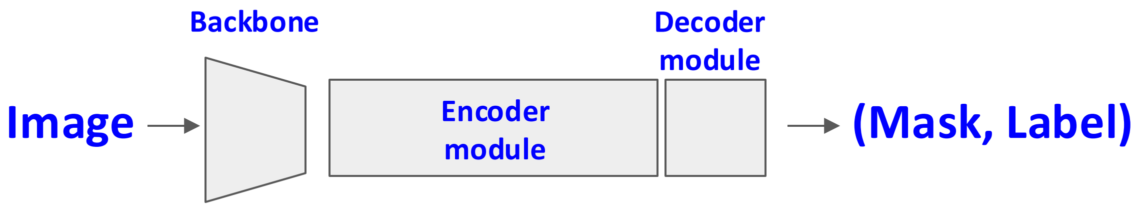
Dense perception

NOTE:

Remember that with current available hardware (GPU/CPU):

- we are <u>memory</u> constrained
- we are also <u>time</u> constrained
- we want the <u>best</u> performance

Mask2Former:



- Decoder module = Take the "enhanced" features from encoder and decode them into masks and corresponding labels.

Image

⇩

Transformer Architecture

⇩

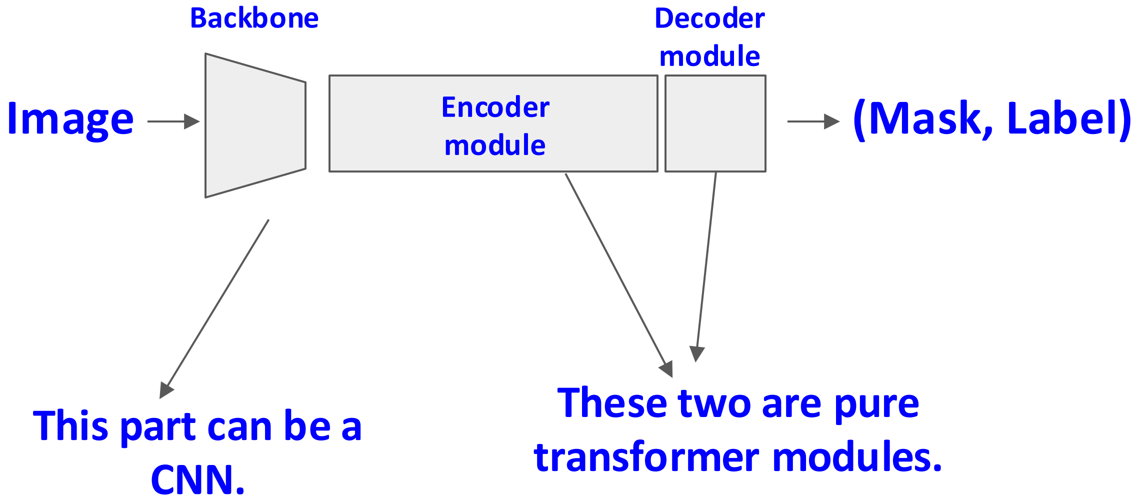
Dense perception

NOTE:

Remember that with current available hardware (GPU/CPU):

- we are memory constrained
- we are also time constrained
- we want the best performance

Mask2Former:



[Ref.7] End-to-End Object Detection with Transformers, ECCV 2020

Image

⇩

Transformer Architecture

⇩

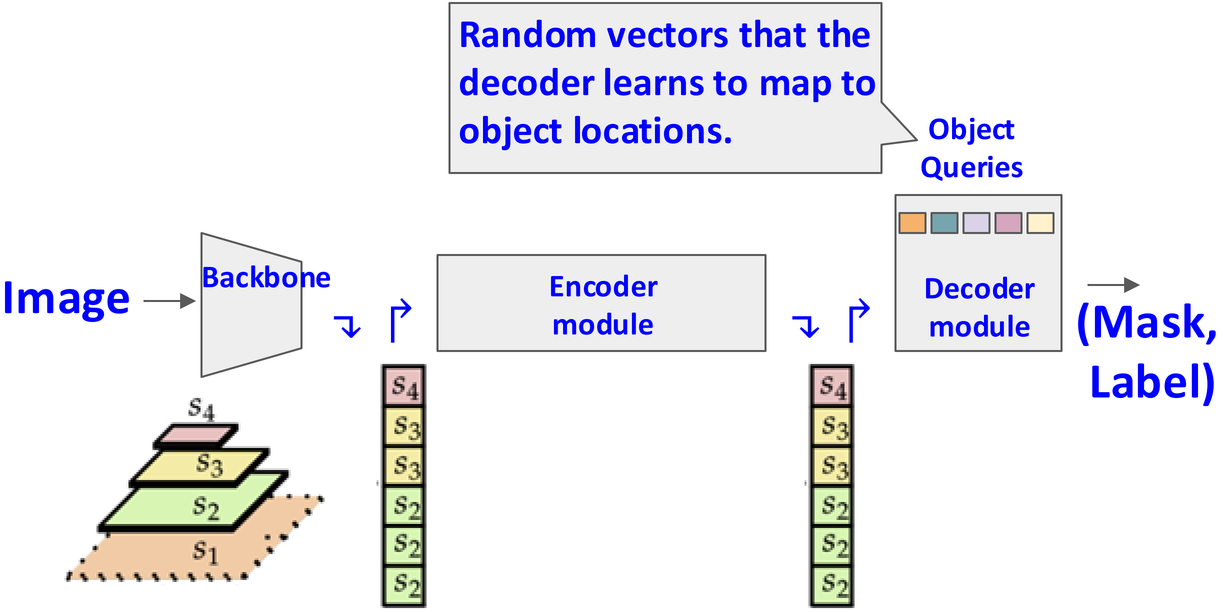
Dense perception

NOTE:

Remember that with current available hardware (GPU/CPU):

- we are <u>memory</u> constrained
- we are also <u>time</u> constrained
- we want the <u>best</u> performance

Mask2Former:



Image

⇩

Transformer Architecture

⇩

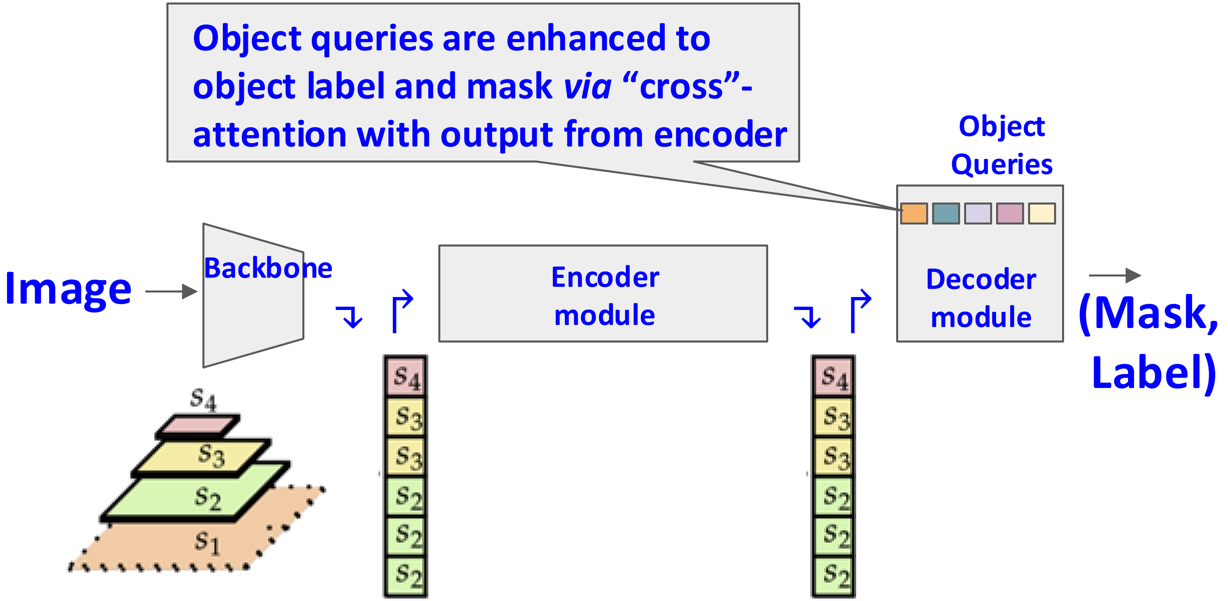
Dense perception

NOTE:

Remember that with current available hardware (GPU/CPU):

- we are <u>memory</u> constrained
- we are also <u>time</u> constrained
- we want the <u>best</u> performance

Mask2Former:



Image

⇩

Transformer Architecture

⇩

Dense perception

Mask2Former:

NOTE:

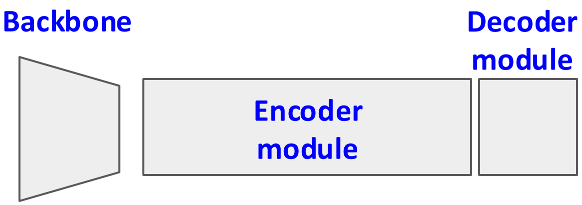
Remember that with current available hardware (GPU/CPU):

- we are <u>memory</u> constrained
- we are also <u>time</u> constrained
- we want the <u>best</u> performance

Image
⇩

Transformer Architecture

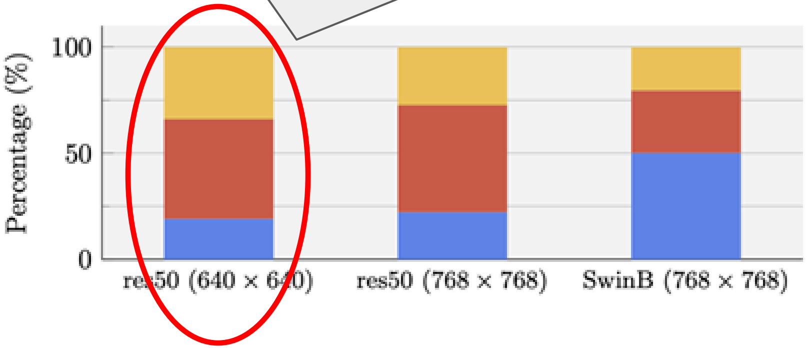
⇩
Dense perception



Mask2Former:

- **Unfortunately, for Mask2Former, good performance comes at a price of expensive computations.**

We made the backbone smaller, but now the encoder brings the most computations.



Image

⇩

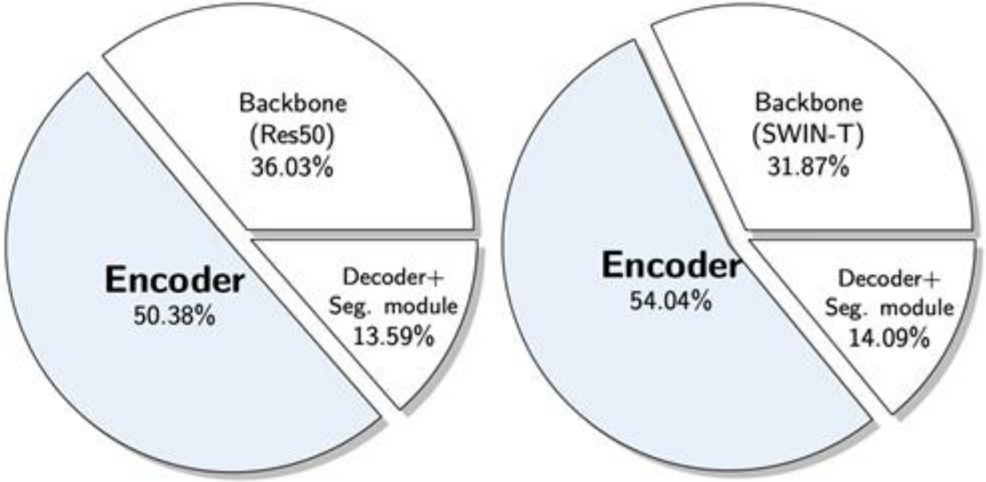
Transformer Architecture

⇩

Dense perception

Mask2Former:

- Unfortunately for Mask2Former, good performance comes at a price of expensive computations.



Image

⇩

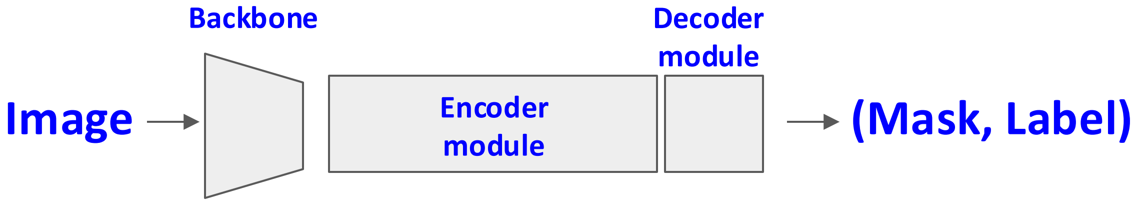
Transformer Architecture

⇩

Dense perception

Mask2Former:

- Introducing PRO-SCALE (recently accepted to ICLR 2025)



Image

⇩

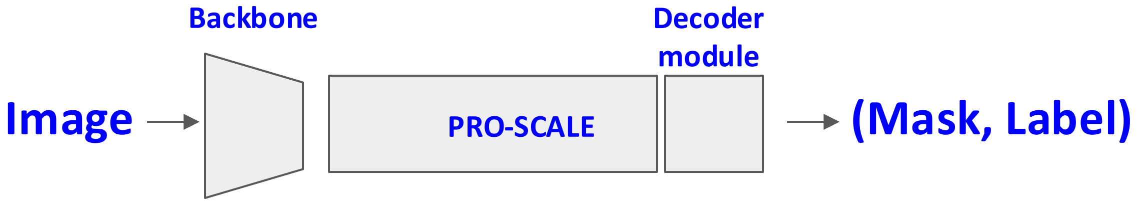
Transformer Architecture

⇩

Dense perception

Mask2Former:

- Introducing PRO-SCALE (recently accepted to ICLR 2025)



Image

⇩

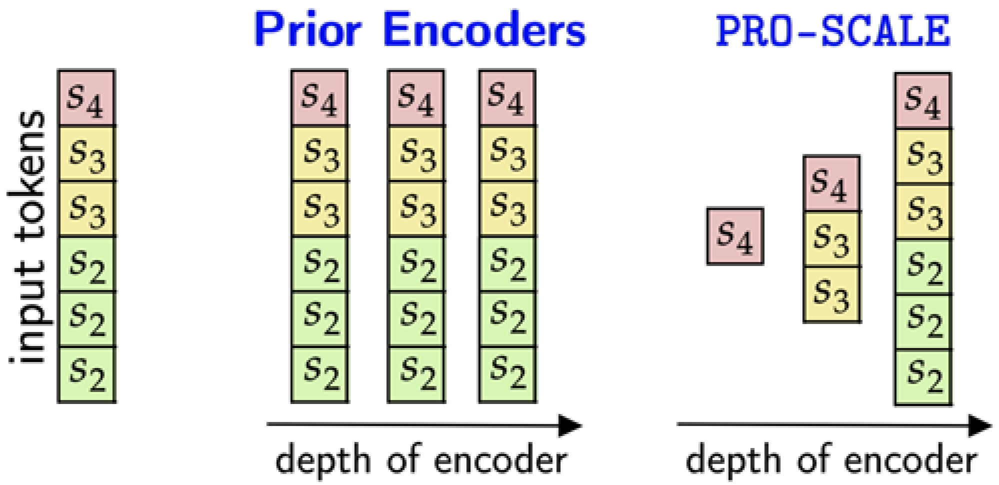
Transformer Architecture

⇩

Dense perception

Mask2Former:

- Introducing PRO-SCALE (recently accepted to ICLR 2025)



Image

⇩

Transformer Architecture

⇩

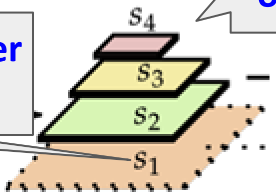
Dense perception

Mask2Former:

- Introducing PRO-SCALE (recently accepted to ICLR 2025)

Extracted from the last layer of the backbone

Extracted from the initial layer of the backbone



- s_4 tokens computed from the backbone are already contain a lot of good information!

Image

⇩

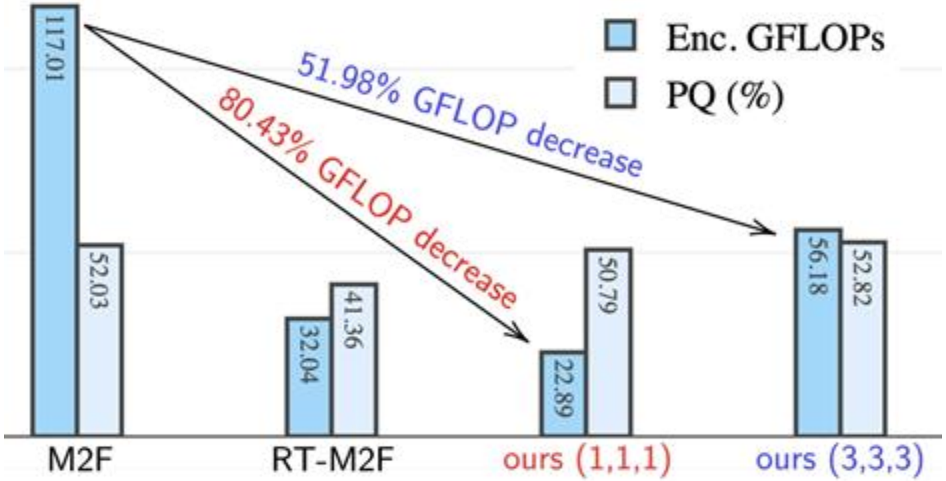
Transformer Architecture

⇩

Dense perception

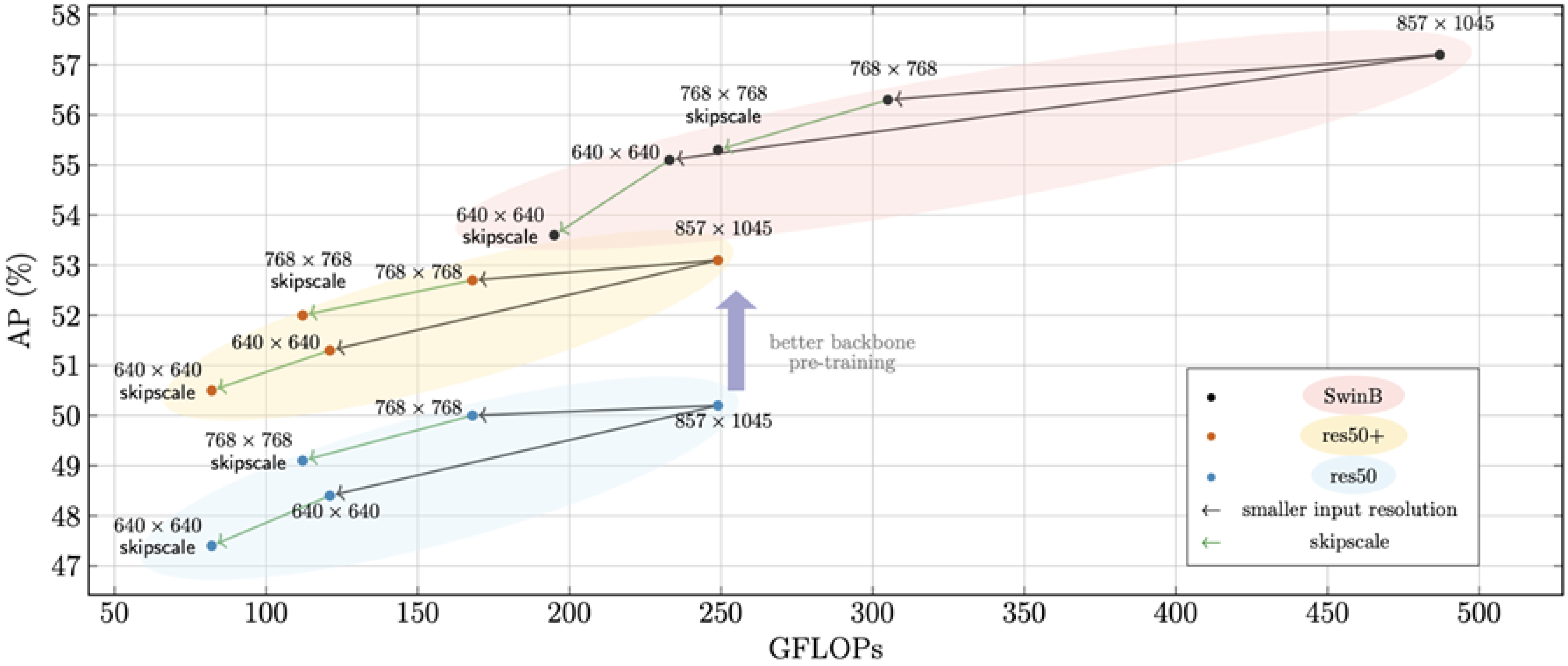
Mask2Former:

- PRO-SCALE works great !
- Maintains performance while reducing computations.



From Paper to Deployment

So did we solve the computations problem?



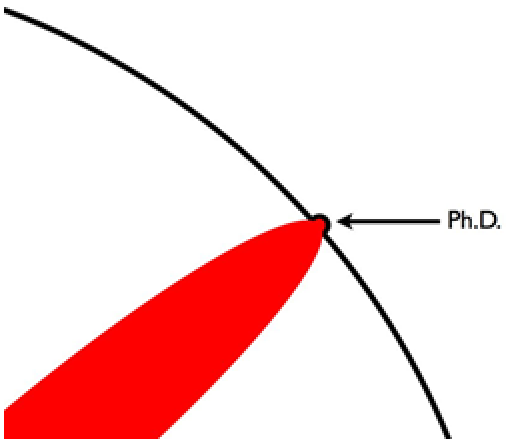
857 × 1045
768 × 768
768 × 768
skipscale
640 × 640
640 × 640
skipscale
857 × 1045
768 × 768
skipscale
768 × 768
640 × 640
640 × 640
skipscale
better backbone
pre-training
768 × 768
857 × 1045
768 × 768
skipscale
640 × 640
640 × 640
skipscale
SwinB
res50+
res50
smaller input resolution
skipscale
AP (%)
GFLOPs

For dense perception,

- **Real world usage requires (or hopes for) no re-training.**
- **Pre-training is at core.**
- **Performance is becoming a data problem given compute and transformers.**
- **Making the models smaller with same performance is still an active research area.**

(unrelated) Pursuing Ph.D.

- **Getting a mentor early helps immensely.**
- **You just have to be right once.**
- **Industry has become a solid option.**
- **AI has driven Engineers and Scientists to same point, going hand-in-hand.**



[Ref.9] The illustrated guide to a Ph.D., Matt Might

Thank you!