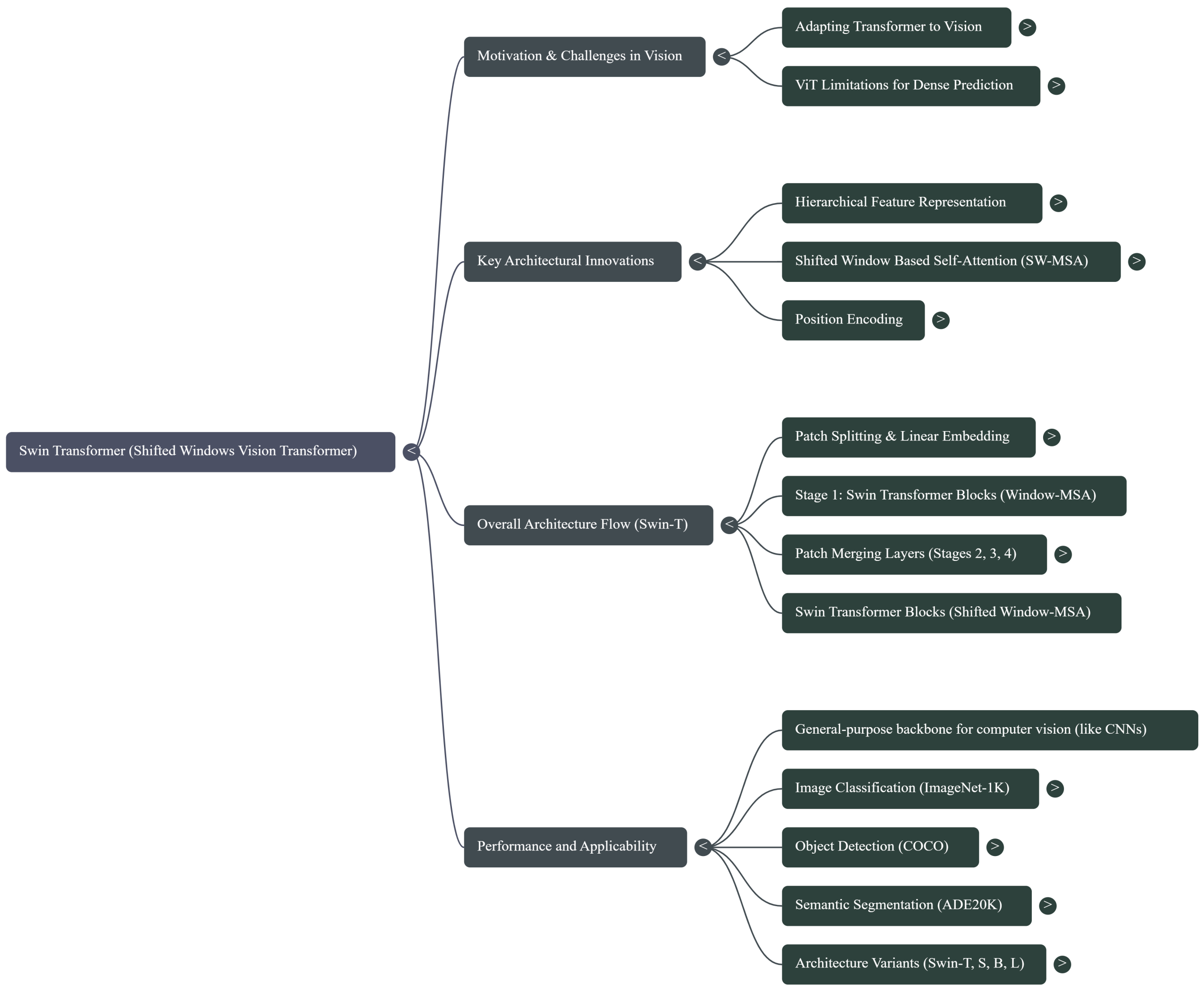
SWIN Transformer research paper notes



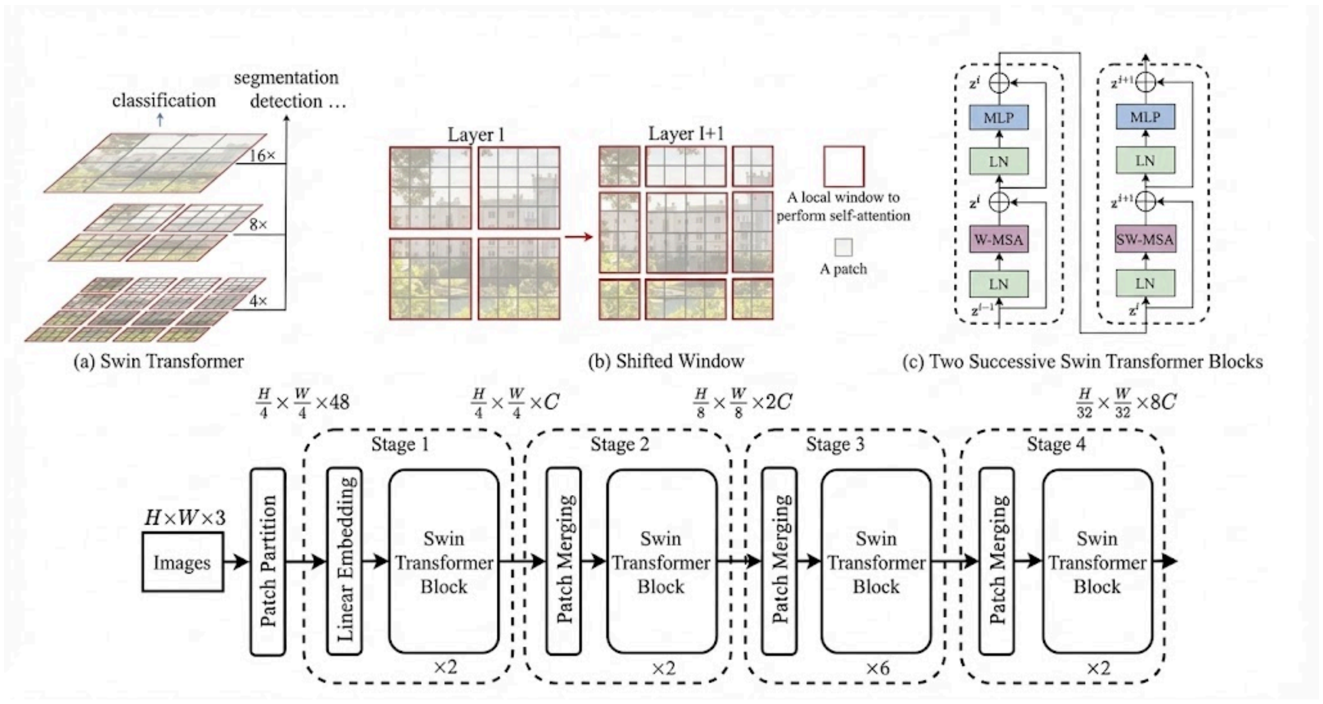
Swin Transformer (Shifted Windows Vision Transformer)
Motivation & Challenges in Vision
Adapting Transformer to Vision
ViT Limitations for Dense Prediction
Key Architectural Innovations
Hierarchical Feature Representation
Shifted Window Based Self-Attention (SW-MSA)
Position Encoding
Overall Architecture Flow (Swin-T)
Patch Splitting & Linear Embedding
Stage 1: Swin Transformer Blocks (Window-MSA)
Patch Merging Layers (Stages 2, 3, 4)
Swin Transformer Blocks (Shifted Window-MSA)
Performance and Applicability
General-purpose backbone for computer vision (like CNNs)
Image Classification (ImageNet-1K)
Object Detection (COCO)
Semantic Segmentation (ADE20K)
Architecture Variants (Swin-T, S, B, L)

Summary

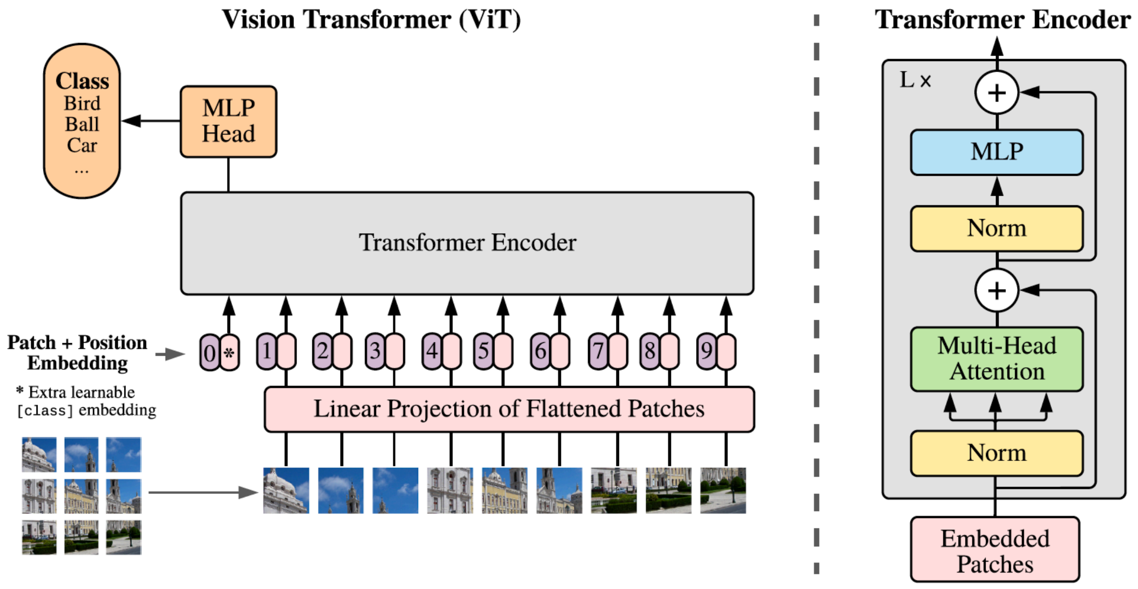
The Swin Transformer modifies the standard Transformer architecture (originally for text) to work efficiently on images by using a hierarchical structure and "shifted windows" to handle different object sizes and reduce computational costs.

Definitions

- **Swin Transformer (Shifted Windows Vision Transformer):** A deep learning model designed to serve as a "backbone" for computer vision tasks (like recognizing objects or separating parts of an image), aiming to replace traditional Convolutional Neural Networks (CNNs).



- **Vision Transformer (ViT):** The predecessor to Swin. It applied Transformers to images but struggled with high-resolution images because it treated the whole image as one flat sequence.

- 

- **Hierarchical Representation:** A method of processing data where the model starts looking at very small details (pixels) and gradually "zooms out" to understand larger patterns and objects.
  - **Real World Example:** Imagine looking at a forest. First, you see individual leaves (low level). Then you group them into branches, then trees, and finally, you see the whole forest (high level). Swin does this step-by-step, whereas previous Transformers tried to see everything at once.
- **Quadratic Computational Complexity:** A mathematical term meaning that if you double the size of the image, the work required doesn't just double—it quadruples (or worse). This makes processing big images impossible for standard Transformers.

- **Dense Prediction:** Computer vision tasks that require a label for every single pixel, such as Semantic Segmentation.
    - **Real World Example:** In a self-driving car camera feed, the computer needs to paint every pixel belonging to "road" gray, every pixel for "pedestrian" red, and "sky" blue. This requires very high resolution and detail.

Methodology

The Swin Transformer solves two main problems found in standard Transformers: **Scale Variation** (objects being different sizes) and **High Resolution** (too many pixels to count).

1. The Hierarchical Approach

- **Patching:** The model starts by breaking the image into tiny 4x4 pixel squares (patches).
- **Merging:** As the data moves deeper into the network, the model merges neighboring patches together.
- **Result:** This creates a pyramid-like structure. It starts with high resolution (good for details) and moves to low resolution (good for big objects). This mimics how successful CNNs (like ResNet) work, making Swin compatible with existing tools like Feature Pyramid Networks (FPN).

2. The Shifted Window Scheme

- **Window-Based Attention:** Instead of comparing every pixel to every other pixel (Global Attention), Swin divides the image into fixed-size windows. It only calculates relationships *inside* each window.
    - *Benefit:* This changes the math from Quadratic (very slow) to Linear (fast).
- **The "Shift":** If we only look inside fixed windows, the windows can't talk to each other. To fix this, Swin shifts the grid of windows slightly between layers.
    - **Layer 1:** Standard grid.
    - **Layer 2:** The grid moves (shifts) down and to the right.
- **Result:** This shifting allows information to cross the boundaries of the windows, connecting the whole image together without the heavy cost of Global Attention.

Results & Conclusion

- **Efficiency:** The Swin Transformer successfully reduces computational complexity from quadratic to linear. This makes it possible to use Transformers on high-resolution images (like Full HD) without running out of memory.
- **Versatility:** Unlike the original ViT, which was mostly good for simple classification, Swin works excellent for complex "dense" tasks like **Object Detection** and **Semantic**

Segmentation.
- **Performance:** It proves that Transformers can perform as well as, or better than, state-of-the-art CNNs as a general-purpose backbone for computer vision.

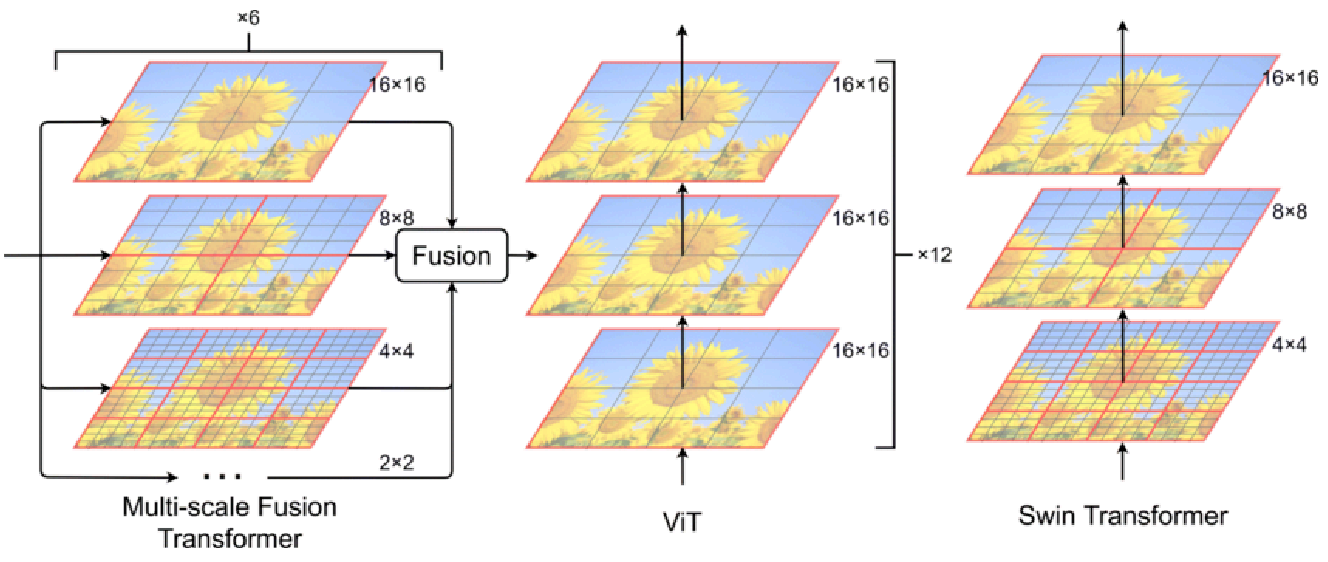
Architectural Innovations

To make Transformers work efficiently for computer vision, Swin introduces a hierarchical structure (merging pixels to understand scale) and a shifted window mechanism (limiting attention to local areas) to achieve linear computational speed while maintaining global context.

- **Hierarchical Feature Maps:** A structural design where the model produces feature maps at different resolutions. It starts with fine details and moves to coarse, abstract concepts.
  - **Significance:** This mimics traditional Convolutional Neural Networks (CNNs) like VGG or ResNet, making Swin compatible with existing detection tools like Feature Pyramid Networks (FPN).
- **Patch Merging:** The mechanism used to create the hierarchy. Deep inside the network, neighboring image patches are grouped together to reduce the resolution and increase the depth of information.
- **Linear vs. Quadratic Complexity:**
  - *Quadratic (Old ViT):* If you double the image size, the work quadruples. (Bad for high-res).
  - *Linear (Swin):* If you double the image size, the work only doubles. (Good for high-res).
- **Relative Position Bias:** A learned parameter added to the attention calculation that tells the model where pixels are *relative to each other* (e.g., "left of," "above") rather than their absolute x,y coordinates.

Methodology / How it Works

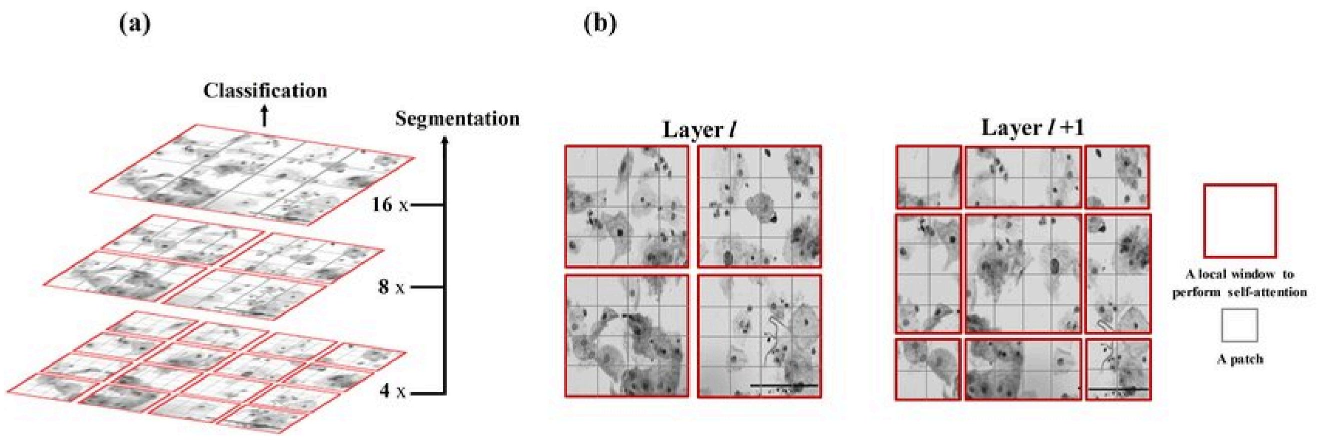
The Swin Transformer reshapes the standard Transformer architecture through three specific innovations:



1. Creating Hierarchy (Addressing Scale)

- **Input:** The image is split into small 4x4 pixel patches.
- **Merging:** As the data moves through the 4 stages of the network, "Patch Merging" layers combine 2×2 groups of neighboring patches.
- **Outcome:** This downsamples the resolution by 2x at each stage while doubling the feature dimension. This allows the model to detect small objects (early stages) and large objects (later stages).

2. Shifted Window Attention (Addressing Efficiency)



- **Local Windows:** Instead of looking at the whole image at once, Swin divides the image into non-overlapping windows (default size $8M = 7$). Attention is calculated *only* inside these windows.
- **The Shift:** To prevent these windows from becoming isolated islands of information, the model alternates configurations:
  - *Layer 1:* Regular grid partition.
  - *Layer 2:* The grid is shifted (displaced), causing new windows to straddle the boundaries of the old windows.
- **Cyclic Shifting:** To make this shift fast on computer hardware, the authors use "cyclic shifting"—moving parts of the image from the top-left to the bottom-right mathematically—rather than using slow padding methods.

3. Relative Position Encoding

- Instead of assigning a fixed number to every pixel (Absolute Embedding), Swin adds a bias term (13B) to the attention formula:

$$Attention(Q, K, V) = SoftMax(QK^T/d + B)V$$

- This improves the model's ability to recognize objects regardless of where they appear in the image (translation invariance).

Key Results & Conclusion

- **Speed:** The Window-Based approach successfully reduces computational complexity from quadratic 15$\Omega(MSA)$ to linear 16$\Omega(W - MSA)$. This makes processing high-resolution images (like 1920x1080) feasible.
- **Connectivity:** The "Shifted" mechanism is proven to be effective. It allows information to cross window boundaries, ensuring the model understands the global context of the image despite only looking at local windows.
- **Versatility:** Because of these innovations, Swin is not just an image classifier; it serves as a powerful "backbone" for complex tasks like **Object Detection** and **Semantic Segmentation**, outperforming previous methods.

Applicability & Performance Metrics

The Swin Transformer establishes itself as a superior general-purpose vision backbone by outperforming state-of-the-art CNNs and previous Transformers (ViT/DeiT) in image classification, object detection, and semantic segmentation through its scalable, hierarchical design.

- **General-Purpose Backbone:** A foundational model architecture that can be reused across many different types of computer vision tasks, rather than being specialized for just one.
- **Dense Prediction Tasks:** Computer vision tasks that require making a decision for *every single pixel* in an image, rather than just one label for the whole image.
- **Ablation Study:** A research method where parts of the model are removed or changed one by one to see how much each part contributes to the final performance.
- **Translation Invariance:** The ability of a model to recognize an object (like a cat) regardless of where it is shifted (translated) in the image (top-left vs. bottom-right). Swin achieves this via Relative Position Bias.
- The Swin Transformer was built to bridge the gap between language Transformers and visual requirements.

Results & Conclusion

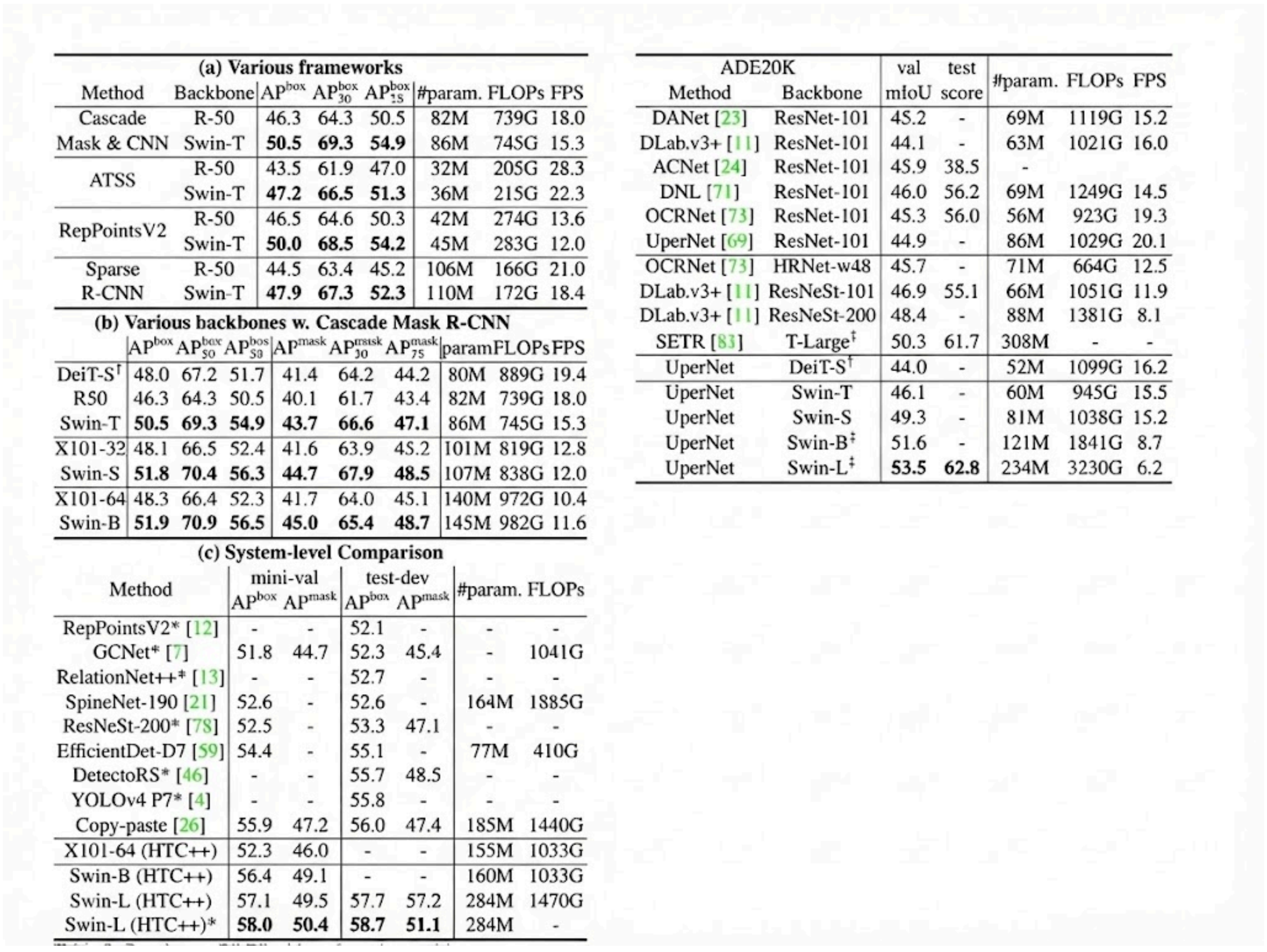
The Swin Transformer (specifically Swin-L) achieves State-of-the-Art (SOTA) results, proving it is better than both traditional CNNs and previous Transformers.

2. Comparison vs. Previous Transformers (ViT / DeiT)

- **Weakness of ViT:** ViT uses Global Attention (Quadratic complexity), making it too slow and heavy for high-resolution dense tasks.
- **Strength of Swin:** Swin matches ViT/DeiT in classification but **dominates** in detection and segmentation because of its hierarchical structure.
  - *Stat:* Swin-S is **+5.3 mIoU** higher than DeiT-S in segmentation.

3. Comparison vs. CNNs (ResNet)

- Swin outperforms the best Convolutional Networks (like ResNeXt and EfficientNet) while maintaining similar speed/latency.
  - *Stat:* Swin-T provides a consistent **+3.4 to +4.2 box AP** gain over ResNet-50.



(a) Various frameworks

Method	Backbone	AP^{box}	AP^{box}_{30}	AP^{box}_{75}	#param.	FLOPs	FPS
Cascade	R-50	46.3	64.3	50.5	82M	739G	18.0
Mask & CNN	Swin-T	**50.5**	**69.3**	**54.9**	86M	745G	15.3
ATSS	R-50	43.5	61.9	47.0	32M	205G	28.3
	Swin-T	**47.2**	**66.5**	**51.3**	36M	215G	22.3
RepPointsV2	R-50	46.5	64.6	50.3	42M	274G	13.6
	Swin-T	**50.0**	**68.5**	**54.2**	45M	283G	12.0
Sparse	R-50	44.5	63.4	45.2	106M	166G	21.0
R-CNN	Swin-T	**47.9**	**67.3**	**52.3**	110M	172G	18.4

(b) Various backbones w. Cascade Mask R-CNN

	AP^{box}	AP^{box}_{50}	AP^{box}_{50}	AP^{mask}	AP^{mask}_{30}	AP^{mask}_{75}	param	FLOPs	FPS
DeiT-S†	48.0	67.2	51.7	41.4	64.2	44.2	80M	889G	19.4
R50	46.3	64.3	50.5	40.1	61.7	43.4	82M	739G	18.0
Swin-T	**50.5**	**69.3**	**54.9**	**43.7**	**66.6**	**47.1**	86M	745G	15.3
X101-32	48.1	66.5	52.4	41.6	63.9	45.2	101M	819G	12.8
Swin-S	**51.8**	**70.4**	**56.3**	**44.7**	**67.9**	**48.5**	107M	838G	12.0
X101-64	48.3	66.4	52.3	41.7	64.0	45.1	140M	972G	10.4
Swin-B	**51.9**	**70.9**	**56.5**	**45.0**	**65.4**	**48.7**	145M	982G	11.6

(c) System-level Comparison

Method	mini-val		test-dev		#param.	FLOPs
	AP^{box}	AP^{mask}	AP^{box}	AP^{mask}		
RepPointsV2* [12]	-	-	52.1	-	-	-
GCNet* [7]	51.8	44.7	52.3	45.4	-	1041G
RelationNet++* [13]	-	-	52.7	-	-	-
SpineNet-190 [21]	52.6	-	52.6	-	164M	1885G
ResNeSt-200* [78]	52.5	-	53.3	47.1	-	-
EfficientDet-D7 [59]	54.4	-	55.1	-	77M	410G
DetectoRS* [46]	-	-	55.7	48.5	-	-
YOLOv4 P7* [4]	-	-	55.8	-	-	-
Copy-paste [26]	55.9	47.2	56.0	47.4	185M	1440G
X101-64 (HTC++)	52.3	46.0	-	-	155M	1033G
Swin-B (HTC++)	56.4	49.1	-	-	160M	1033G
Swin-L (HTC++)	57.1	49.5	57.7	57.2	284M	1470G
Swin-L (HTC++)*	**58.0**	**50.4**	**58.7**	**51.1**	284M	-

ADE20K		val	test	#param.	FLOPs	FPS
Method	Backbone	mIoU	score			
DANet [23]	ResNet-101	45.2	-	69M	1119G	15.2
DLab.v3+ [11]	ResNet-101	44.1	-	63M	1021G	16.0
ACNet [24]	ResNet-101	45.9	38.5	-		
DNL [71]	ResNet-101	46.0	56.2	69M	1249G	14.5
OCRNet [73]	ResNet-101	45.3	56.0	56M	923G	19.3
UperNet [69]	ResNet-101	44.9	-	86M	1029G	20.1
OCRNet [73]	HRNet-w48	45.7	-	71M	664G	12.5
DLab.v3+ [11]	ResNeSt-101	46.9	55.1	66M	1051G	11.9
DLab.v3+ [11]	ResNeSt-200	48.4	-	88M	1381G	8.1
SETR [83]	T-Large‡	50.3	61.7	308M	-	-
UperNet	DeiT-S†	44.0	-	52M	1099G	16.2
UperNet	Swin-T	46.1	-	60M	945G	15.5
UperNet	Swin-S	49.3	-	81M	1038G	15.2
UperNet	Swin-B‡	51.6	-	121M	1841G	8.7
UperNet	Swin-L‡	**53.5**	**62.8**	234M	3230G	6.2