Lecture 25: Recent architectures

Transformer slides from S. Lazebnik

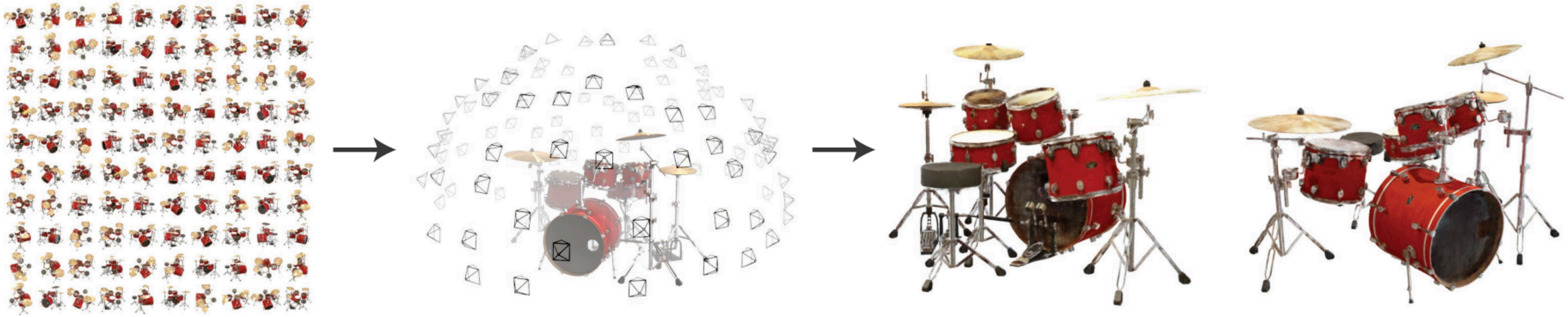
Reminders

- Sign up for final presentation (see Piazza)

Today

- Neural fields
- Transformers for vision

3D view synthesis



Input views

Create model

Render new views

What representation should we use?

[Source: Mildenhall et al., “NeRF”, 2020]

Idea #1: Image-based rendering

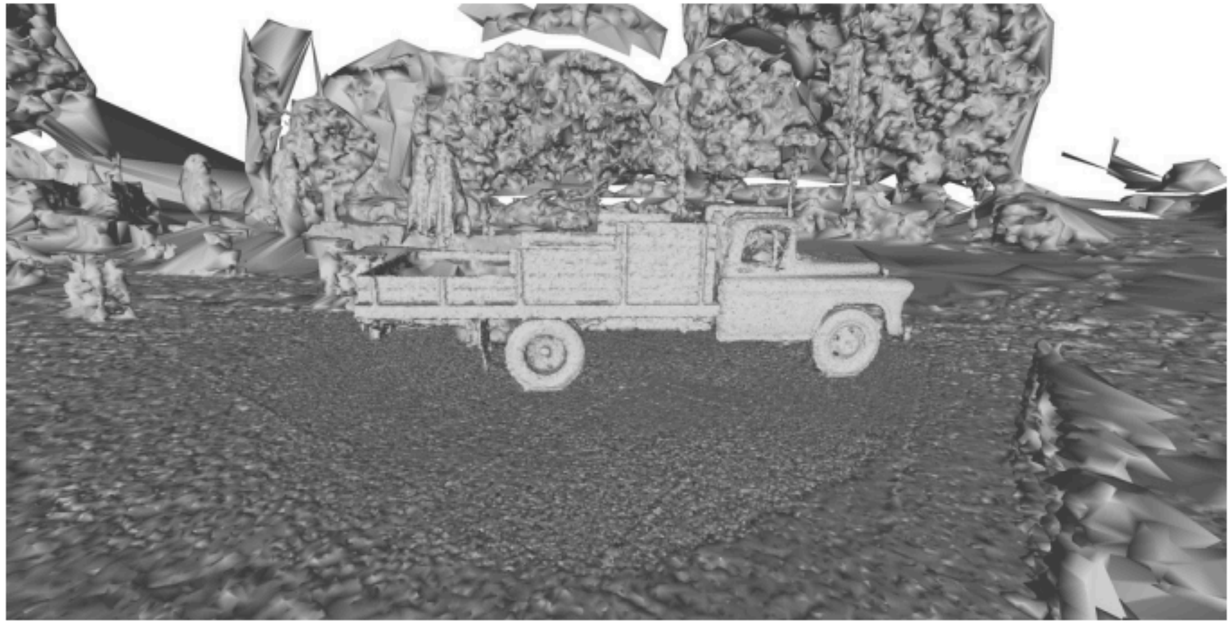
Point cloud
(reconstructed with SfM + multi-view stereo)

→ View from a different angle

[Riegler and Koltun, 2020]

Idea #1: Image-based rendering

Point cloud

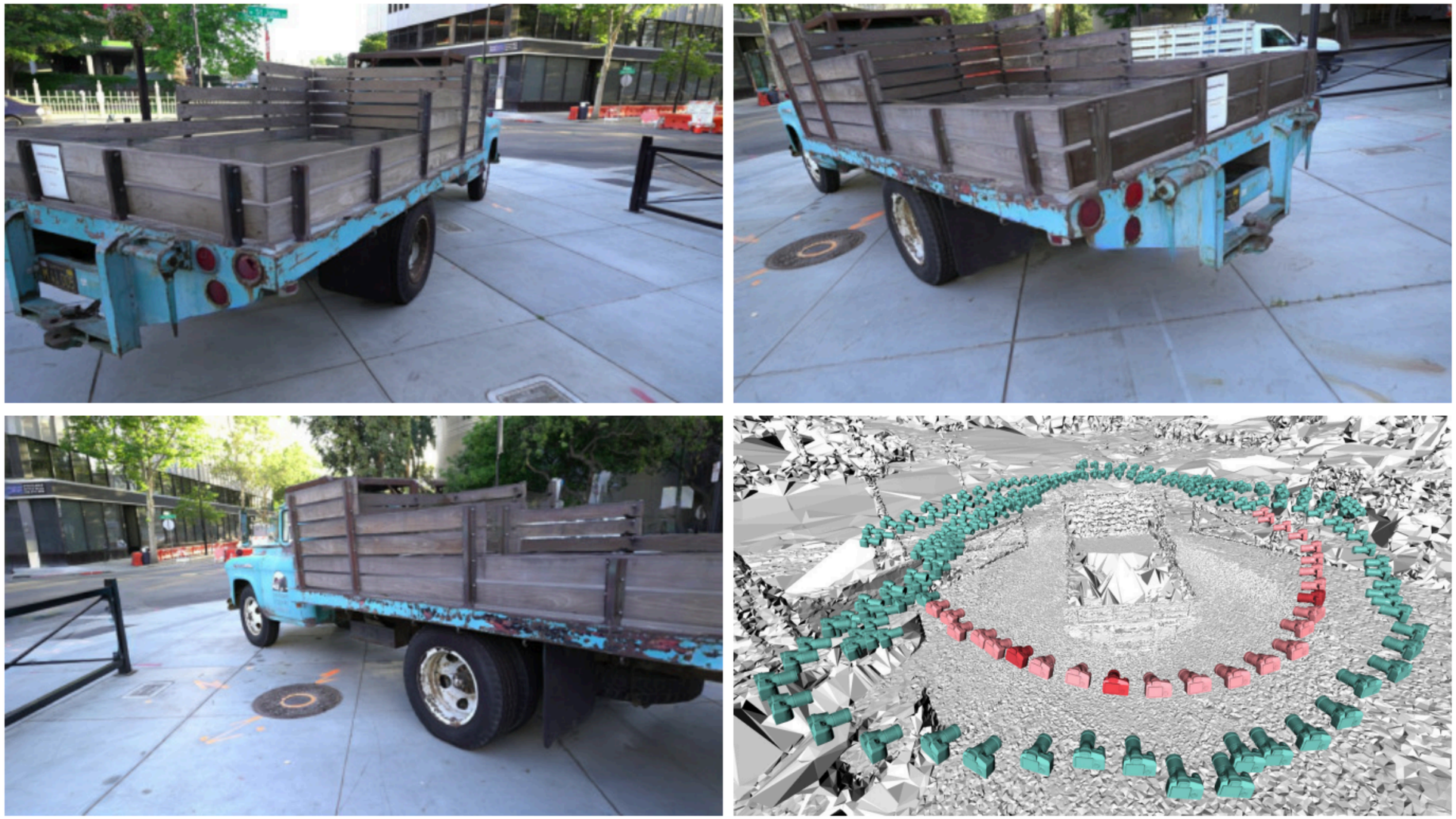


Proxy geometry (a mesh)

To synthesize a new view, select colors from existing views using proxy geometry.

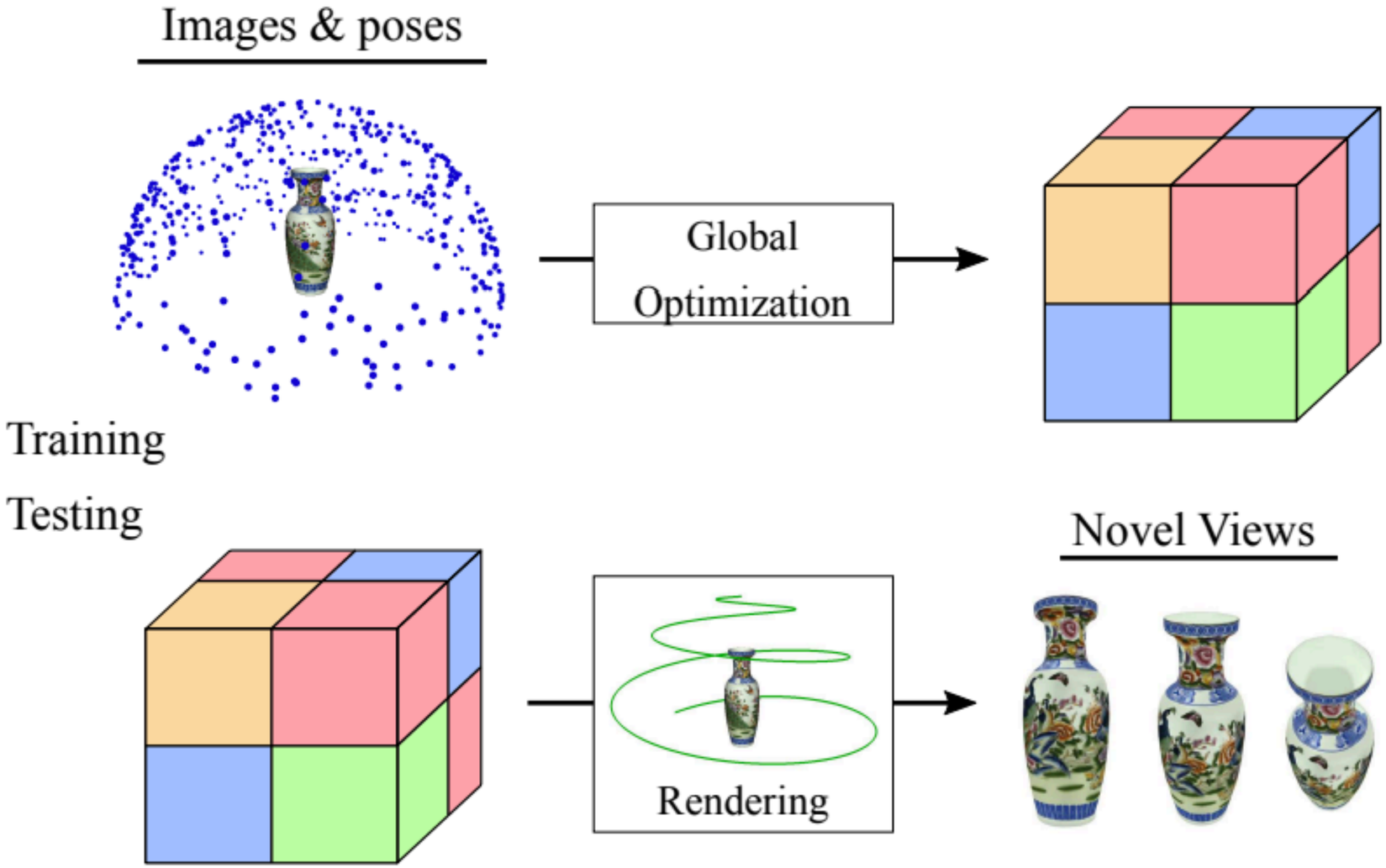
[Riegler and Koltun, 2020]

Idea #1: Image-based rendering



[Riegler and Koltun, 2020]

Idea #2: voxel representation



[Source: Sitzmann et al., “DeepVoxels”, 2019]

Idea #2: voxel representation

Input views

Position Viewing direction Color Density

$$V[x, y, z, \theta, \phi] = (R, G, B, \sigma)$$

Idea #2: voxel representation

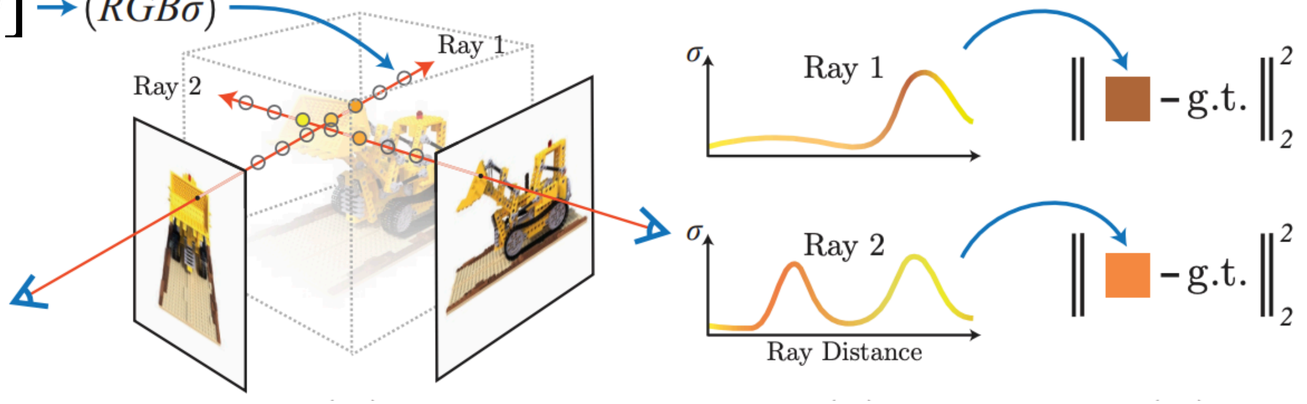
Input views

Position Viewing direction Color Density

$$V[x, y, z, \theta, \phi] = (R, G, B, \sigma)$$

Training:

$$V[x, y, z, \theta, \phi] \rightarrow (RGB\sigma)$$



Problem: A huge table! $\mathcal{O}(D^3A^2)$

Idea #3: neural radiance field (NeRF)

Input views

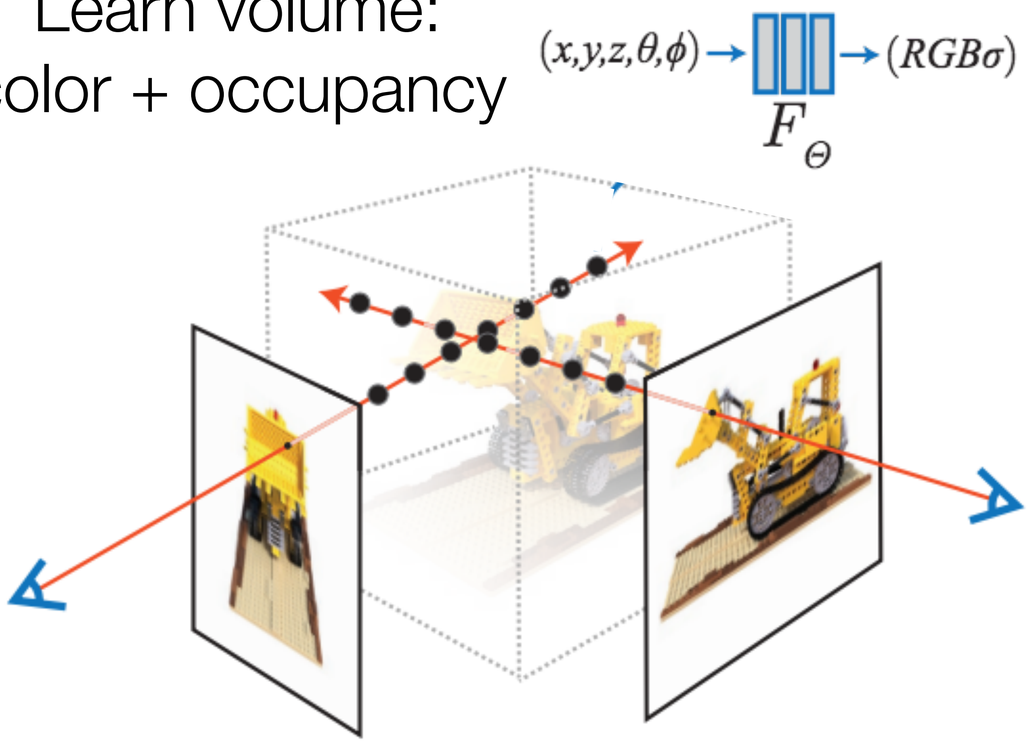
$$\rightarrow \quad F_\Theta(x, y, z, \theta, \phi) = (R, G, B, \sigma)$$

- Represent using a **neural radiance field**.
- Function that maps a (x, y, z, θ, ϕ) to a color and density.
- Typically parameterized as a multi-layer perceptron (MLP)
- Goal: find parameters Θ for MLP that explain the images

Idea #3: neural radiance field (NeRF)

3D scene

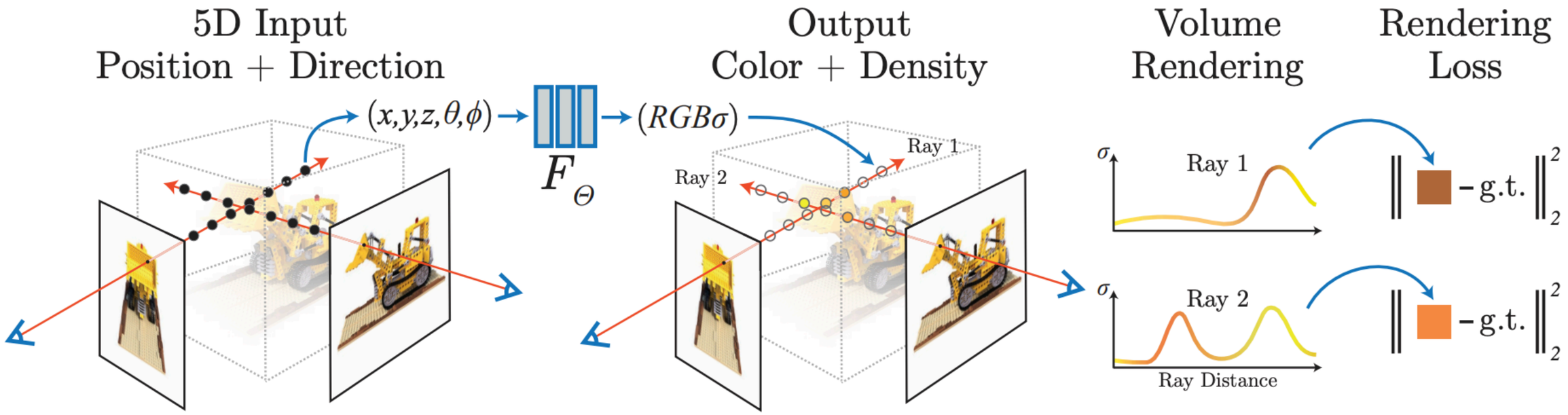
Learn volume:
color + occupancy



Viewpoints

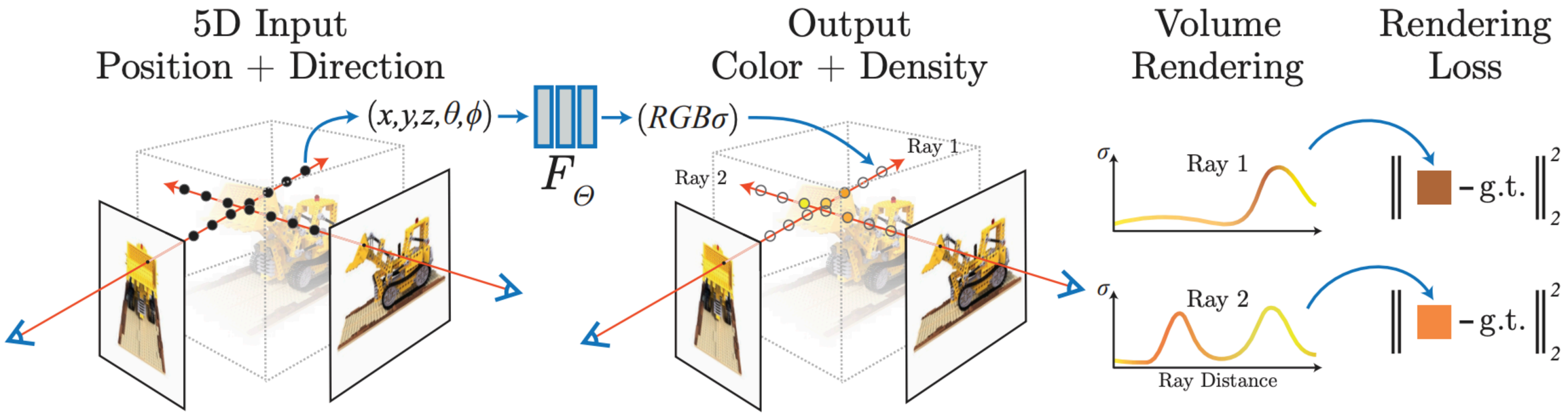
[Mildenhall*, Srinivasan*, Tanick*, et al., Neural radiance fields, 2020]

Learning a NeRF



[Source: Mildenhall et al., “NeRF”, 2020]

Neural rendering

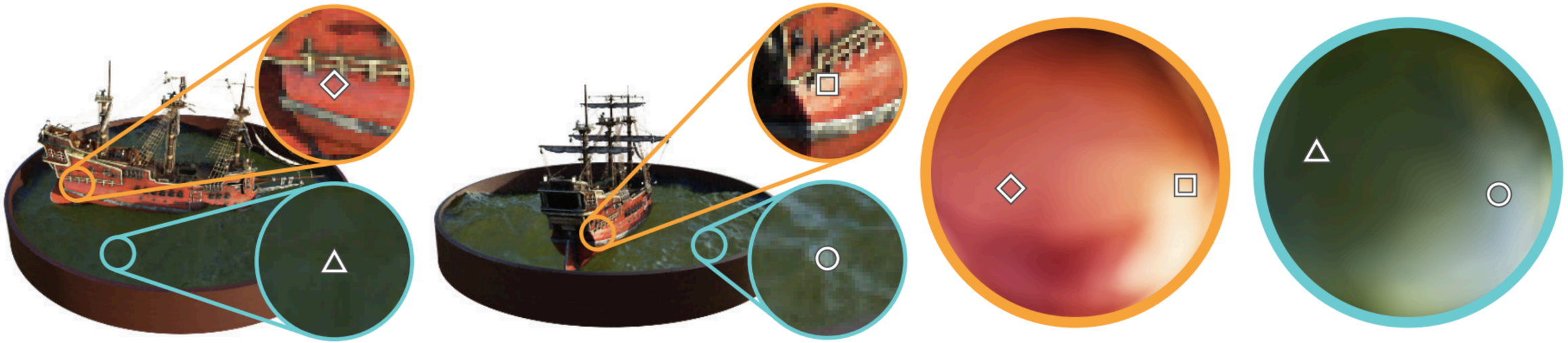


$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt\,, \text{ where } T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds\right)$$

Ray: $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$ For color c and density σ.

[Source: Mildenhall et al., “NeRF”, 2020]

Why is it good to be view-dependent?



(a) View 1 (b) View 2 (c) Radiance Distributions

[Source: Mildenhall et al., “NeRF”, 2020]

Representing the inputs

Input views

$$\rightarrow \quad F_\Theta(x, y, z, \theta, \phi) = (R, G, B, \sigma)$$

- In theory, could just plug in 4 inputs x, y, z, θ, ϕ
- However, this leads to blurry results.
- Neural nets show a bias toward low frequency functions.

Representing the inputs

Input views

$$\rightarrow \quad F_\Theta(x, y, z, \theta, \phi) = (R, G, B, \sigma)$$

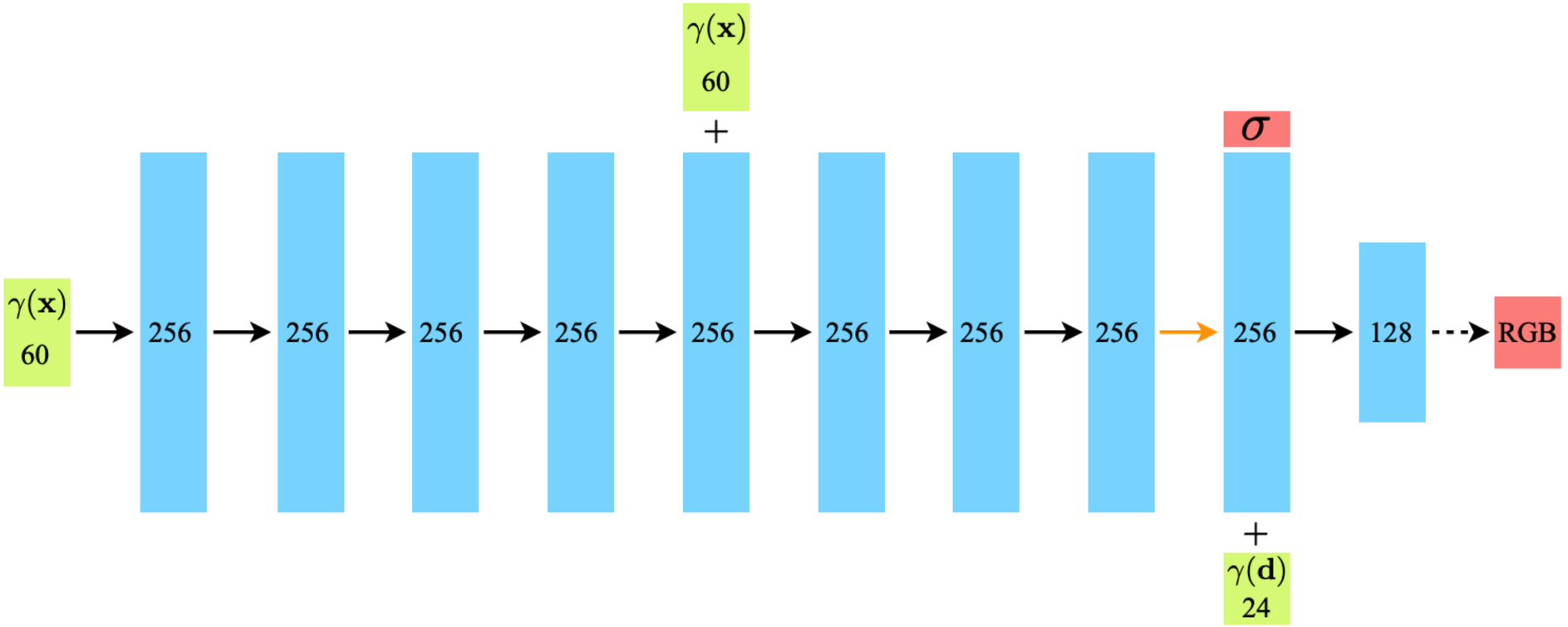
- Use a **positional encoding**. Given a scalar p, compute:

$$\gamma(p) = \left(\sin\left(2^0 \pi p\right), \cos\left(2^0 \pi p\right), \cdots, \sin\left(2^{L-1} \pi p\right), \cos\left(2^{L-1} \pi p\right)\right)$$

- Plug in the coordinate to sinusoids at different frequencies (e.g. L = 10). Creates a high-frequency input.

[Source: Mildenhall et al., “NeRF”, 2020]

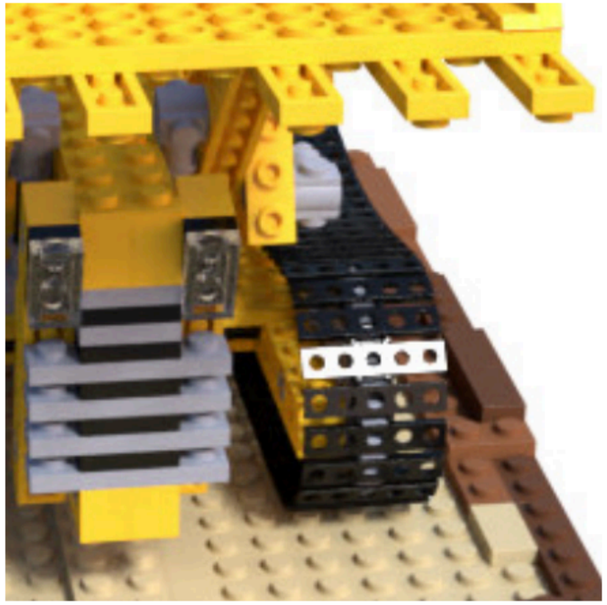
MLP architecture



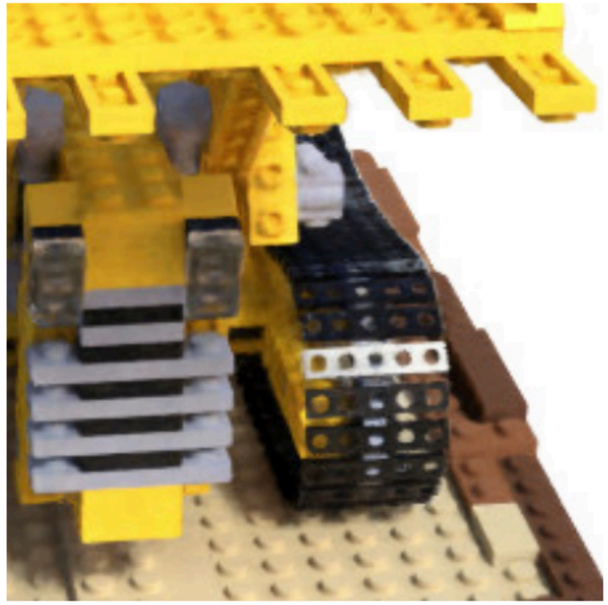
$$\gamma(p) = \left(\sin\left(2^0\pi p\right), \cos\left(2^0\pi p\right), \cdots, \sin\left(2^{L-1}\pi p\right), \cos\left(2^{L-1}\pi p\right)\right)$$

[Source: Mildenhall et al., “NeRF”, 2020]

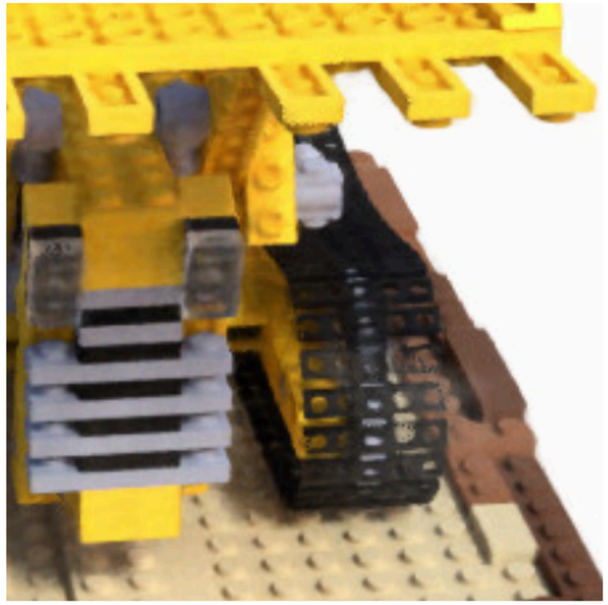
Results for a novel viewpoint



Ground Truth



Complete Model



No View Dependence

No Positional Encoding

[Source: Mildenhall et al., “NeRF”, 2020]

Results

[Mildenhall*, Srinivasan*, Tanick*, et al. 2020]

Results

[Mildenhall*, Srinivasan*, Tanick*, et al. 2020]

Results

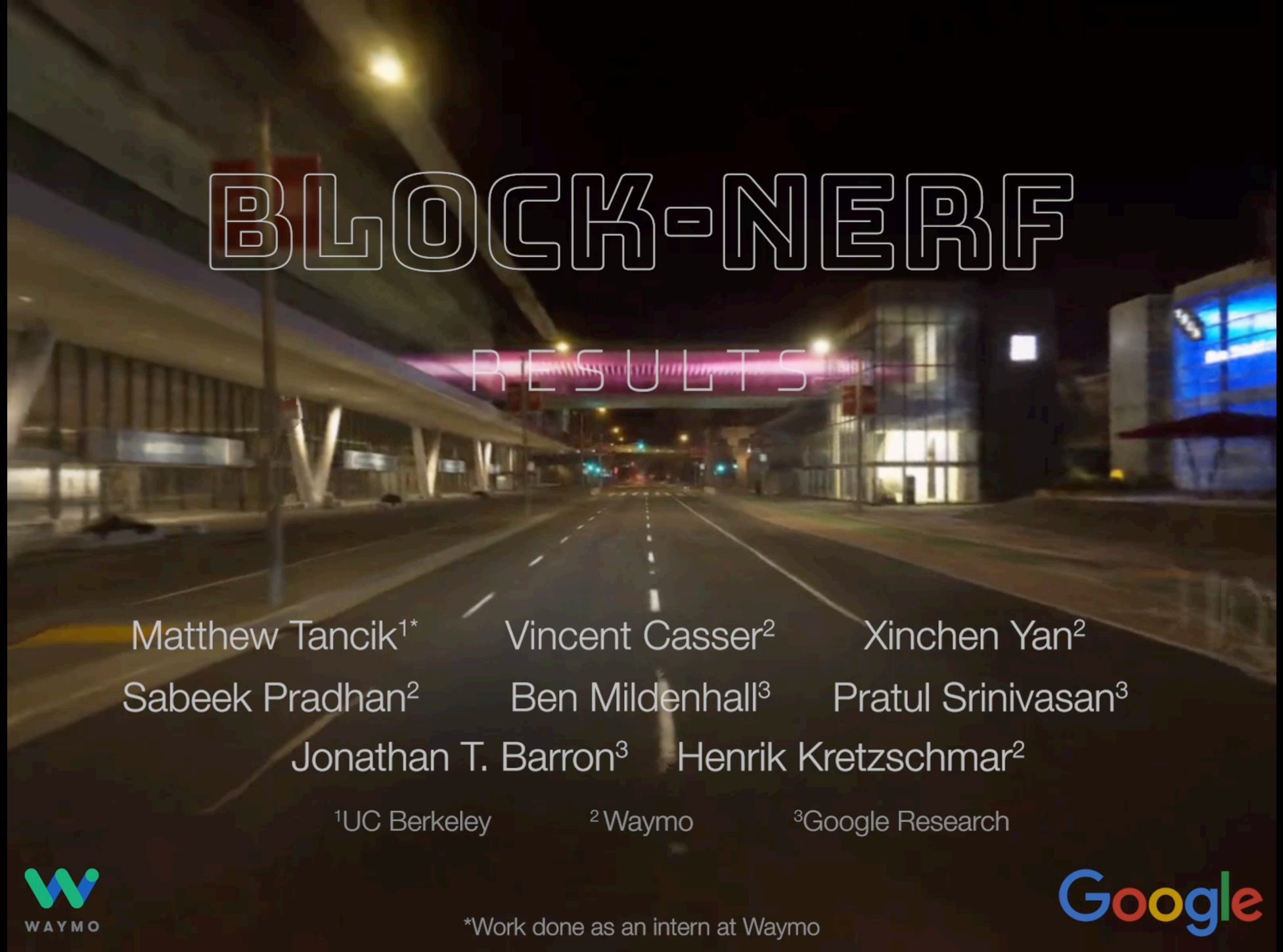
[Mildenhall*, Srinivasan*, Tanick*, et al. 2020]

Extension: internet photo collections

[Martin-Brualla, Radwan et al. “NeRF in the Wild”, 2020]

Extension: internet photo collections

[Martin-Brualla, Radwan et al. “NeRF in the Wild”, 2020]



Matthew Tancik[1*] Vincent Casser[2] Xinchen Yan[2]
Sabeek Pradhan[2] Ben Mildenhall[3] Pratul Srinivasan[3]
Jonathan T. Barron[3] Henrik Kretzschmar[2]

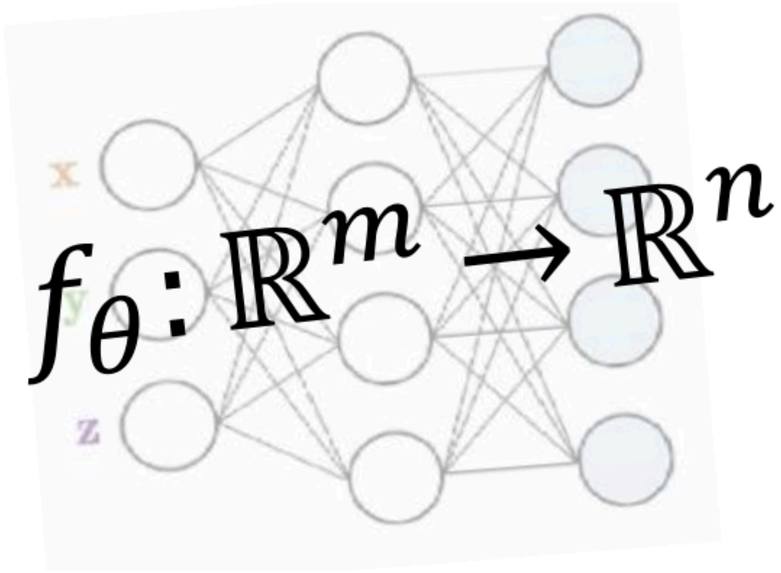
[1]UC Berkeley [2]Waymo [3]Google Research

WAYMO

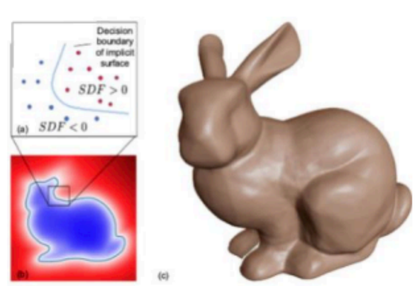
*Work done as an intern at Waymo

Google

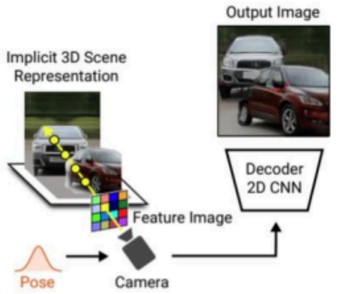
Lots of other applications



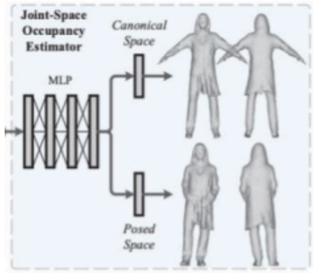
Neural field



2D and 3D Reconstruction

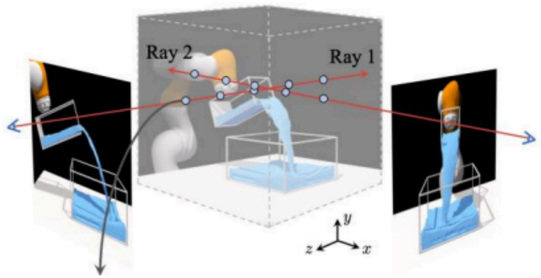


Generative Models

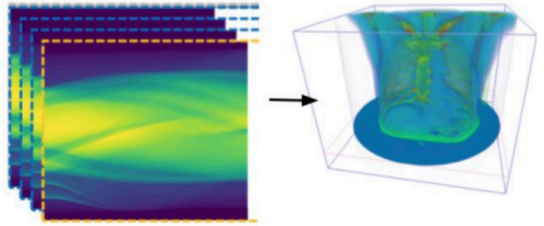


Digital Humans

Compression



Robotics



…and Beyond!

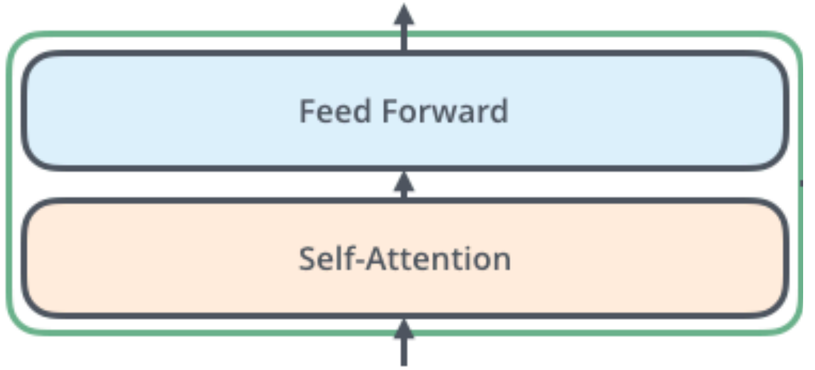
[Xie et al., “Neural Fields in Visual Computing and Beyond”, 2021]

Today

- Neural fields
- **Transformers for vision**

Recall: Transformers

- Build whole model out of self-attention
- Uses only point-wise processing and attention (no recurrent units or convolutions)



A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, NeurIPS 2017

Source: S. Lazebnik

Image source

Self-attention layer

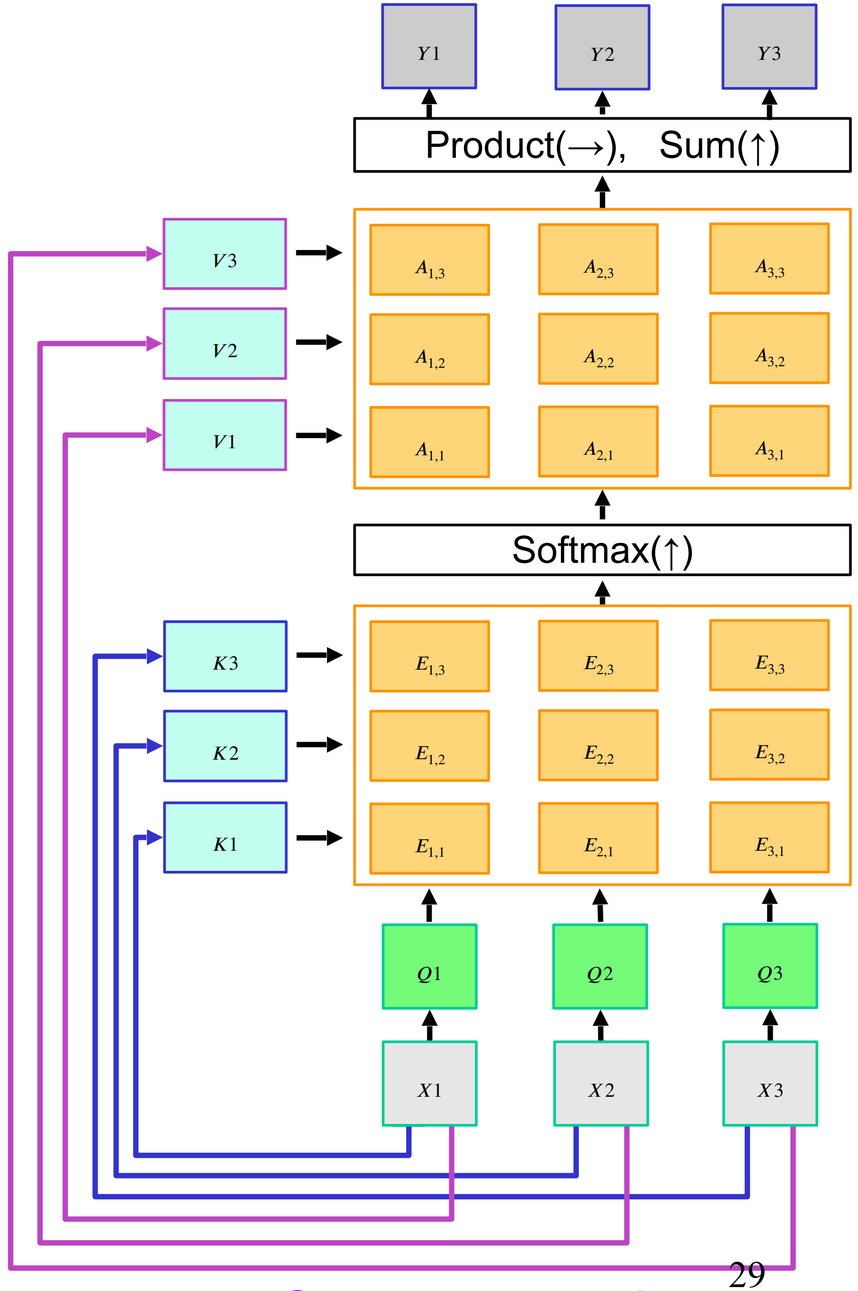
- Query vectors: $Q = XW_Q$
- Key vectors: $K = XW_K$
- Value vectors: $V = XW_V$
- Similarities: *scaled dot-product attention*

$$E_{i,j} = \frac{(Q_i \cdot Kj)}{\sqrt{D}} \text{ or } E = QK^T/\sqrt{D}$$

(D is the dimensionality of the keys)

- Attn. weights: $A = \text{softmax}(E, \text{dim} = 1)$
- Output vectors:

$$Y_i = \sum_j A_{i,j} V_j \text{ or } Y = AV$$

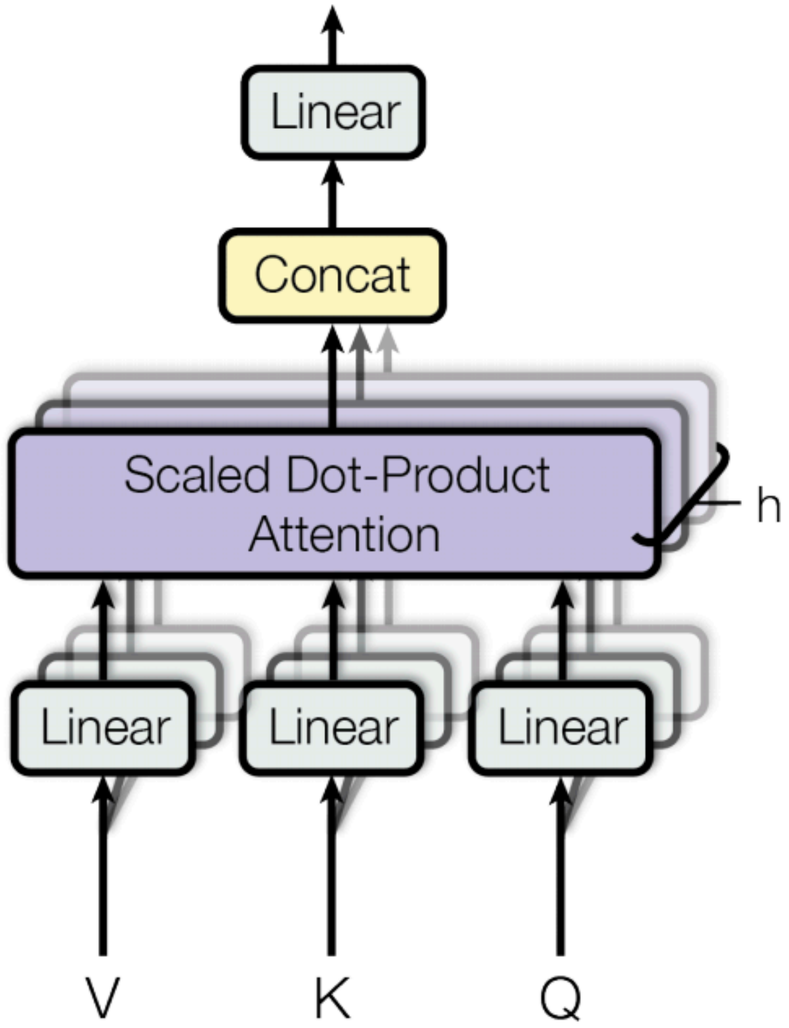


One query per input vector

Adapted from J. Johnson and S. Lazebnik.

Multi-head attention

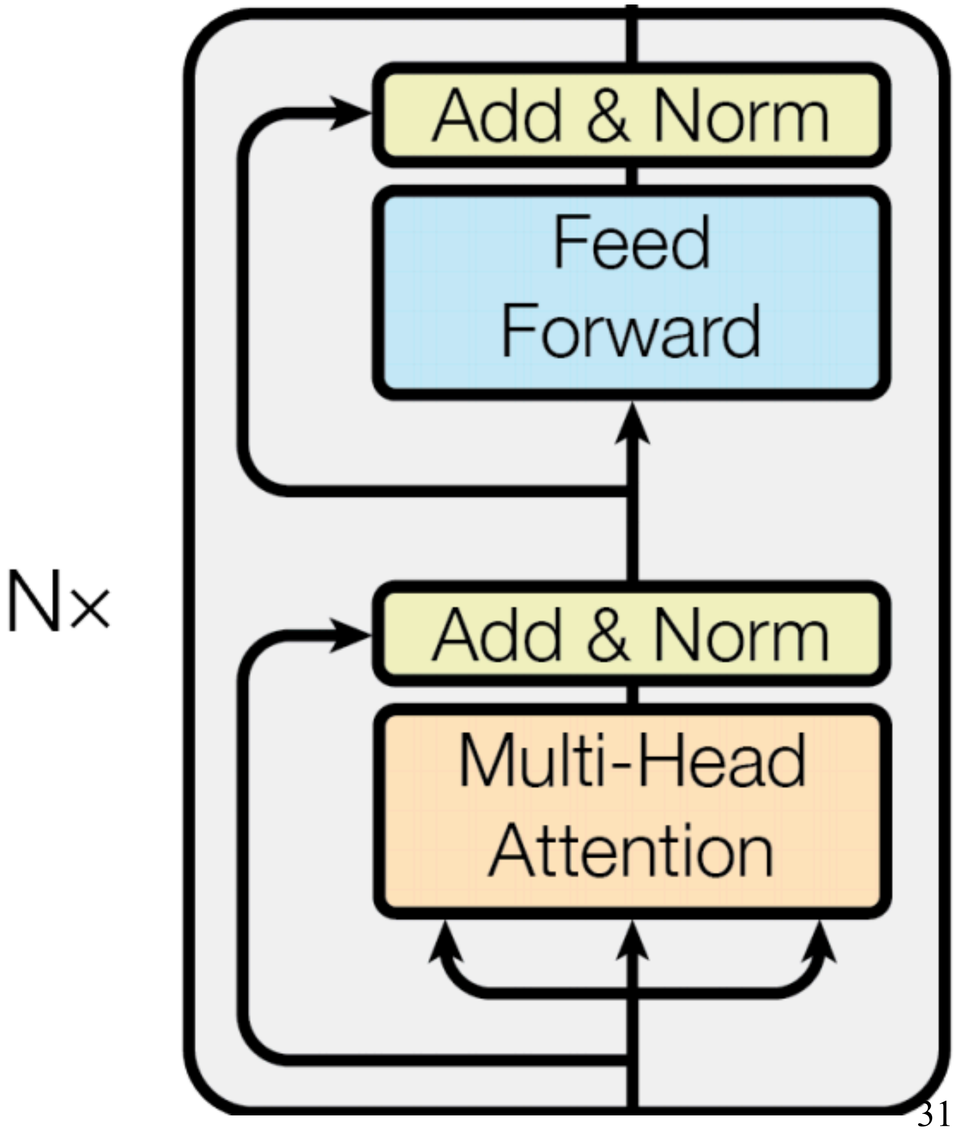
- Run h attention models in parallel on top of different linearly projected versions of Q, K, V; concatenate and linearly project the results
- Intuition: enables model to attend to different kinds of information at different positions



Source: S. Lazebnik

Transformer blocks

- A **Transformer** is a sequence of transformer blocks
  - Vaswani et al.: N=12 blocks, embedding dimension = 512, 6 attention heads
  - **Add & Norm:** residual connection followed by layer normalization
  - **Feedforward:** two linear layers with ReLUs in between, applied independently to each vector
- Attention is the only interaction between inputs!



Source: S. Lazebnik

Self-supervised learning in Natural Language Processing

1. Download A LOT of text from the internet
2. Train a giant transformer using a suitable pretext task
3. Fine-tune the transformer on desired NLP task

Model Alias	Org.	Article Reference
ULMfit	fast.ai	*Universal Language Model Fine-tuning for Text Classification* Howard and Ruder
 ELMo	AllenNLP	*Deep contextualized word representations* Peters et al.
OpenAI GPT	OpenAI	*Improving Language Understanding by Generative Pre-Training* Radford et al.
 BERT	Google	*BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding* Devlin et al.
XLM	Facebook	*Cross-lingual Language Model Pretraining* Lample and Conneau

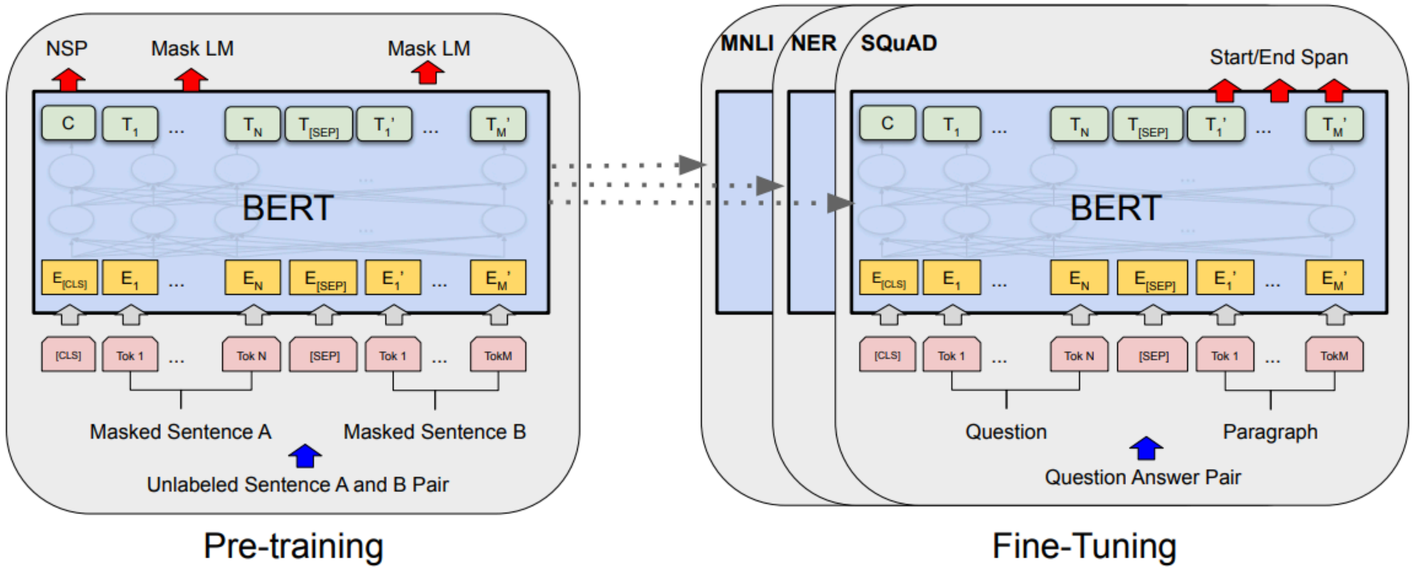
Source: S. Lazebnik

Image source

Self-supervised language modeling with transformers

1. Download A LOT of text from the internet
2. Train a giant transformer using a suitable pretext task
3. Fine-tune the transformer on desired NLP task

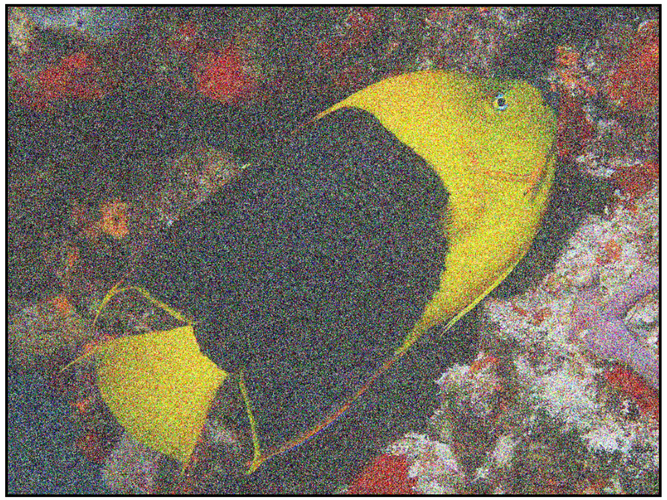
Bidirectional Encoder Representations from Transformers (BERT)



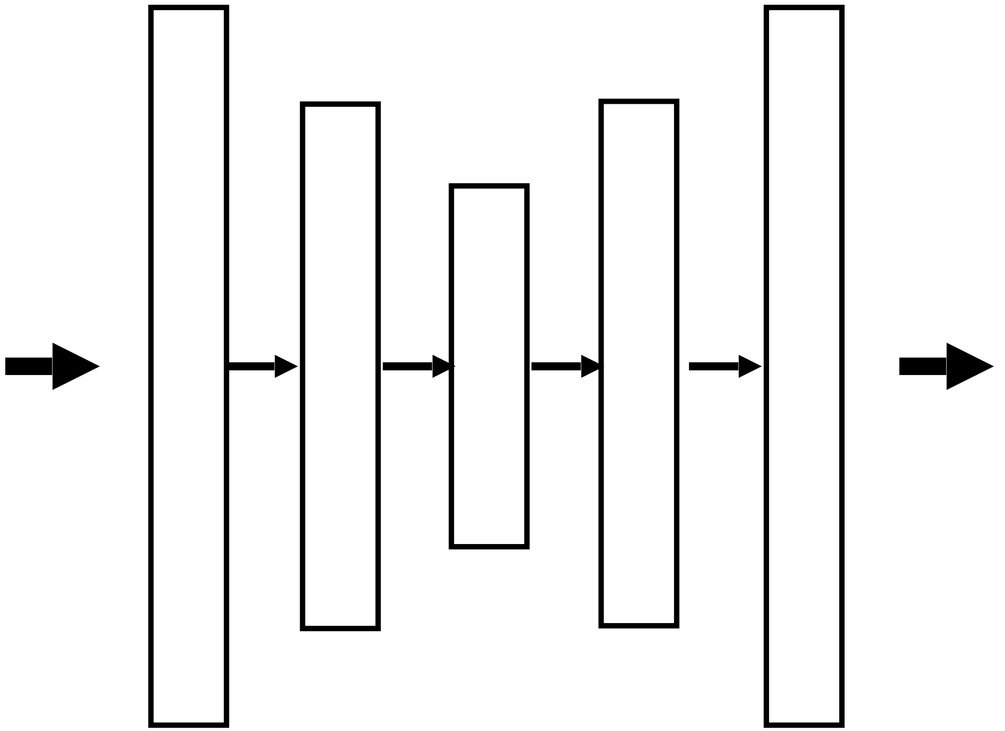
J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, EMNLP 2018

Source: S. Lazebnik

Recall: denoising autoencoder



Noisy image

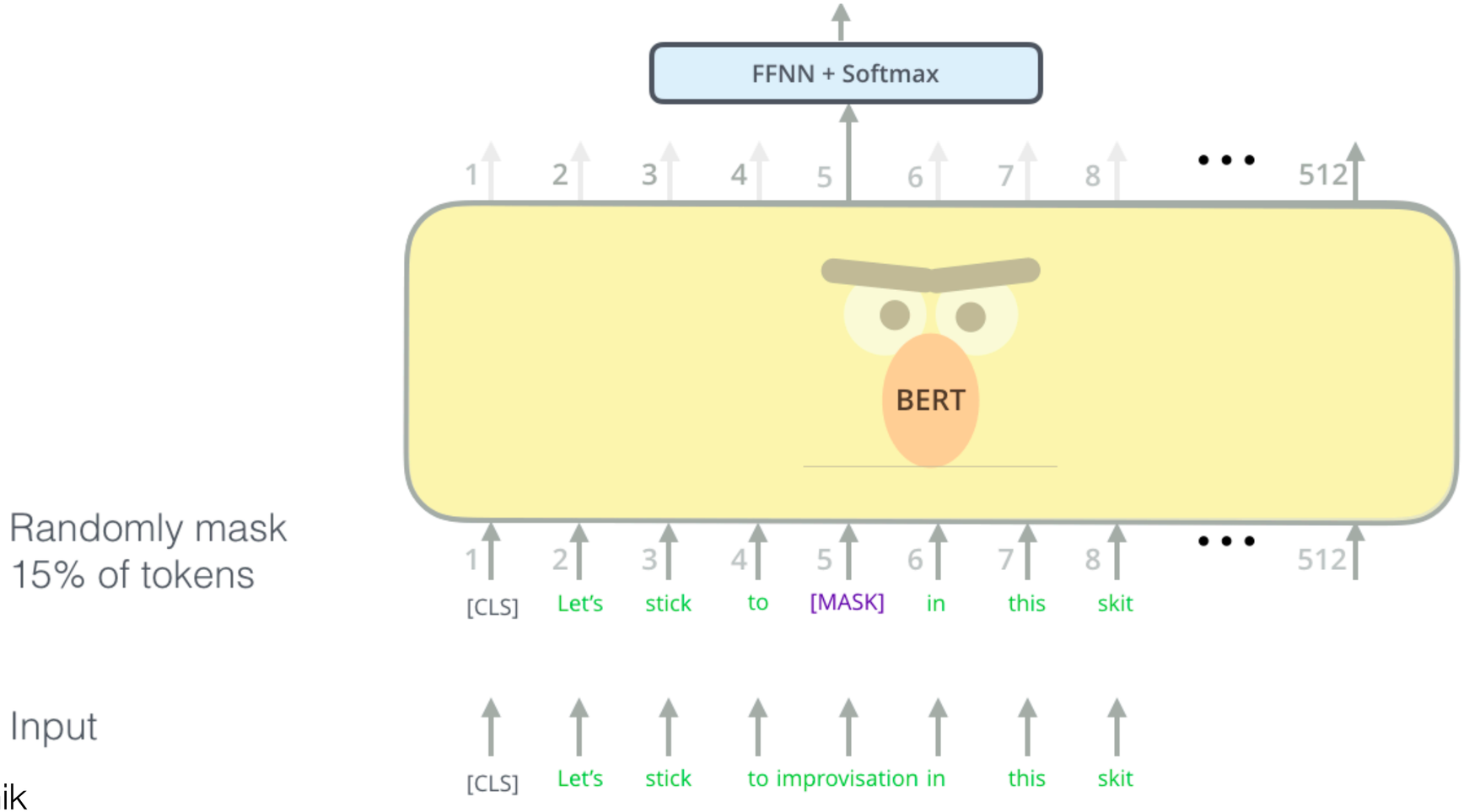


Reconstructed image

[Vincent et al., 2008]

BERT: Pretext tasks

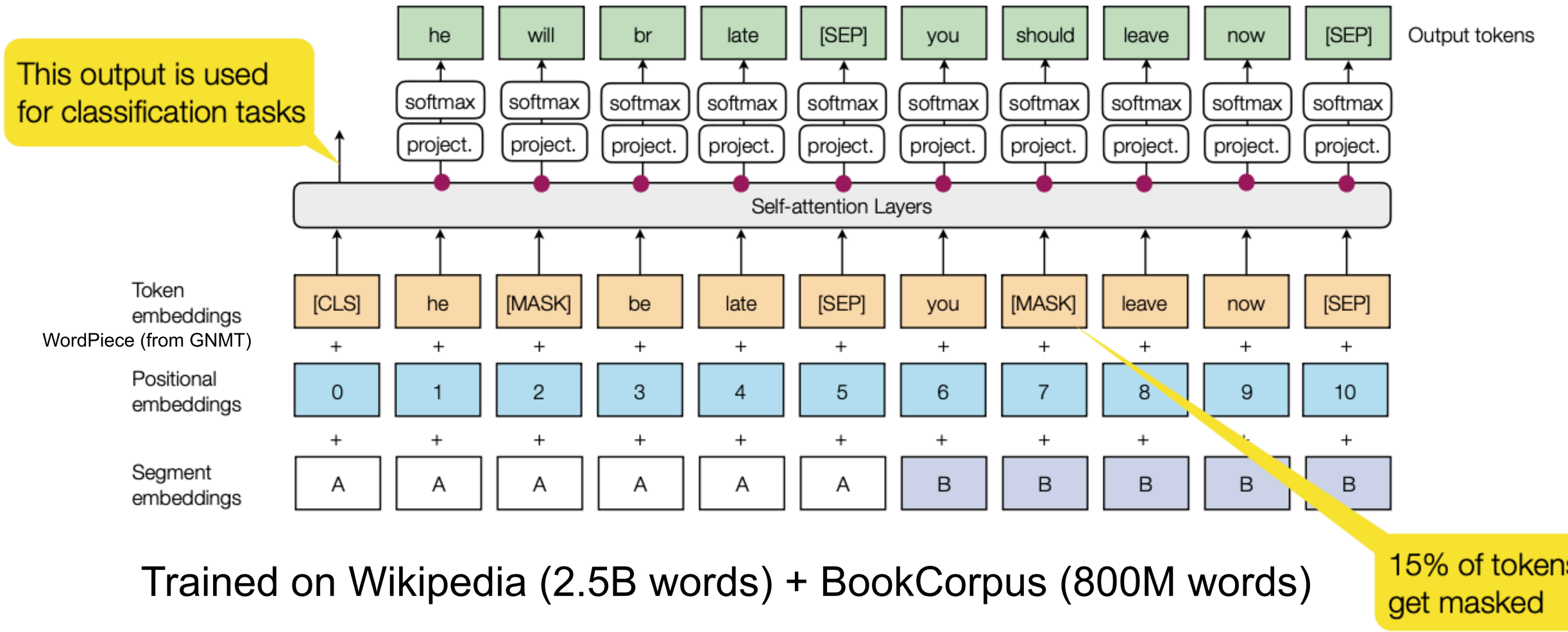
- Masked language model (MLM)
  - Randomly mask 15% of tokens in input sentences, goal is to reconstruct them using bidirectional context



Source: S. Lazebnik

Image source

BERT: More detailed view



Trained on Wikipedia (2.5B words) + BookCorpus (800M words)

Source: S. Lazebnik

Image source

BERT: Evaluation

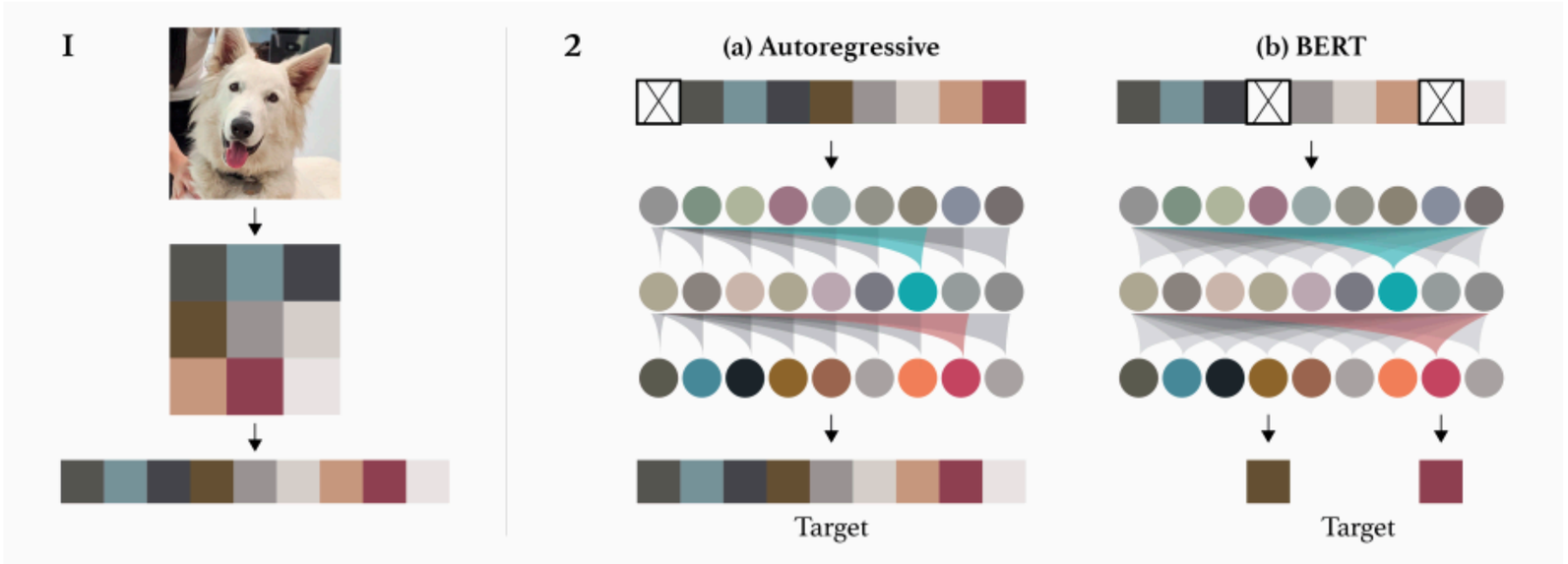
- General Language Understanding Evaluation (GLUE) benchmark (gluebenchmark.com)

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Source: S. Lazebnik

Image GPT

- Image resolution up to 64x64, color values quantized to 512 levels (9 bits), dense attention
- For transfer learning, average-pool encoded features across all positions



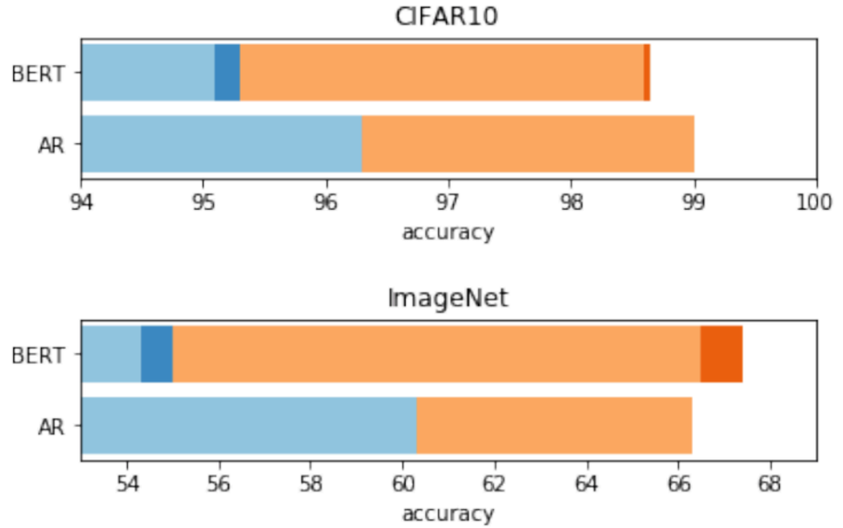
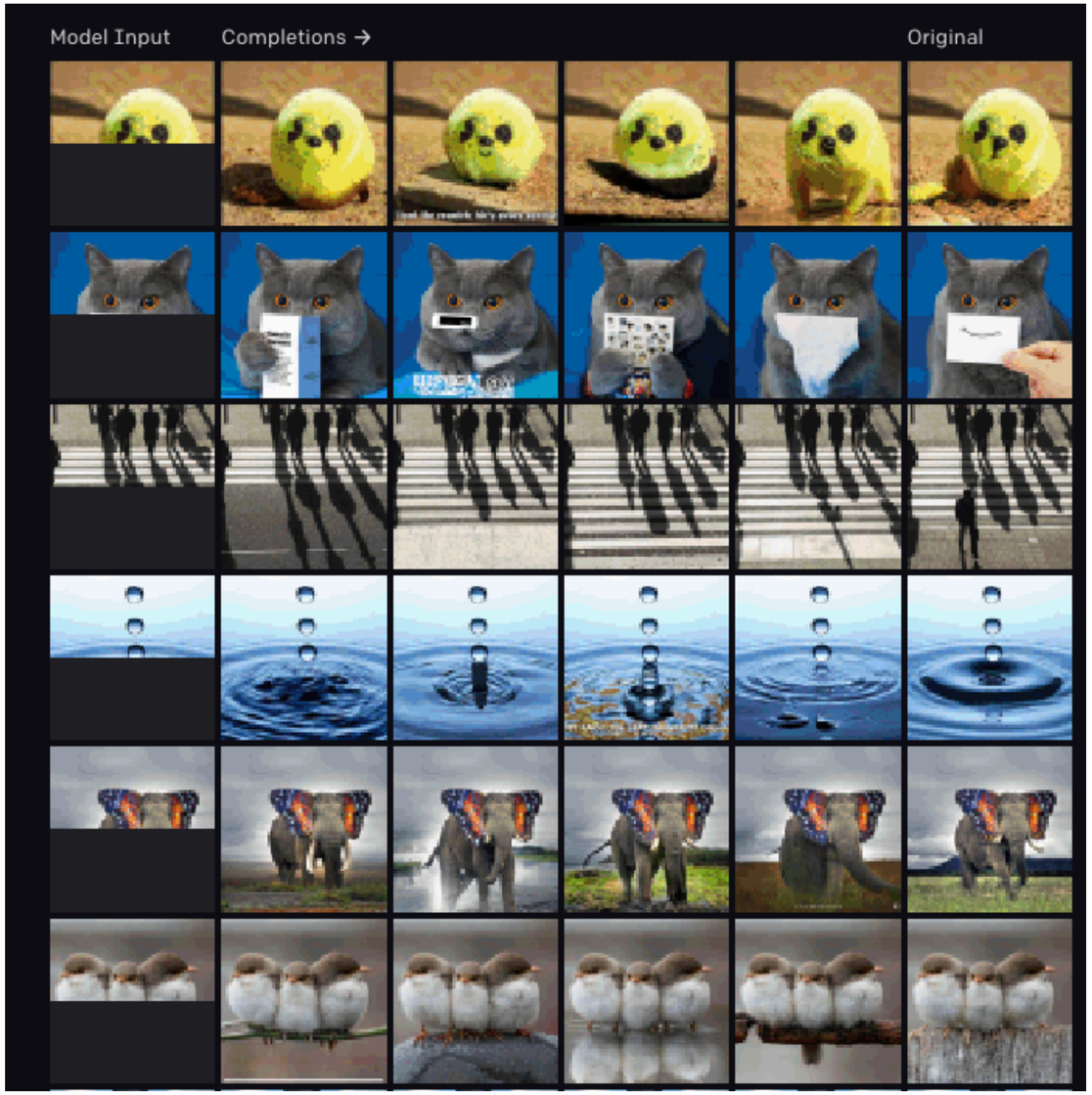


Figure 4. Comparison of auto-regressive pre-training with BERT pre-training using iGPT-L at an input resolution of $32^2 \times 3$. Blue bars display linear probe accuracy and orange bars display fine-tune accuracy. Bold colors show the performance boost from ensembling BERT masks. We see that auto-regressive models produce much better features than BERT models after pre-training, but BERT models catch up after fine-tuning.

M. Chen et al., Generative pretraining from pixels, ICML 2020

Source: S. Lazebnik

Image GPT – OpenAI



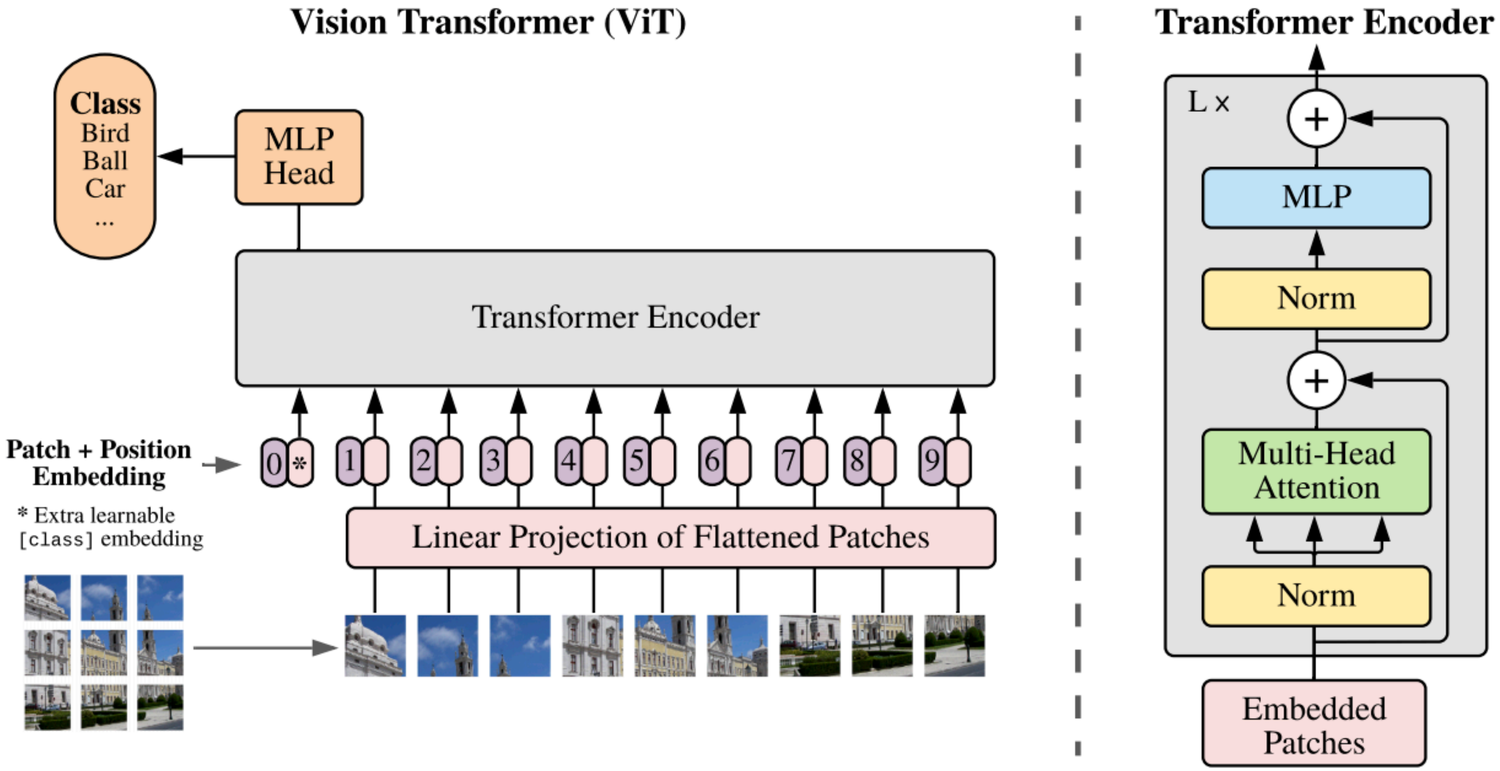
https://openai.com/blog/image-gpt/

M. Chen et al., Generative pretraining from pixels, ICML 2020

Source: S. Lazebnik

Vision transformer (ViT)

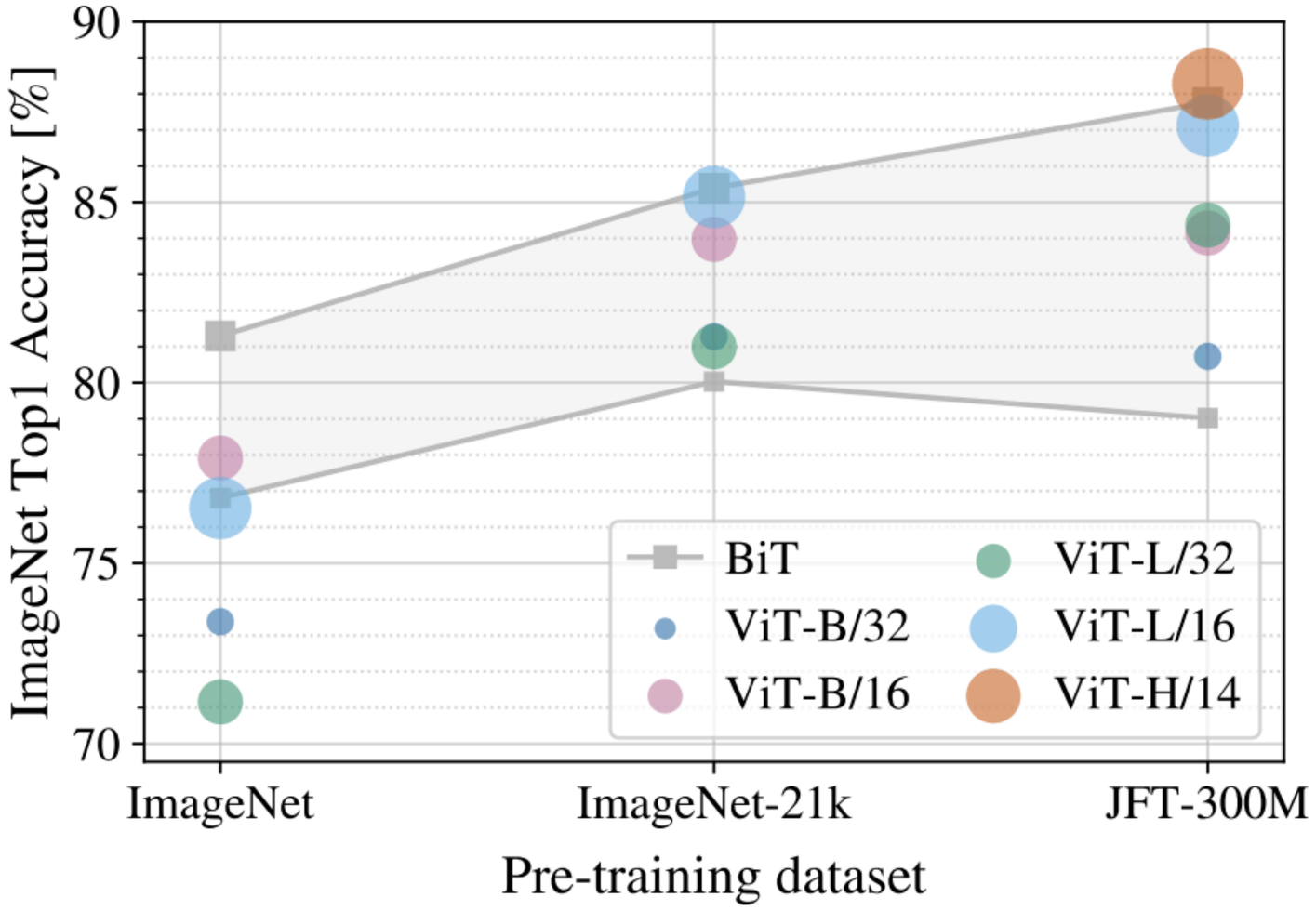
- Split an image into patches, feed linearly projected patches into standard transformer encoder
  - With patches of 14x14 pixels, you need 16x16=256 patches to represent 224x224 images



A. Dosovitskiy et al. An image is worth 16x16 words: Transformers for image recognition at scale. ICLR 2021

Source: S. Lazebnik

Vision transformer (ViT)



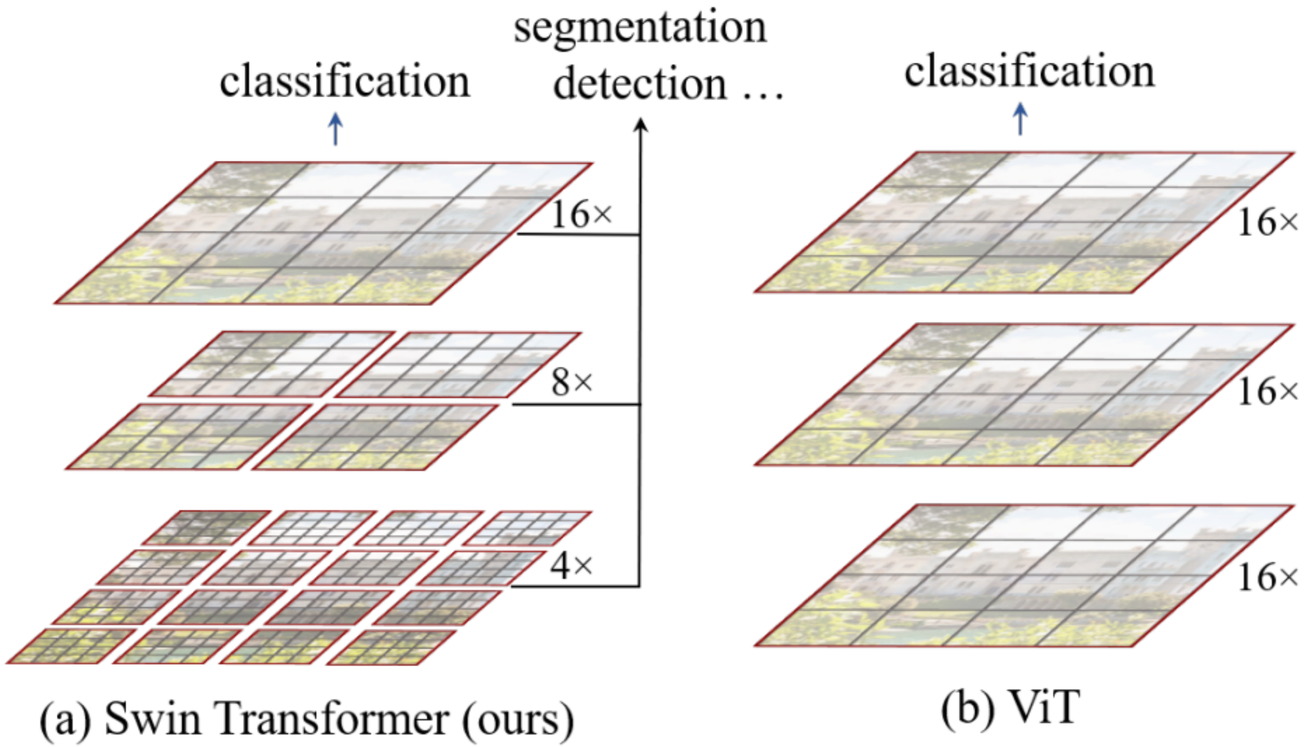
BiT: Big Transfer (ResNet)
ViT: Vision Transformer (Base/Large/Huge, patch size of 14x14, 16x16, or 32x32)

Internal Google dataset (not public)

A. Dosovitskiy et al. An image is worth 16x16 words: Transformers for image recognition at scale. ICLR 2021

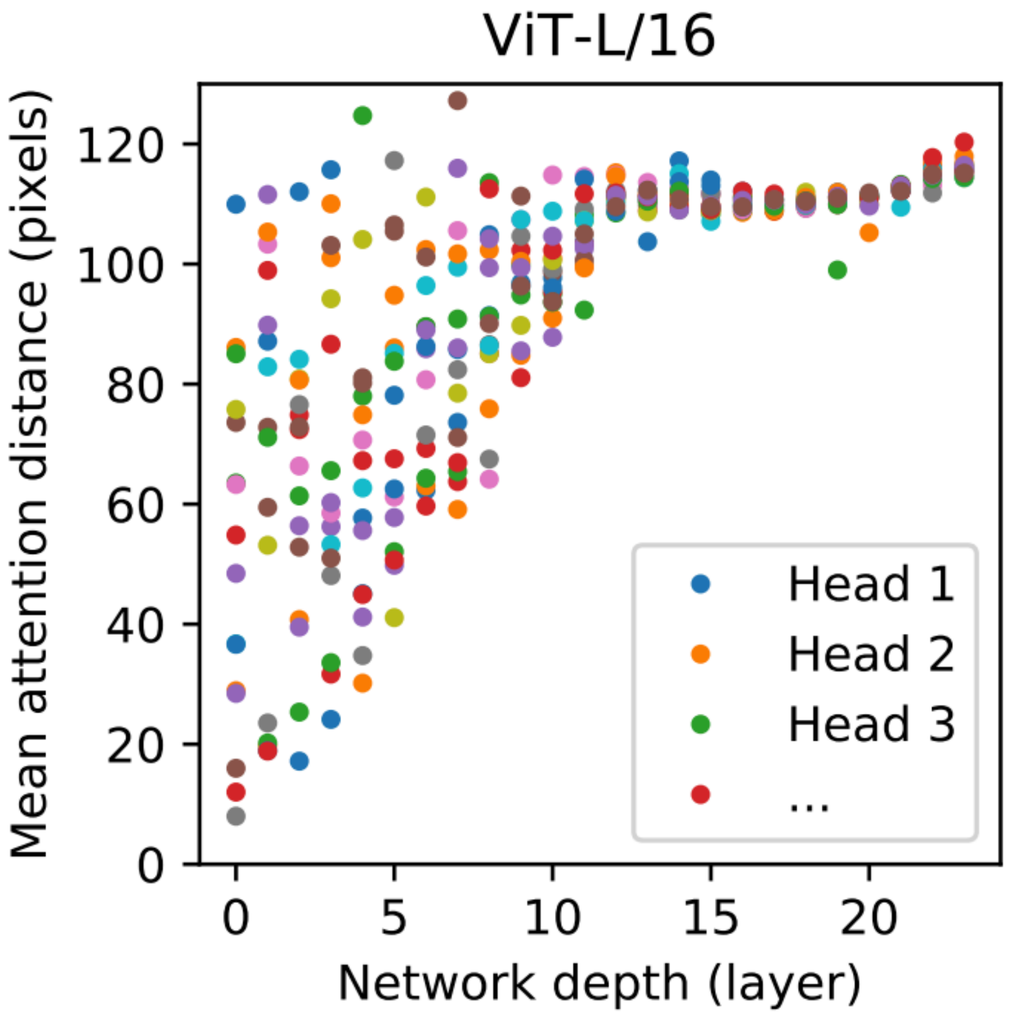
Source: S. Lazebnik

Swin Transformer: windowed attention



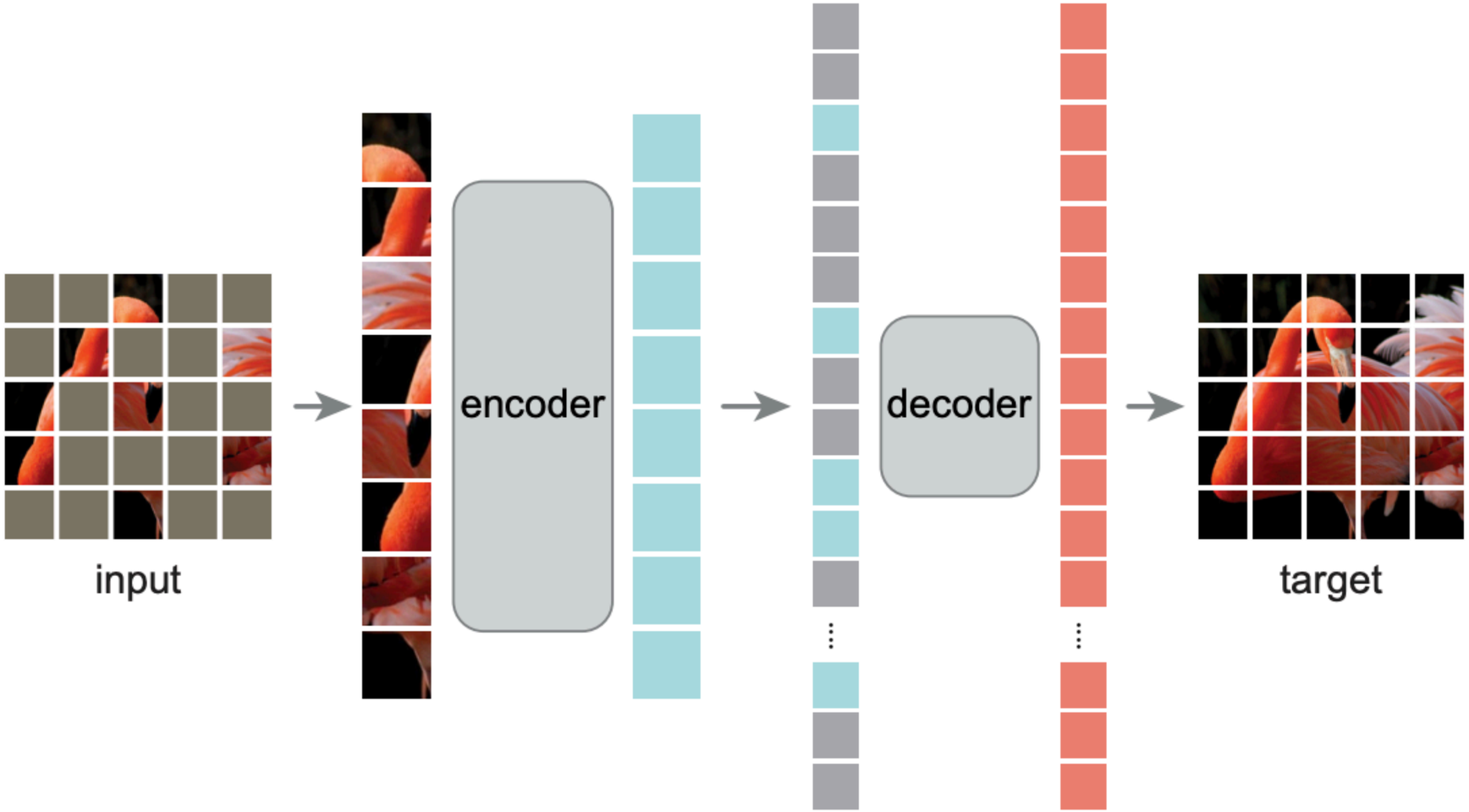
[Liu et al., “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”]

Vision transformer (ViT)



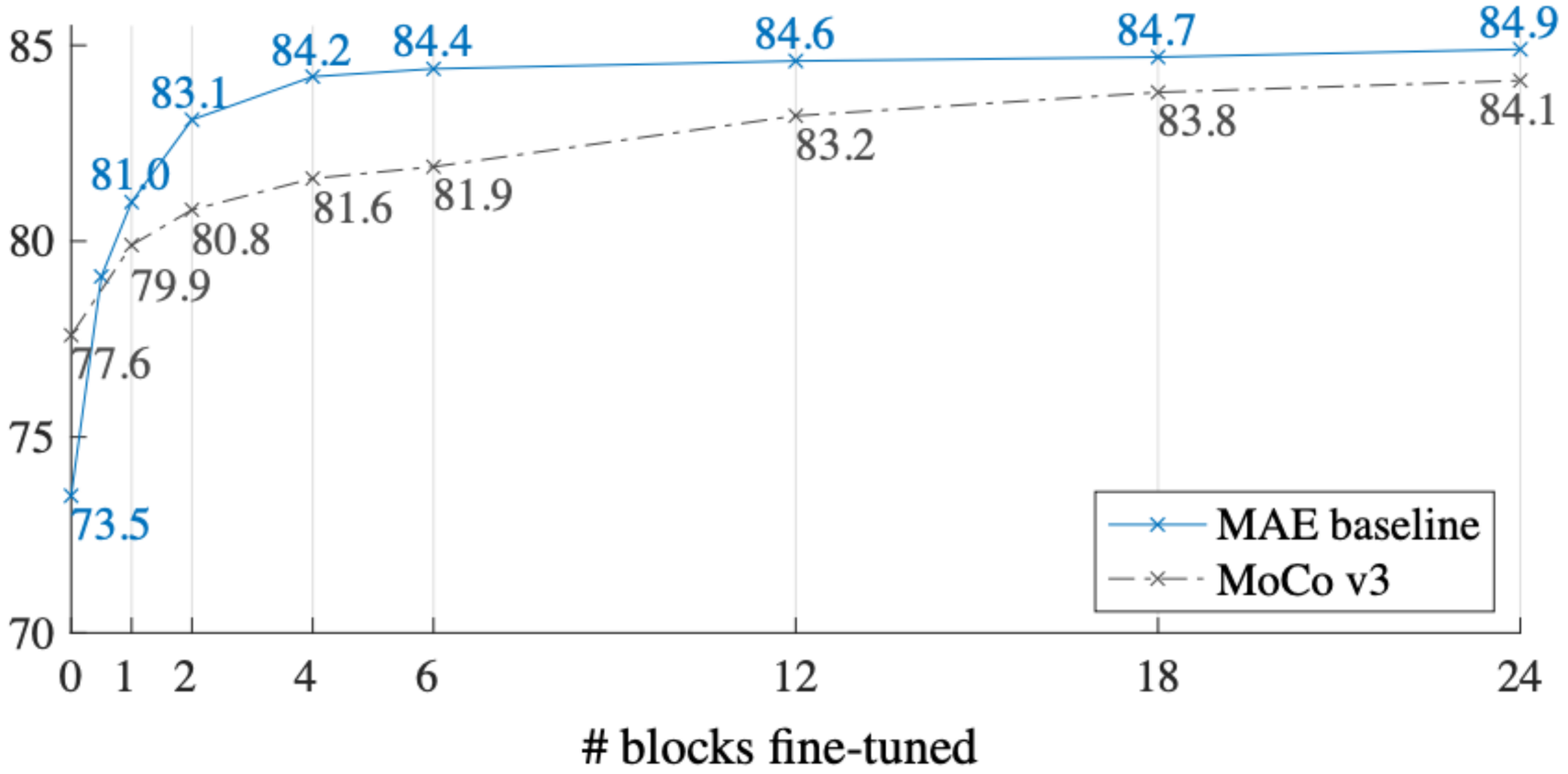
A. Dosovitskiy et al. An image is worth 16x16 words: Transformers for image recognition at scale. ICLR 2021

Application: self-supervised learning



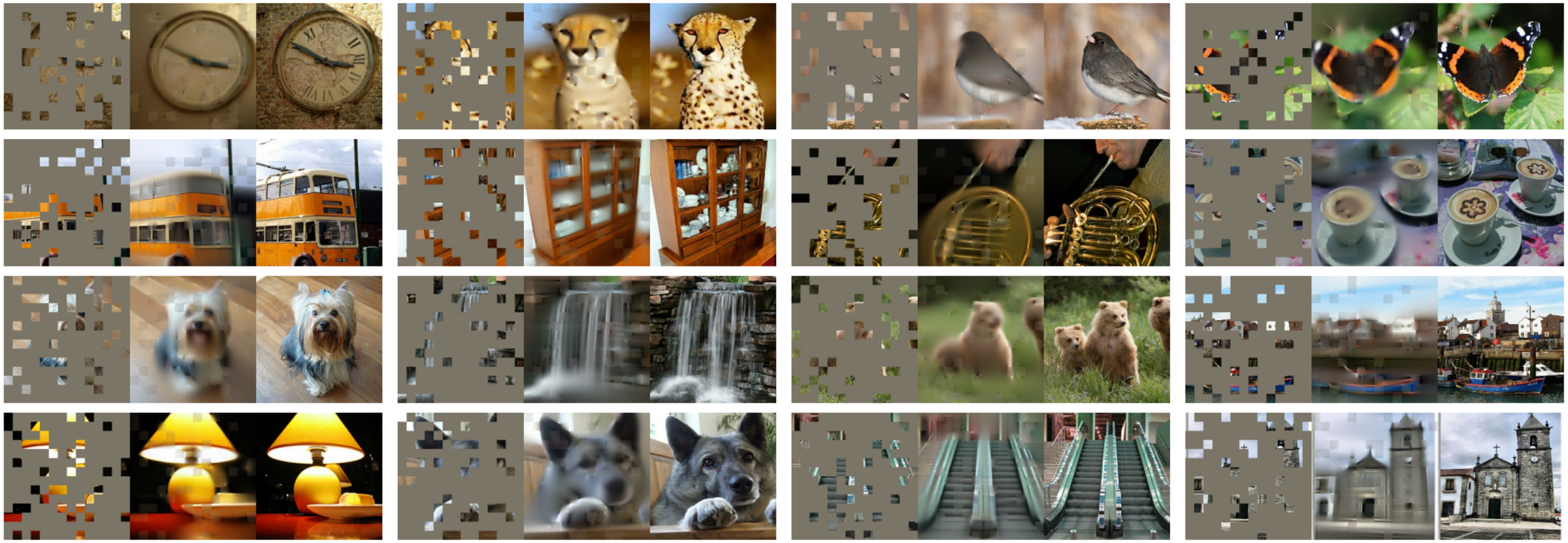
[Kaiming He et al., “Masked Autoencoders Are Scalable Vision Learners”, 2021]

Application: self-supervised learning



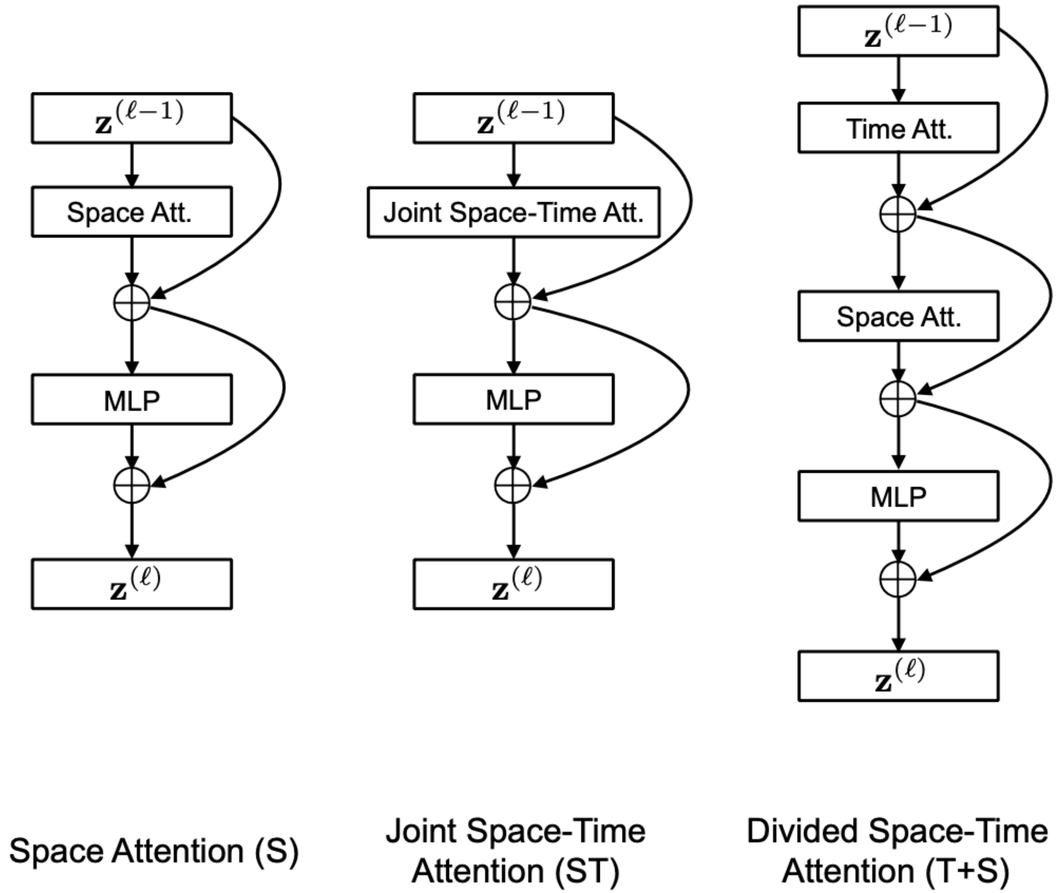
[Kaiming He et al., “Masked Autoencoders Are Scalable Vision Learners”, 2021]

Application: self-supervised learning



[Kaiming He et al., “Masked Autoencoders Are Scalable Vision Learners”, 2021]

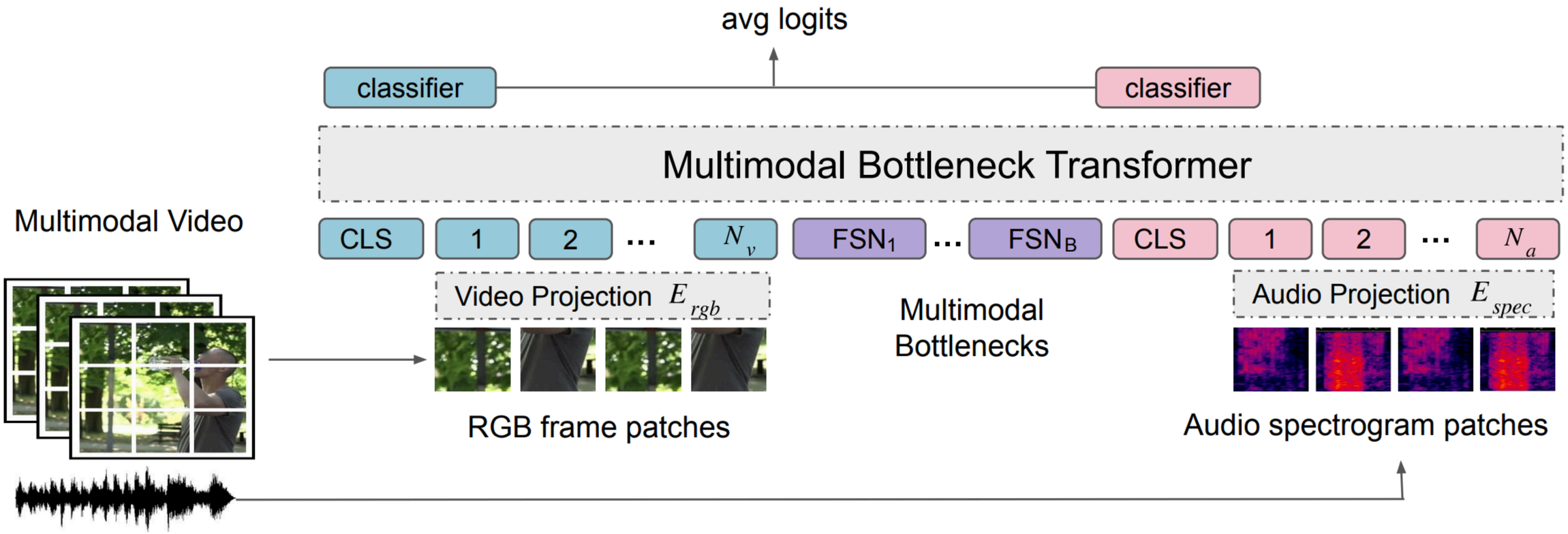
Application: video



Space Attention (S) | Joint Space-Time Attention (ST) | Divided Space-Time Attention (T+S)

Source: [Gedas Bertasius et al, “Is Space-Time Attention All You Need for Video Understanding?”, 2021]

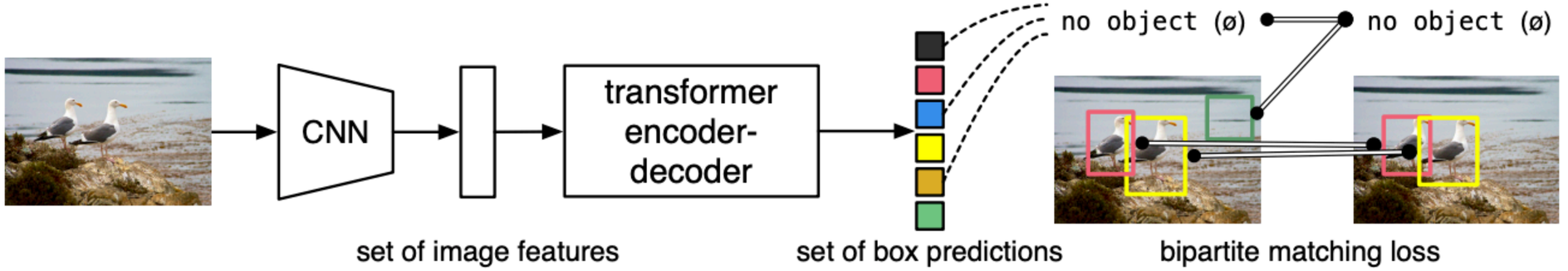
Application: multimodal models

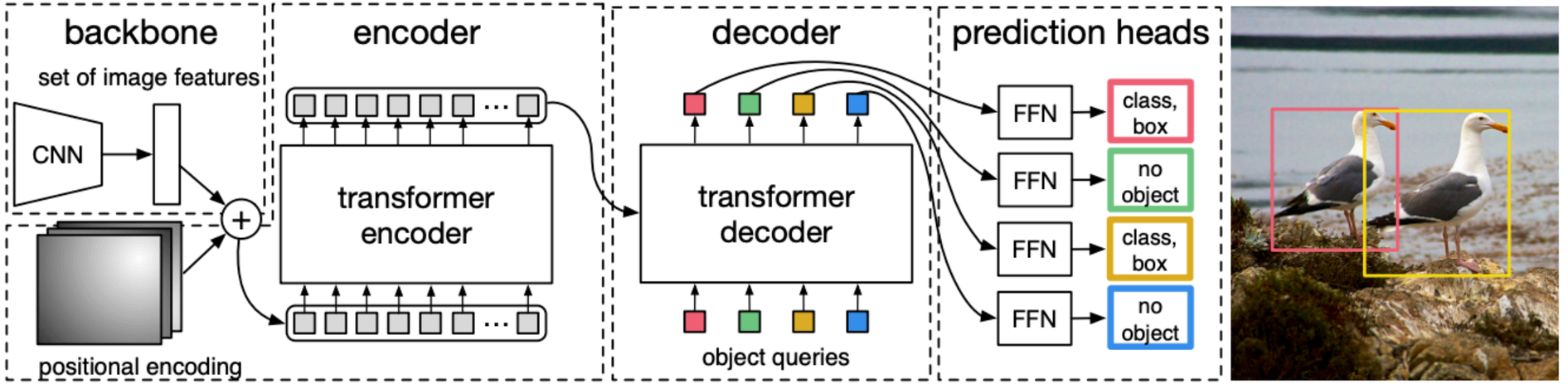


[Arsha Nagrani et al, “Attention Bottlenecks for Multimodal Fusion”, 2021]

Detection Transformer (DETR) – Facebook AI

- Hybrid of CNN and transformer, aimed at standard recognition task



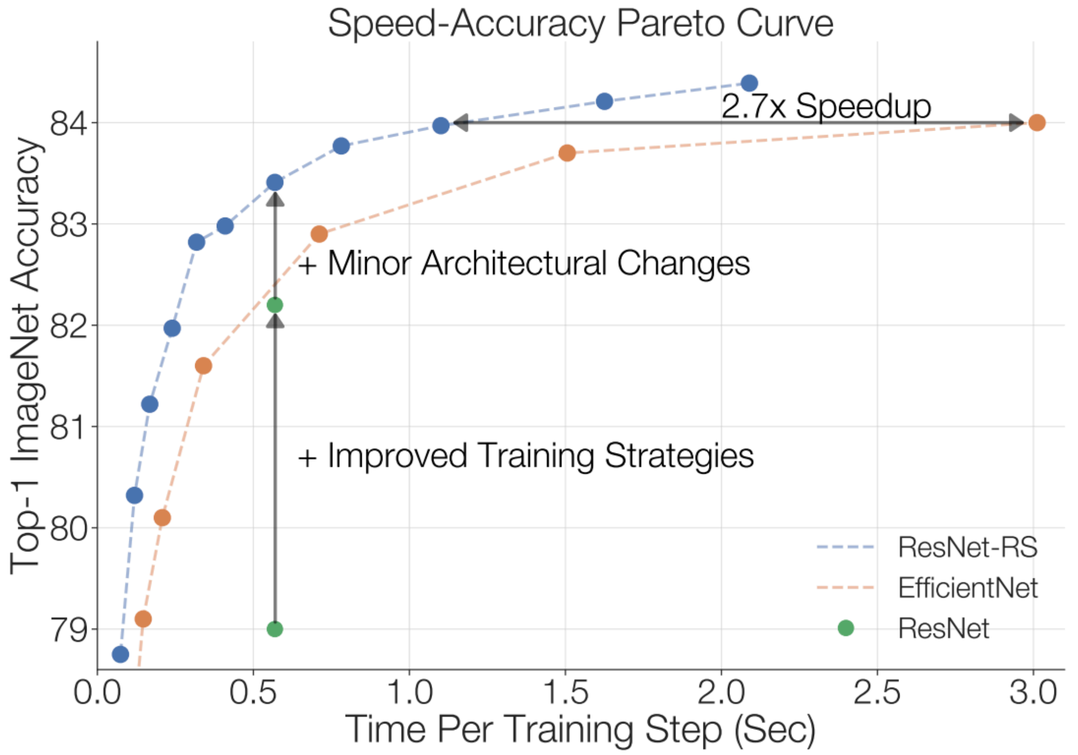


N. Carion et al., End-to-end object detection with transformers, ECCV 2020

Source: S. Lazebnik

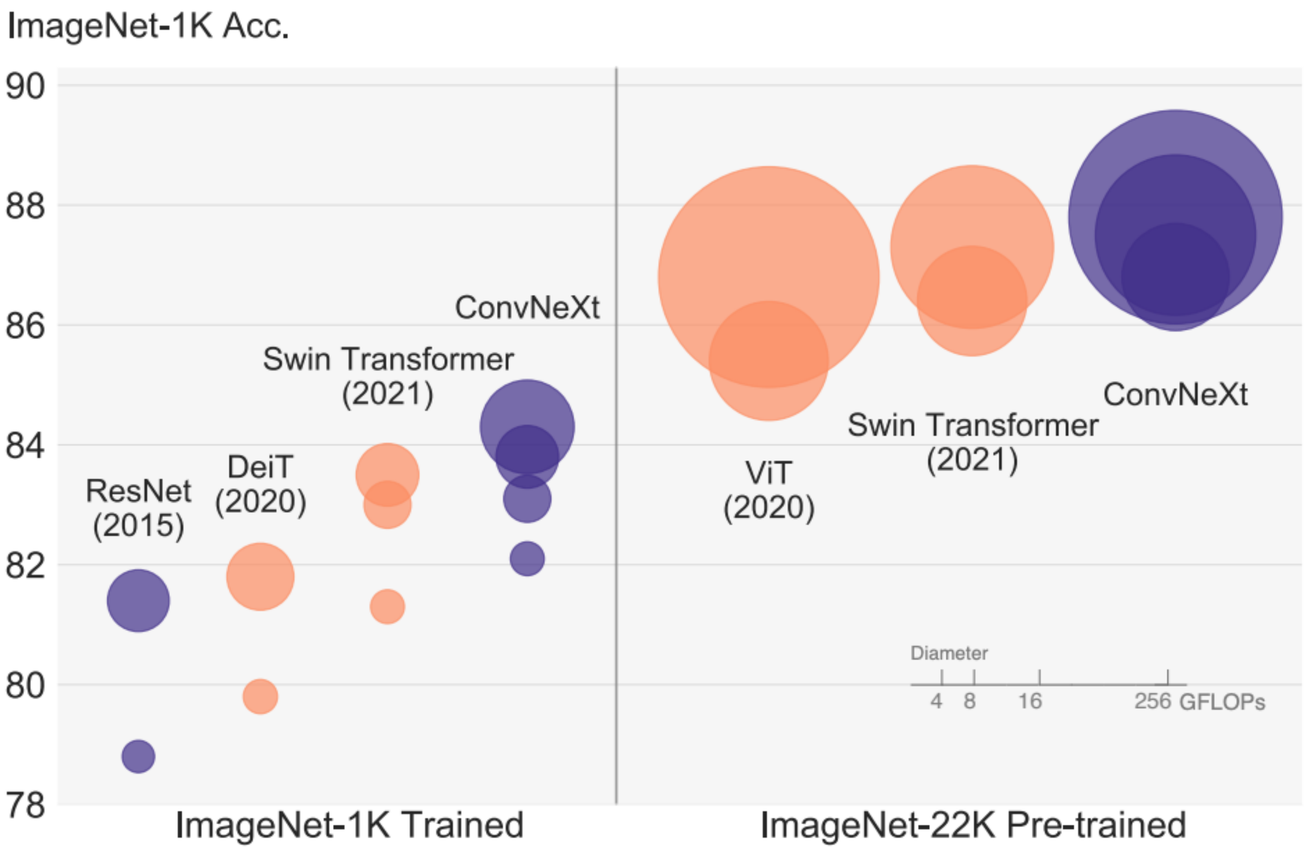
CNNs can be improved too, though.

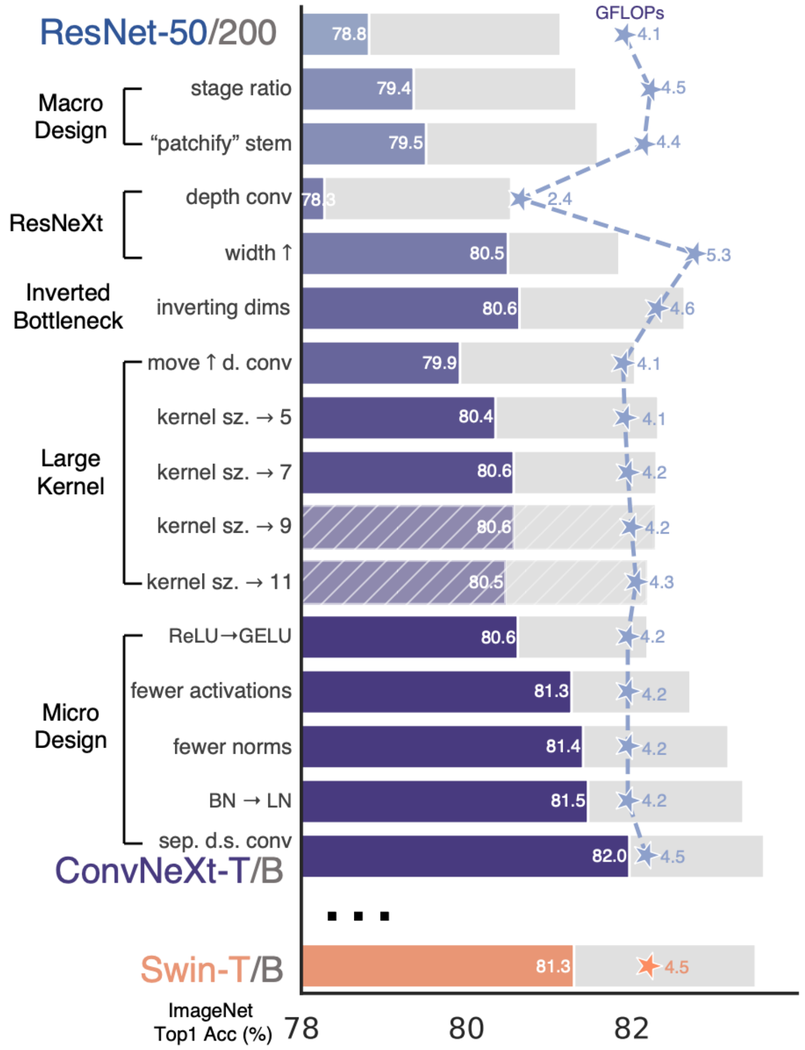
Improvements	Top-1	Δ
ResNet-200	79.0	—
+ Cosine LR Decay	79.3	**+0.3**
+ Increase training epochs	78.8 †	-0.5
+ EMA of weights	79.1	**+0.3**
+ Label Smoothing	80.4	**+1.3**
+ Stochastic Depth	80.6	**+0.2**
+ RandAugment	81.0	**+0.4**
+ Dropout on FC	80.7 ‡	-0.3
+ Decrease weight decay	82.2	**+1.5**
+ Squeeze-and-Excitation	82.9	**+0.7**
+ ResNet-D	83.4	**+0.5**



[Irwan Bello et al. "Revisiting ResNets: Improved Training and Scaling Strategies", 2021]

CNNs can be improved too, though.





[Liu et al. "A ConvNet for the 2020s", 2022]

Next class: Bias and ethics