Deep Learning for Computer Vision

From Transformers to Vision Transformers

Vineeth N Balasubramanian

Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad

Acknowledgements

- Most of this lecture's slides are based on Justin Johnson's Deep Learning for Computer Vision course from University of Michigan
- Unless explicitly specified, assume that content and figures are either directly taken or adapted from above source

Transformers in NLP

- Transformers[1], have revolutionized Natural Language Processing (NLP) tasks by introducing a novel architecture based on self-attention mechanisms

[1] Vaswani et al, Attention is All You Need, NeurIPS 2017

Transformers in NLP

- Transformers[1], have revolutionized Natural Language Processing (NLP) tasks by introducing a novel architecture based on self-attention mechanisms
- Sample NLP tasks include:
  - Machine Translation
  - Language Understanding
  - Text Generation
  - Question Answering
  - Named Entity Recognition

[1] Vaswani et al, Attention is All You Need, NeurIPS 2017

Transformers in NLP

- Transformers[1], have revolutionized Natural Language Processing (NLP) tasks by introducing a novel architecture based on self-attention mechanisms
- Sample NLP tasks include:
  - Machine Translation
  - Language Understanding
  - Text Generation
  - Question Answering
  - Named Entity Recognition
- BERT [1], RoBERTa [2], DeBERTa [3], T5 [4], GPT v1-4 [5] represent major contributions in transformer-based models developed for these NLP tasks

[1] Vaswani et al, Attention is All You Need, NeurIPS 2017

Transformers in Vision

Creating the input sequence

- Transformer encoder requires a sequence of vectors as input
- In language applications, this sequence is just the set of (ordered) word embeddings that correspond to the tokens within our input

Transformers in Vision

Creating the input sequence

- Transformer encoder requires a sequence of vectors as input
- In language applications, this sequence is just the set of (ordered) word embeddings that correspond to the tokens within our input
- How can we form such a token sequence if our input is an image?
- What if we use each pixel as a token?

Transformers in Vision

Creating the input sequence

- Transformer encoder requires a sequence of vectors as input
- In language applications, this sequence is just the set of (ordered) word embeddings that correspond to the tokens within our input
- How can we form such a token sequence if our input is an image?
- What if we use each pixel as a token?
- It blows up the memory usage and computations! For an image of size $3 \times 224 \times 224$, we need to store $(3 \times 224 \times 224)^2 = 22$ billion parameters for self-attention calculation.

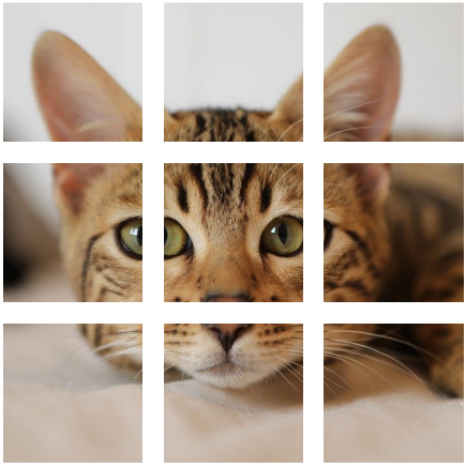
Vision Transformer[2]

Consider an RGB image of size
224×224

[2]Dosovitskiy et al, An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, ICLR 2021

Vision Transformer

Divide the image into 196 (14×14) small images of size 16×16



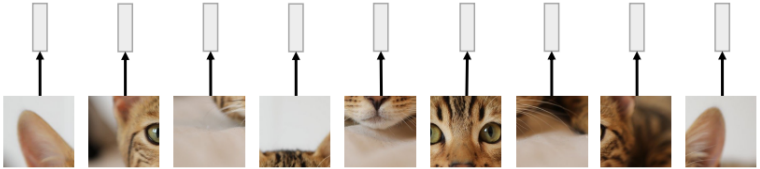
Vision Transformer

N input patches, each of shape 3x16x16

Vision Transformer

Linear projection to D-dimensional vector

N input patches, each of shape 3x16x16

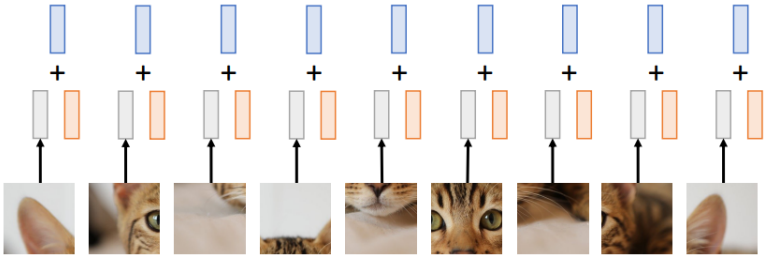


Vision Transformer

Add positional embedding: learned D-dim vector

Linear projection to D-dimensional vector

N input patches, each of shape 3x16x16



Vision Transformer

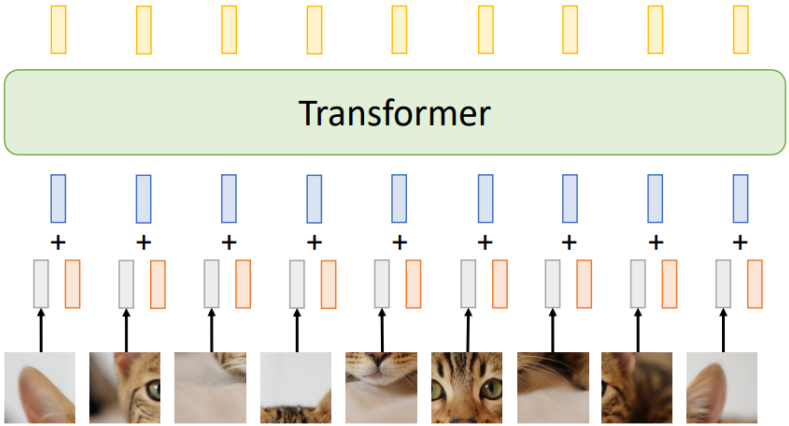
Output vectors

Exact same as NLP Transformer!

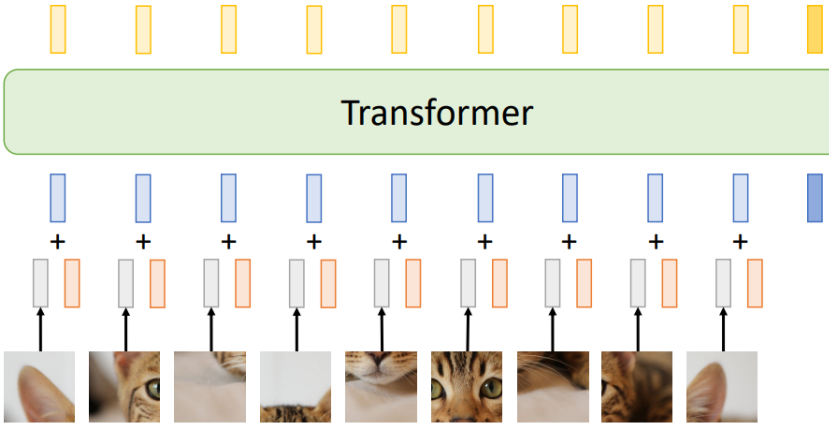
Add positional embedding: learned D-dim vector

Linear projection to D-dimensional vector

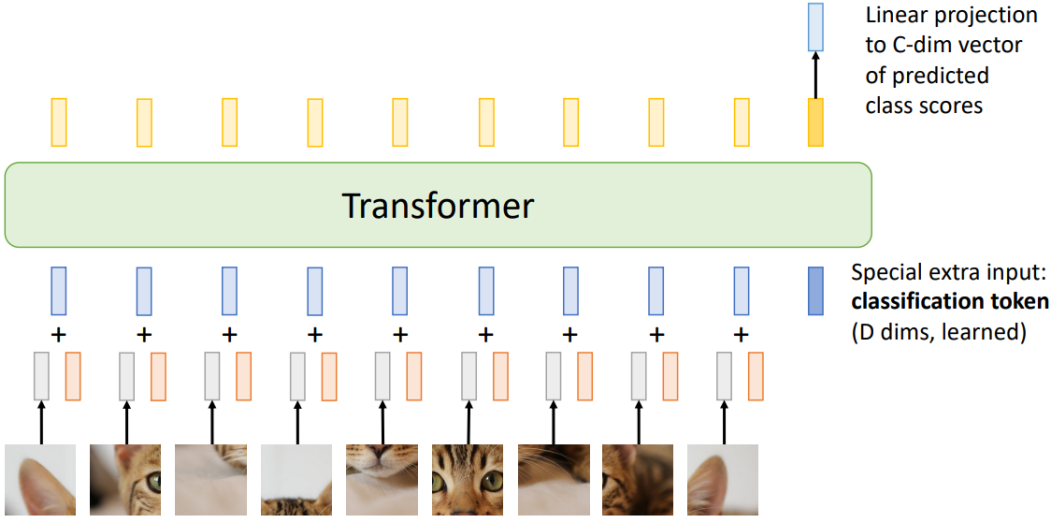
N input patches, each of shape 3x16x16



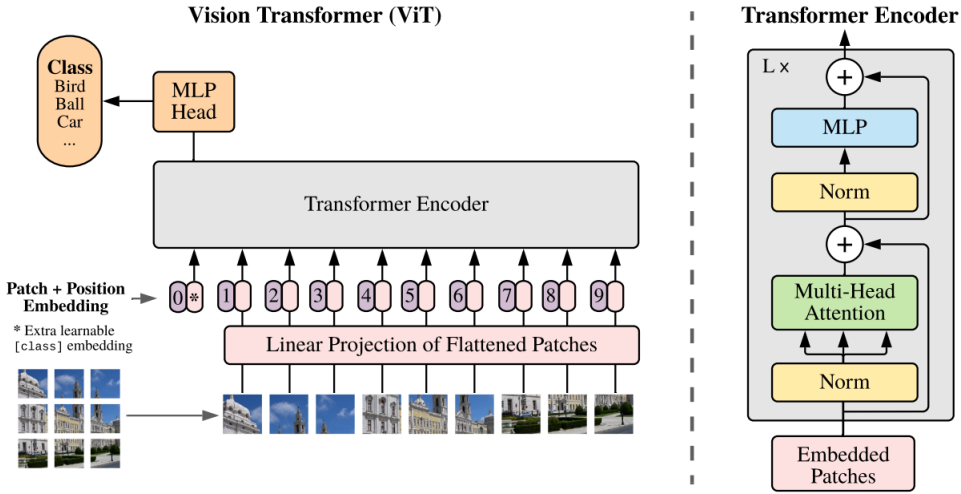
Vision Transformer



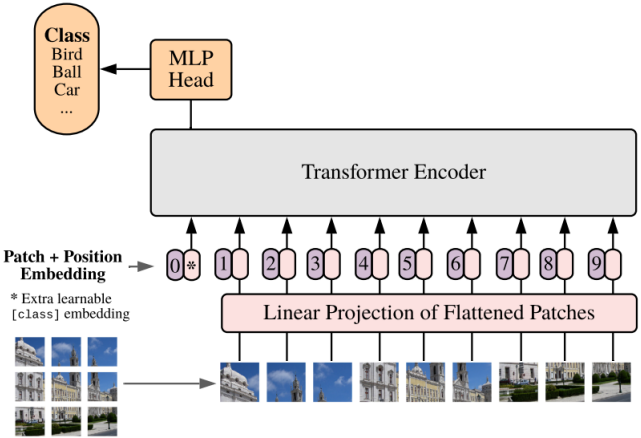
Vision Transformer



Vision Transformer: Overall Architecture

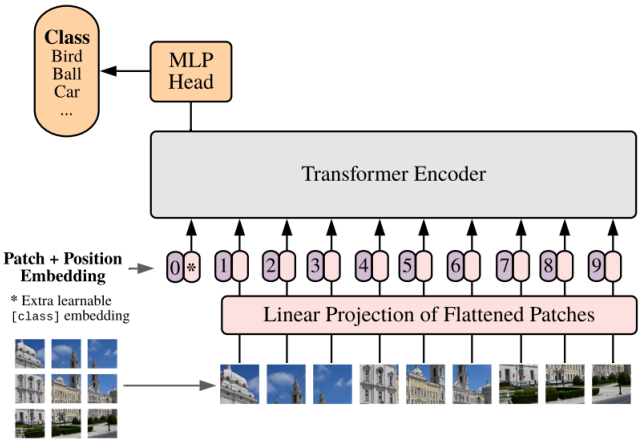


Vision Transformer: Overall Architecture



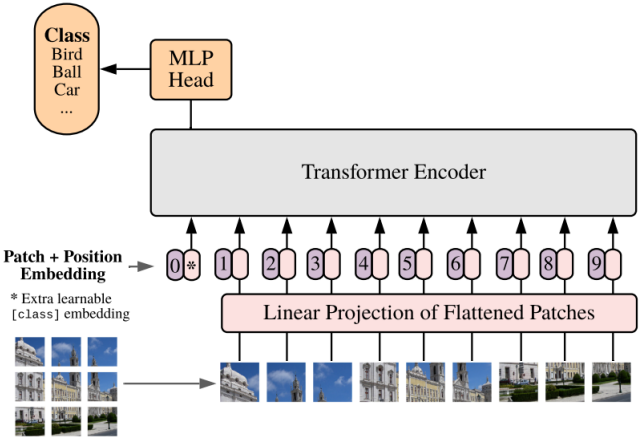
- Using a pixel as a token, for an image of size $3 \times 224 \times 224$, we had $(3 \times 224 \times 224)^2 = 22$ billion parameters for self-attention calculation. Now?

Vision Transformer: Overall Architecture



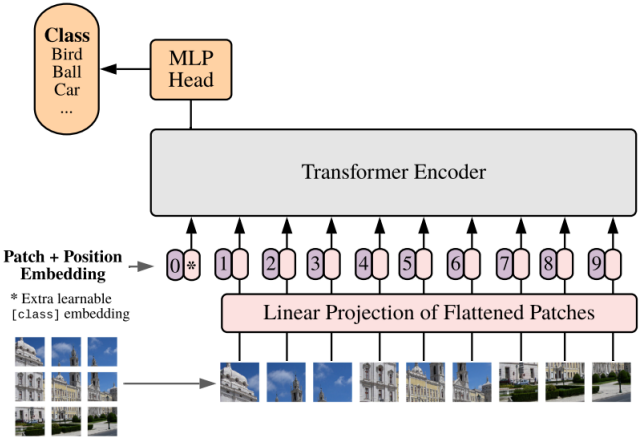
- Using a pixel as a token, for an image of size $3 \times 224 \times 224$, we had $(3 \times 224 \times 224)^2 = 22$ billion parameters for self-attention calculation. Now?
- $(14 \times 14)^2 = 38,416$ parameters (approx 150 KB) per transformer layer!

Vision Transformer: Overall Architecture



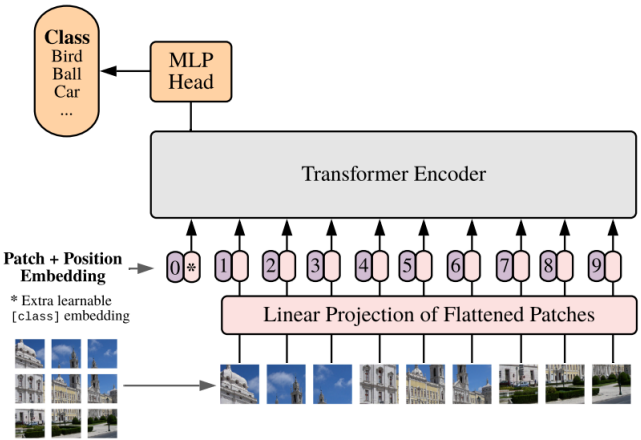
- Do we have a vision architecture without convolution?!

Vision Transformer: Overall Architecture



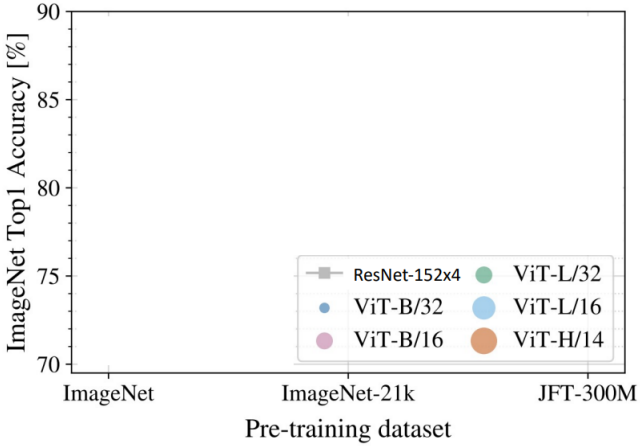
- Do we have a vision architecture without convolution?!
- No! The linear projection from patches is a convolution (with stride being patch size)

Vision Transformer: Overall Architecture



- Do we have a vision architecture without convolution?!
- No! The linear projection from patches is a convolution (with stride being patch size)
- Also, MLPs in Transformer are stacks of 1×1 convolution

Vision Transformer (ViT) vs ResNets[3]



B = Base
L = Large
H = Huge

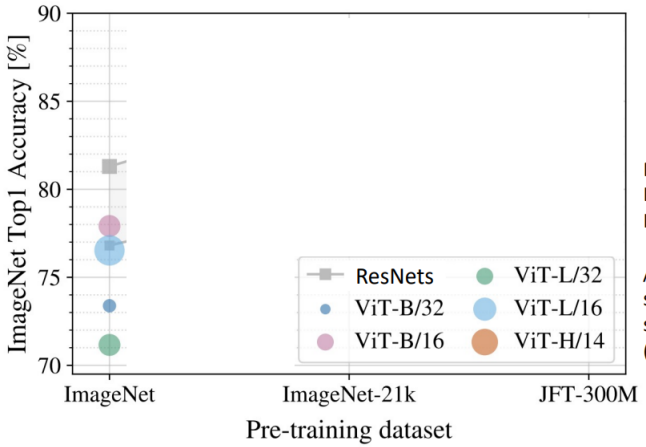
/32, /16, /14 is patch size; smaller patch size is a bigger model (more patches)

[3]Dosovitskiy et al, An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, ICLR 2021

Vision Transformer (ViT) vs ResNets[4]

Recall: ImageNet dataset has 1k categories, 1.2M images

When trained on ImageNet, ViT models perform worse than ResNets



B = Base
L = Large
H = Huge

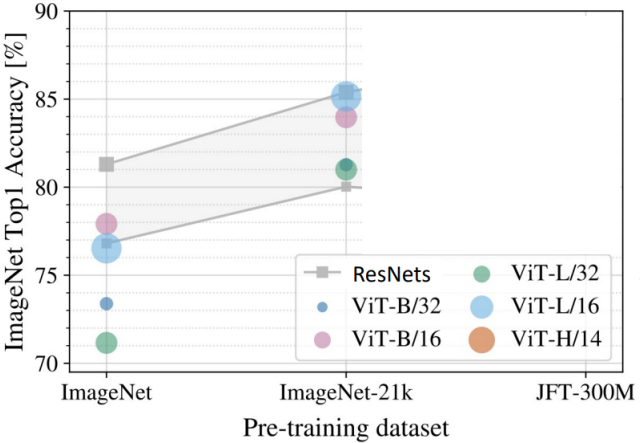
/32, /16, /14 is patch size; smaller patch size is a bigger model (more patches)

[4]Dosovitskiy et al, An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, ICLR 2021

Vision Transformer (ViT) vs ResNets[5]

ImageNet-21k has 14M images with 21k categories

If you pretrain on ImageNet-21k and fine-tune on ImageNet, ViT does better: big ViTs match big ResNets



B = Base
L = Large
H = Huge

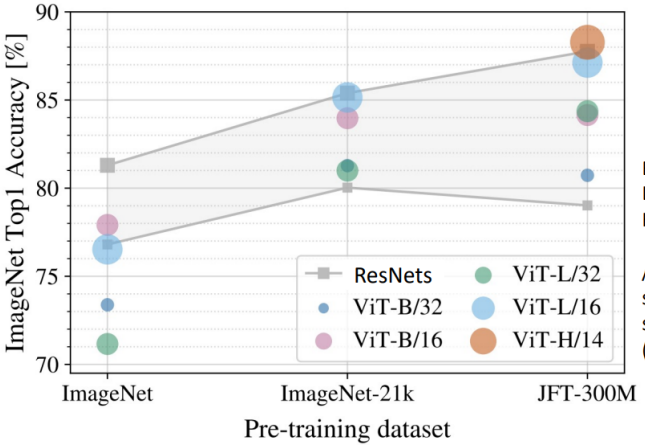
/32, /16, /14 is patch size; smaller patch size is a bigger model (more patches)

[5]Dosovitskiy et al, An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, ICLR 2021

Vision Transformer (ViT) vs ResNets[6]

JFT-300M is an internal Google dataset with 300M labeled images

If you pretrain on JFT and finetune on ImageNet, large ViTs outperform large ResNets



B = Base
L = Large
H = Huge

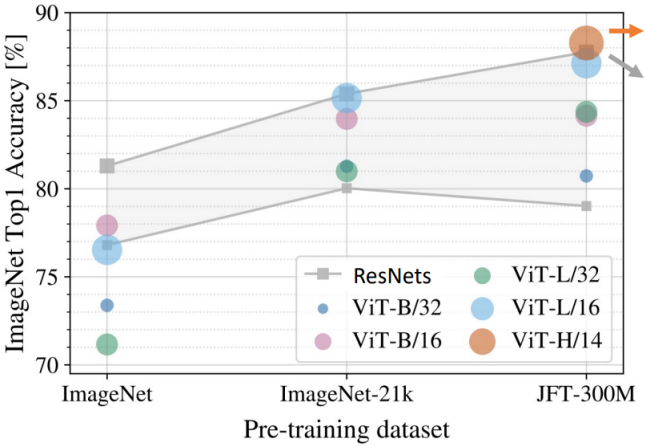
/32, /16, /14 is patch size; smaller patch size is a bigger model (more patches)

[6] Dosovitskiy et al, An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, ICLR 2021

Vision Transformer (ViT) vs ResNets[7]

JFT-300M is an internal Google dataset with 300M labeled images

If you pretrain on JFT and finetune on ImageNet, large ViTs outperform large ResNets



ViT: 2.5k TPU-v3 core days of training

ResNet: 9.9k TPU-v3 core days of training

ViTs make more efficient use of GPU / TPU hardware (matrix multiply is more hardware-friendly than conv)

[7] Dosovitskiy et al, An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, ICLR 2021

Improving Vision Transformer Performance[8]

Regularization and Augmentation

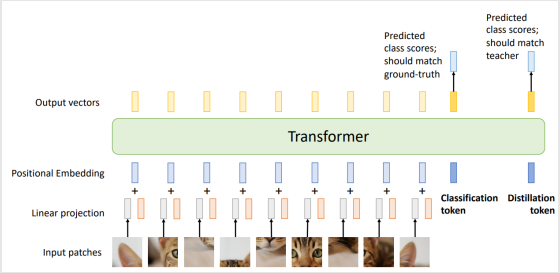
- Regularization
  - Weight Decay
  - Stochastic Depth
  - Dropout (in FFN layers of Transformer)
- Augmentation
  - MixUp
  - RandAugment

[8] Steiner et al, How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers, TMLR 2022

Improving Vision Transformer Performance[9]

ViT with Distillation: Data-efficient image Transformers (DeiT)

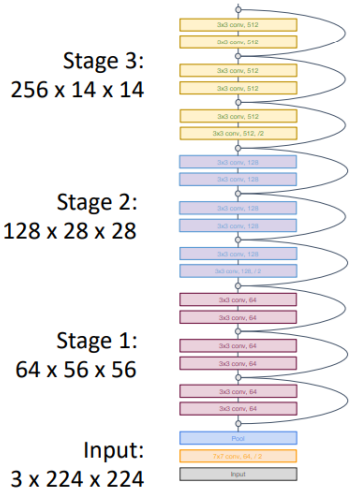
- From a trained teacher model (**large CNN**), train student model (**ViT**) using standard cross-entropy loss AND KL Divergence loss with teacher outputs



- Top-1 accuracy increases from 77.9 to 83.4 (and even more with higher training epochs and resolution)

[9] Touvrom et al, Training data-efficient image transformers distillation through attention, ICML 2021

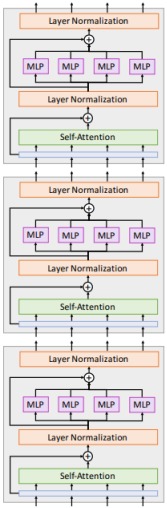
ViT vs CNN



In most CNNs (including ResNets), resolution decreases and number of channels increases as we go deep into the network (**Hierarchical architecture**)

This is useful since objects can occur in multiple scales

ViT vs CNN



3rd block:
768 x 14 x 14

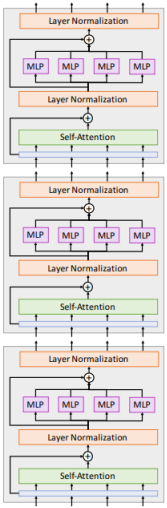
2nd block:
768 x 14 x 14

1st block:
768 x 14 x 14

Input:
3 x 224 x 224

In a ViT, all blocks have same resolution and number of channels (**Isotropic architecture**)

ViT vs CNN



3rd block:
768 x 14 x 14

2nd block:
768 x 14 x 14

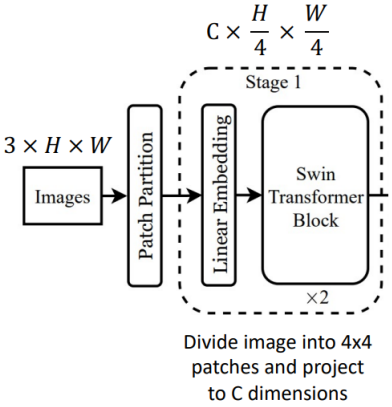
1st block:
768 x 14 x 14

Input:
3 x 224 x 224

In a ViT, all blocks have same resolution and number of channels (**Isotropic architecture**)

Can we build a hierarchical ViT model?

Swin Transformer[10]

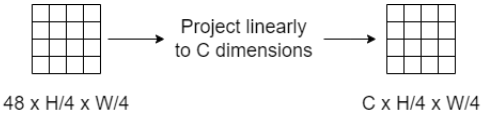


[10] Liu et al, Swin Transformer: Hierarchical Vision Transformer using Shifted Windows, ICCV 2021

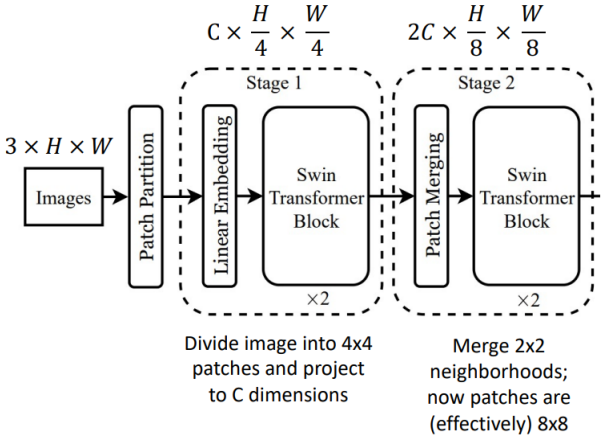
Swin Transformer

Patch Partition:

Linear Embedding:

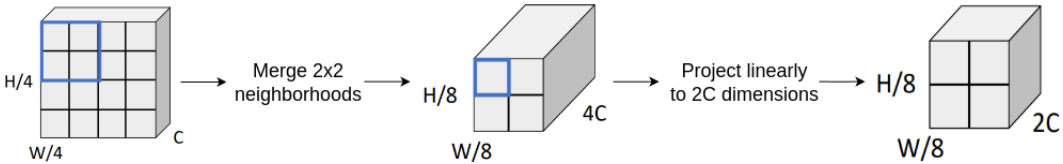


Swin Transformer

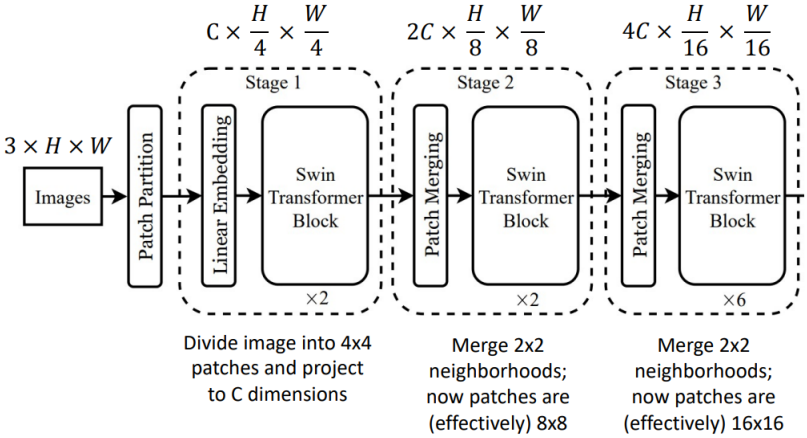


Swin Transformer

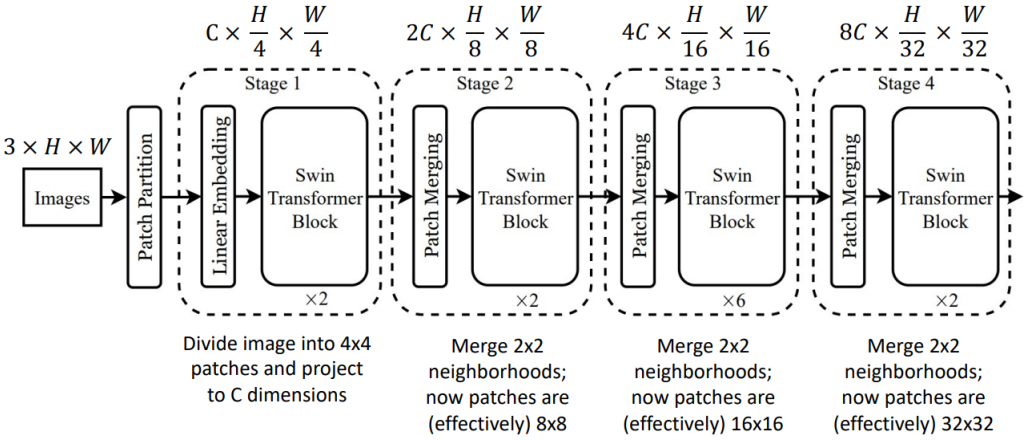
Patch Merging:



Swin Transformer



Swin Transformer



Swin Transformer

Problem: With $H \times W$ grid of tokens, attention matrices are of size H^2W^2, which is quadratic in image size

Swin Transformer

Problem: With $H \times W$ grid of tokens, attention matrices are of size H^2W^2, which is quadratic in image size

Solution: **Window Attention** - Instead of allowing each token to attend to all other tokens in the image, divide image into windows of $M \times M$ tokens and compute attention within each window. Now the attention matrices are of size M^2HW, which is linear in image size for a fixed M. Swin uses M=7 throughout the network.

Swin Transformer

Problem: With window attention, there is no communication between windows, restricting the flow of information across different regions of the image

Swin Transformer

Problem: With window attention, there is no communication between windows, restricting the flow of information across different regions of the image

Solution: **Shifted Window Attention** - Shift the attention windows progressively across the image in a hierarchical manner, enabling interactions between patches across different levels of abstraction.

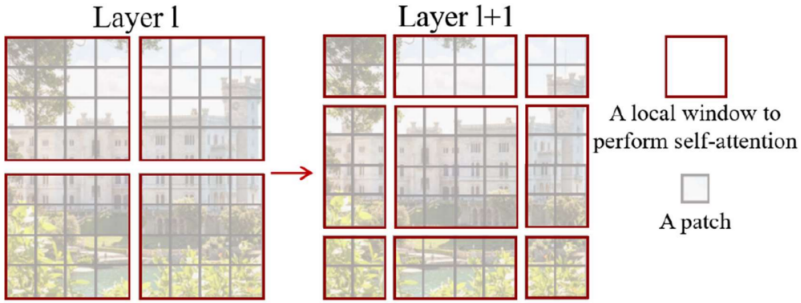
Swin Transformer

Problem: With window attention, there is no communication between windows, restricting the flow of information across different regions of the image

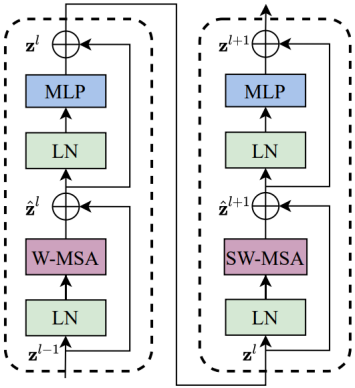
Solution: **Shifted Window Attention** - Shift the attention windows progressively across the image in a hierarchical manner, enabling interactions between patches across different levels of abstraction.

This allows for capturing long-range dependencies while maintaining computational tractability, which is crucial for applications such as image classification, object detection, and semantic segmentation.

Swin Transformer: Shifted Window Attention

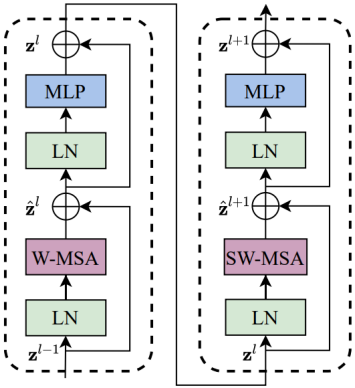


Swin Transformer: Architecture Details



- **LN:** Layer Normalization
- **W-MSA, SW-MSA:** Windowed and Shifted Windowed configurations of self-attention
- **MLP:** Multi Layer Perceptron

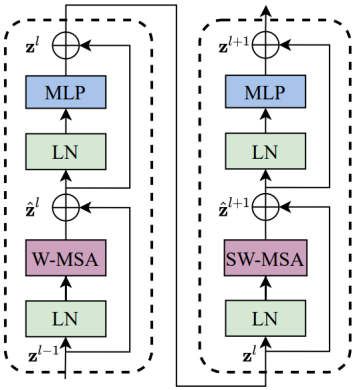
Swin Transformer: Architecture Details



- **LN:** Layer Normalization
- **W-MSA, SW-MSA:** Windowed and Shifted Windowed configurations of self-attention
- **MLP:** Multi Layer Perceptron

What about positional embedding here?

Swin Transformer: Architecture Details

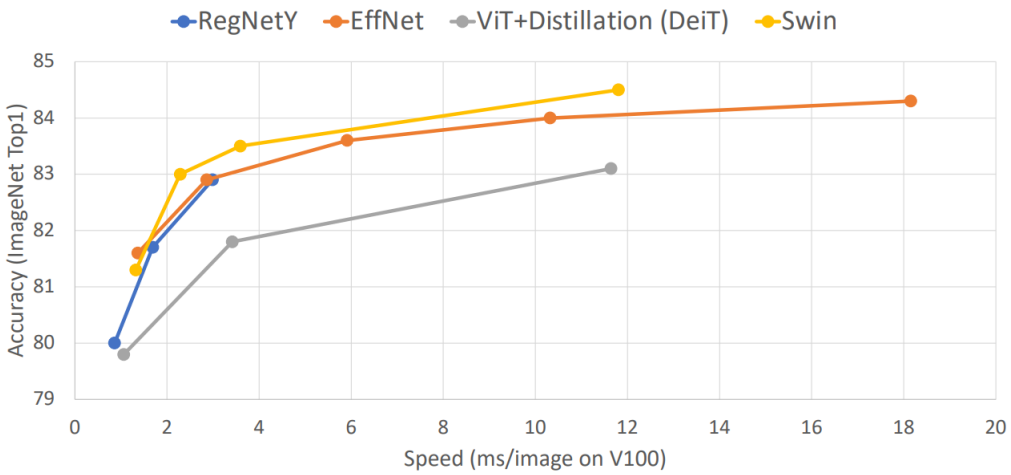


- **LN:** Layer Normalization
- **W-MSA, SW-MSA:** Windowed and Shifted Windowed configurations of self-attention
- **MLP:** Multi Layer Perceptron

What about positional embedding here?
Swin does not use positional embedding! Instead, it encodes relative position between patches when computing attention using learned biases ($B \in \mathbb{R}^{M^2 \times M^2}$). Attention with relative bias defined as:

$$A = \text{Softmax}\left(\frac{QK^T}{\sqrt{D}} + B\right)V$$

Swin Transformer: Speed vs Accuracy Comparison



Task-specific Vision Transformers

- Swin Transformer serves as a general-purpose backbone for diverse vision tasks, including Image Classification, Object Detection, and Semantic Segmentation

[11] Carion et al, "End-to-End Object Detection with Transformers", ECCV 2020
[12] Kirillov, Alexander, et al. "Segment anything", ICCV 2023
[13] Bianchi, Federico, et al. "Contrastive language-image pre-training for the italian language.", arXiv 2021 by Open-AI
[14] Liang, Jingyun, et al. "Swinir: Image restoration using swin transformer.", ICCV 2021

Task-specific Vision Transformers

- Swin Transformer serves as a general-purpose backbone for diverse vision tasks, including Image Classification, Object Detection, and Semantic Segmentation
- In addition to Swin Transformer, recent advancements have seen the emergence of task-specific transformers tailored for specialized vision tasks:
  - **DETR (DEtection TRansformer)**[11]: Specifically designed for object detection tasks, DETR replaces traditional convolutional layers with transformer-based architectures.

[11] Carion et al, "End-to-End Object Detection with Transformers", ECCV 2020
[12] Kirillov, Alexander, et al. "Segment anything", ICCV 2023
[13] Bianchi, Federico, et al. "Contrastive language-image pre-training for the italian language.", arXiv 2021 by Open-AI
[14] Liang, Jingyun, et al. "Swinir: Image restoration using swin transformer.", ICCV 2021

Task-specific Vision Transformers

- Swin Transformer serves as a general-purpose backbone for diverse vision tasks, including Image Classification, Object Detection, and Semantic Segmentation
- In addition to Swin Transformer, recent advancements have seen the emergence of task-specific transformers tailored for specialized vision tasks:
  - **DETR (DEtection TRansformer)**[11]: Specifically designed for object detection tasks, DETR replaces traditional convolutional layers with transformer-based architectures.
  - **Segment Anything (SAM)**[12]: SAM focuses on Semantic Segmentation tasks, leveraging transformer architectures to effectively capture spatial dependencies and context information.

[11] Carion et al, "End-to-End Object Detection with Transformers", ECCV 2020

[12] Kirillov, Alexander, et al. "Segment anything", ICCV 2023

[13] Bianchi, Federico, et al. "Contrastive language-image pre-training for the italian language.", arXiv 2021 by Open-AI

[14] Liang, Jingyun, et al. "Swinir: Image restoration using swin transformer.", ICCV 2021

Task-specific Vision Transformers

- Swin Transformer serves as a general-purpose backbone for diverse vision tasks, including Image Classification, Object Detection, and Semantic Segmentation
- In addition to Swin Transformer, recent advancements have seen the emergence of task-specific transformers tailored for specialized vision tasks:
  - **DETR (DEtection TRansformer)**[11]: Specifically designed for object detection tasks, DETR replaces traditional convolutional layers with transformer-based architectures.
  - **Segment Anything (SAM)**[12]: SAM focuses on Semantic Segmentation tasks, leveraging transformer architectures to effectively capture spatial dependencies and context information.
  - **Other Notable Examples:** CLIP[13] for vision-language tasks, and SwinIR[14] for image restoration.

[11] Carion et al, "End-to-End Object Detection with Transformers", ECCV 2020

[12] Kirillov, Alexander, et al. "Segment anything", ICCV 2023

[13] Bianchi, Federico, et al. "Contrastive language-image pre-training for the italian language.", arXiv 2021 by Open-AI

[14] Liang, Jingyun, et al. "Swinir: Image restoration using swin transformer.", ICCV 2021

References

[1] Jacob Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[2] Yinhan Liu et al. "Roberta: A robustly optimized bert pretraining approach". In: *arXiv preprint arXiv:1907.11692* (2019).

[3] Pengcheng He et al. "Deberta: Decoding-enhanced bert with disentangled attention". In: *arXiv preprint arXiv:2006.03654* (2020).

[4] Colin Raffel et al. "Exploring the limits of transfer learning with a unified text-to-text transformer". In: *Journal of machine learning research* 21.140 (2020), pp. 1–67.

[5] Gokul Yenduri et al. *Generative Pre-trained Transformer: A Comprehensive Review on Enabling Technologies, Potential Applications, Emerging Challenges, and Future Directions*. 2023. arXiv: 2305.10435 [cs.CL].