



Vision Transformers

DL4DS – Spring 2025

A survey in three papers.

slido



What's a fun and/or interesting thing you did over break?

ⓘ Start presenting to display the poll results on this slide.

Reminders

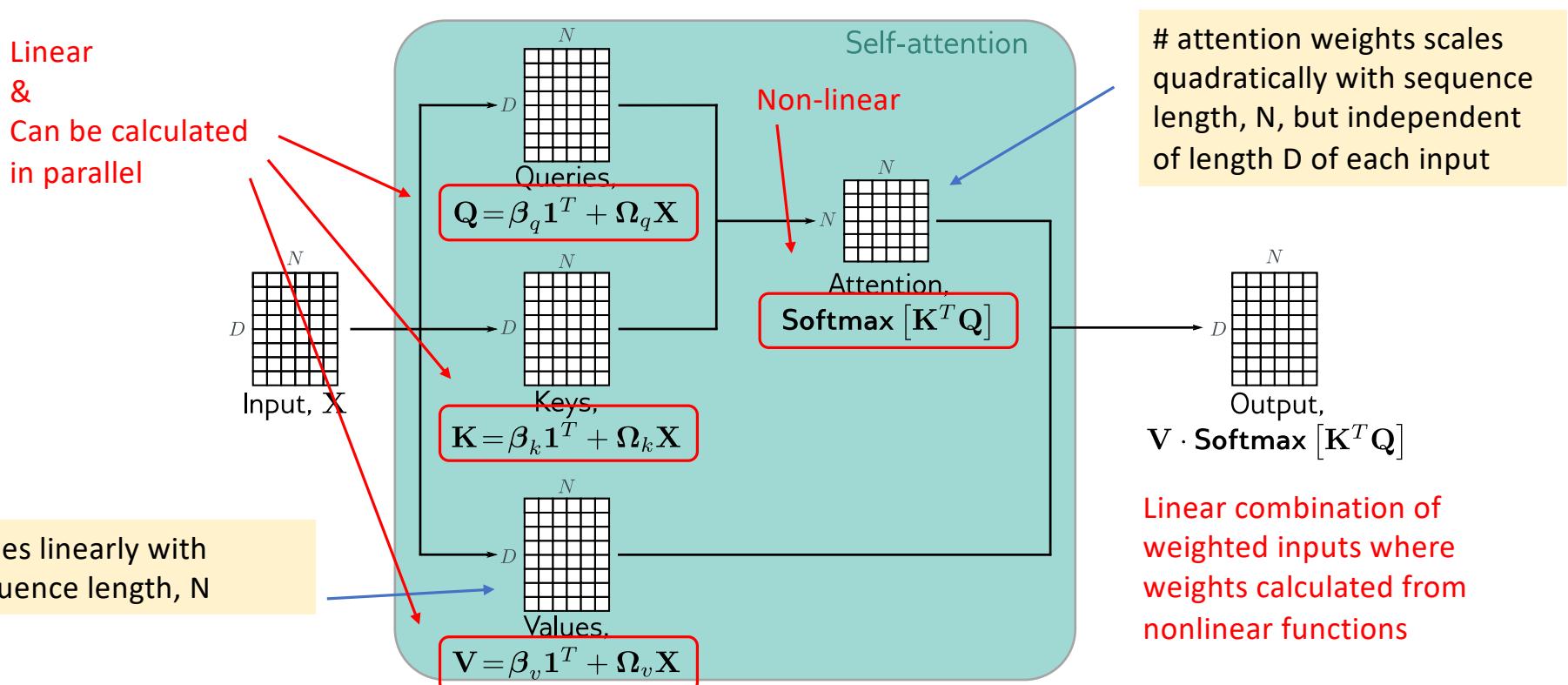
- 3 Jupyter Notebooks assigned
- Midterm Challenge released
- Mid-project check-in

Topics

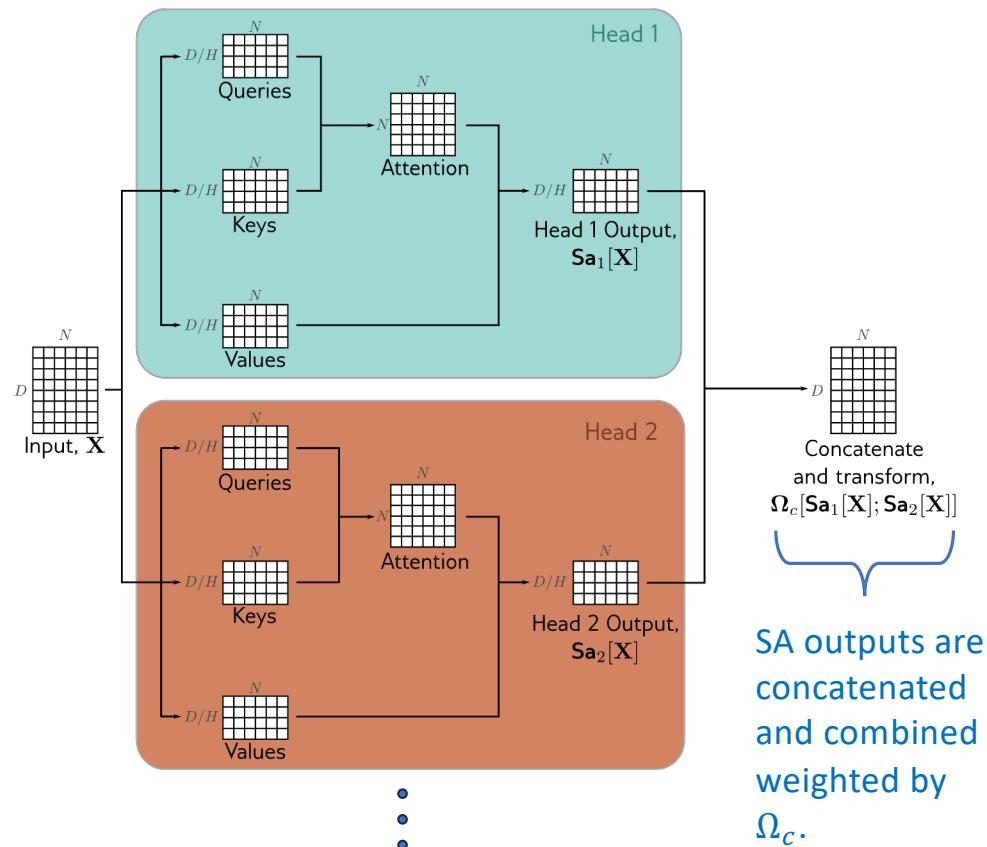
- Transformers Recap
- ImageGPT
- Vision Transformer (ViT)
- CLIP – Contrastive Learning w/ Image Pre-Training

Transformers Recap

Hypernetwork – 1 branch calculates weights of other branch



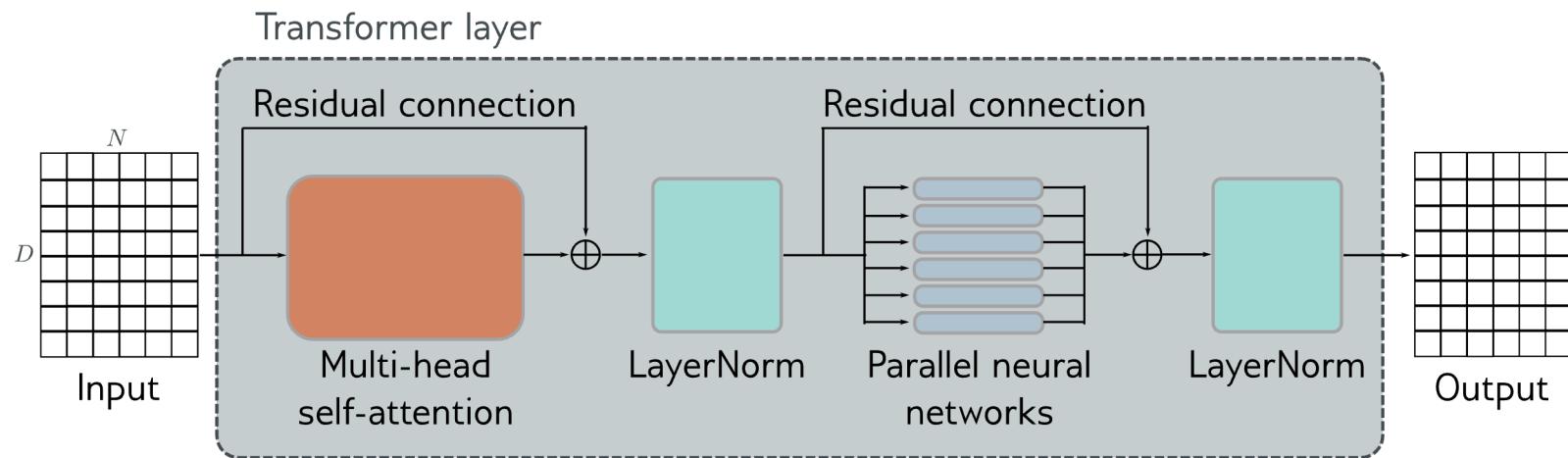
Multi-Head Self Attention



- Multiple self-attention heads are usually applied in parallel
- “allows model to jointly attend to info from different representation subspaces at different positions”
- Original paper used 8 heads
- All can be executed in parallel

SA outputs are concatenated and combined weighted by Ω_c .

Transformer Layer -- Complete



Transform Layer	
\mathbf{X}	$\leftarrow \mathbf{X} + \text{MhSa}[\mathbf{X}]$
\mathbf{X}	$\leftarrow \text{LayerNorm}[\mathbf{X}]$
\mathbf{x}_n	$\leftarrow \mathbf{x}_n + \text{mlp}[\mathbf{x}_n]$
\mathbf{X}	$\leftarrow \text{LayerNorm}[\mathbf{X}],$

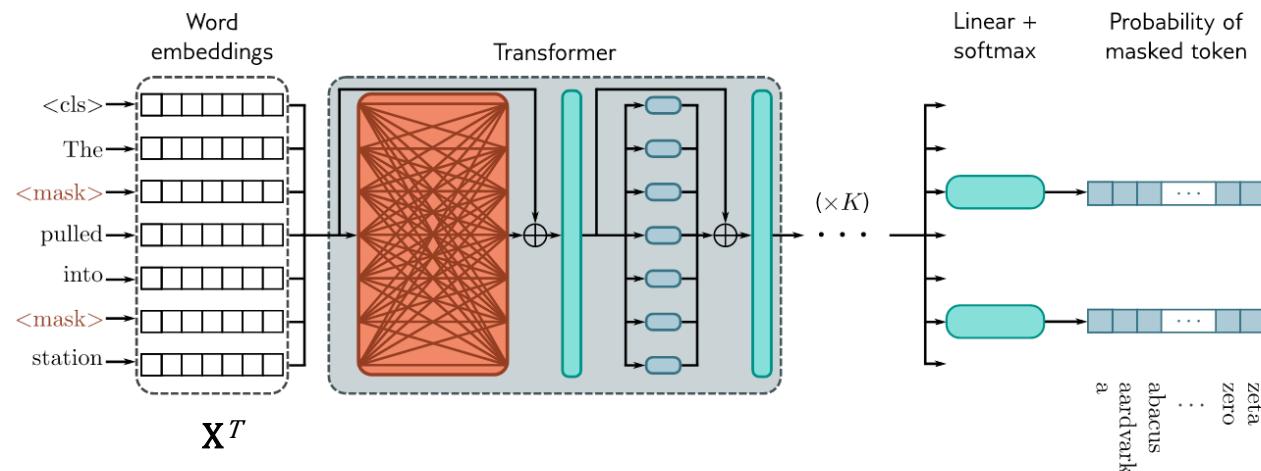
LayerNorm

$$y = \frac{x - \mathbb{E}[x]}{\sqrt{\text{Var}[x] + \epsilon}} * \gamma + \beta$$

Calculated column-wise

Encoder Pre-Training

Special <cls> token used for aggregate sequence representation for classification

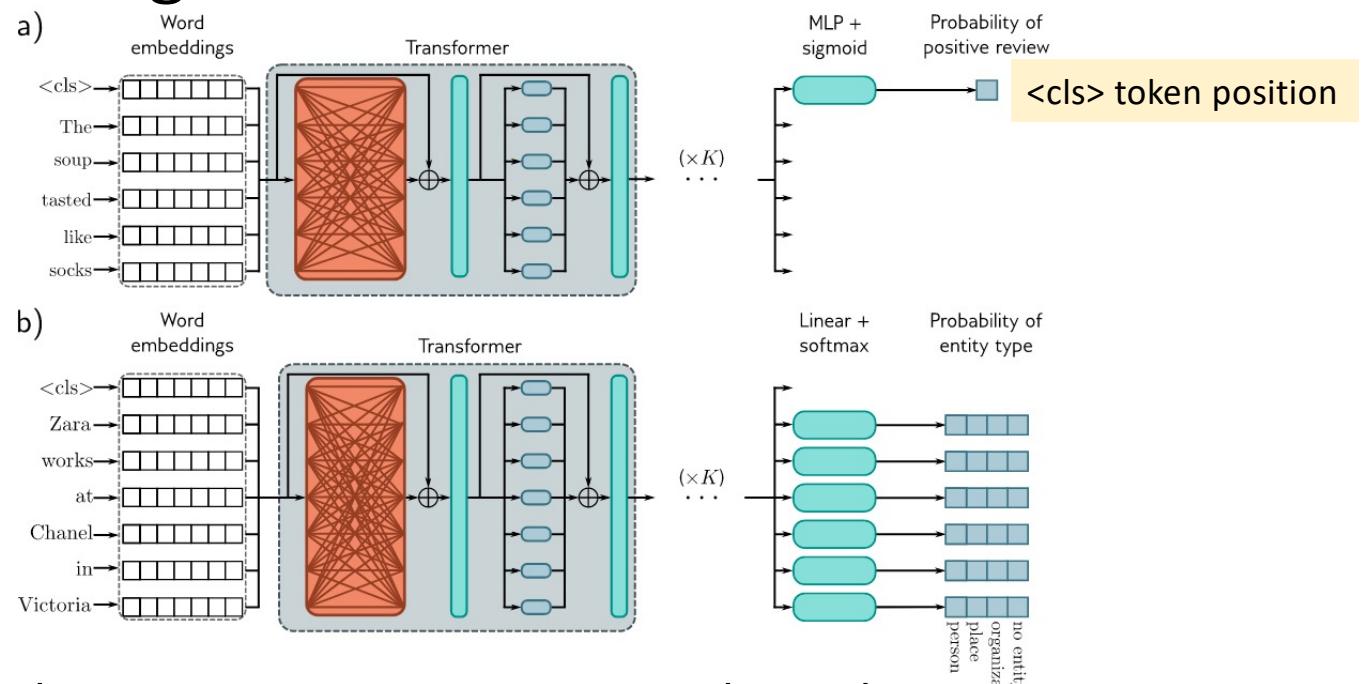


- A small percentage of input embedding replaced with a generic <mask> token
- Predict missing token from output embeddings
- Added linear layer and softmax to generate probabilities over vocabulary
- Trained on BooksCorpus (800M words) and English Wikipedia (2.5B words)

Encoder Fine-Tuning

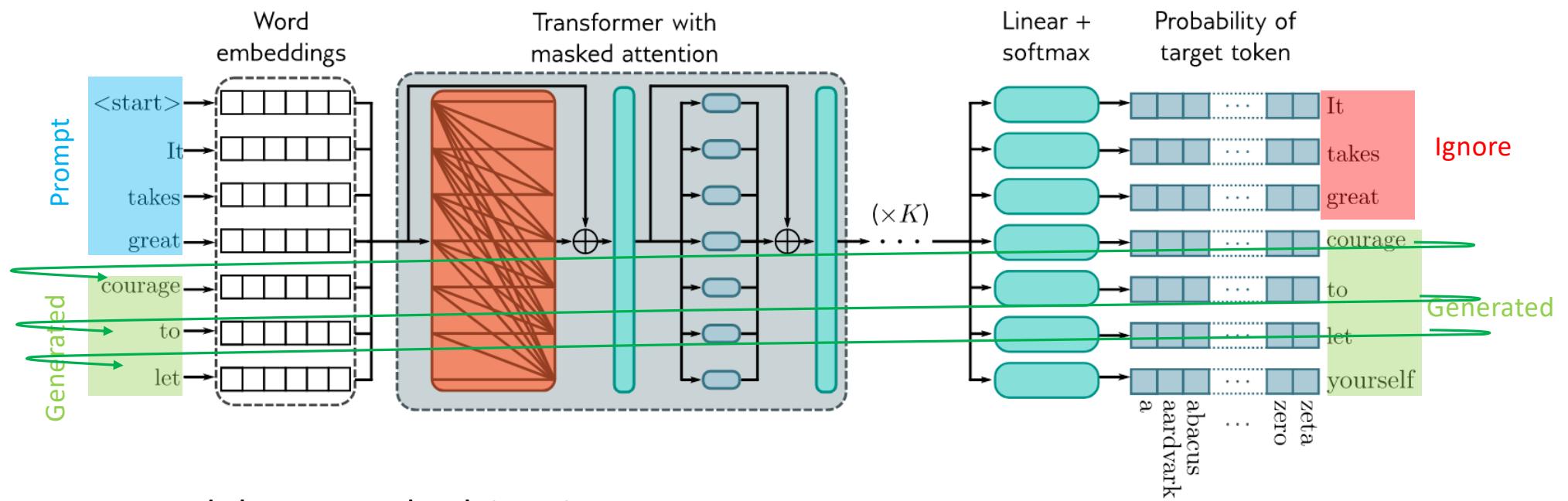
Sentiment Analysis

Named Entity Recognition (NER)

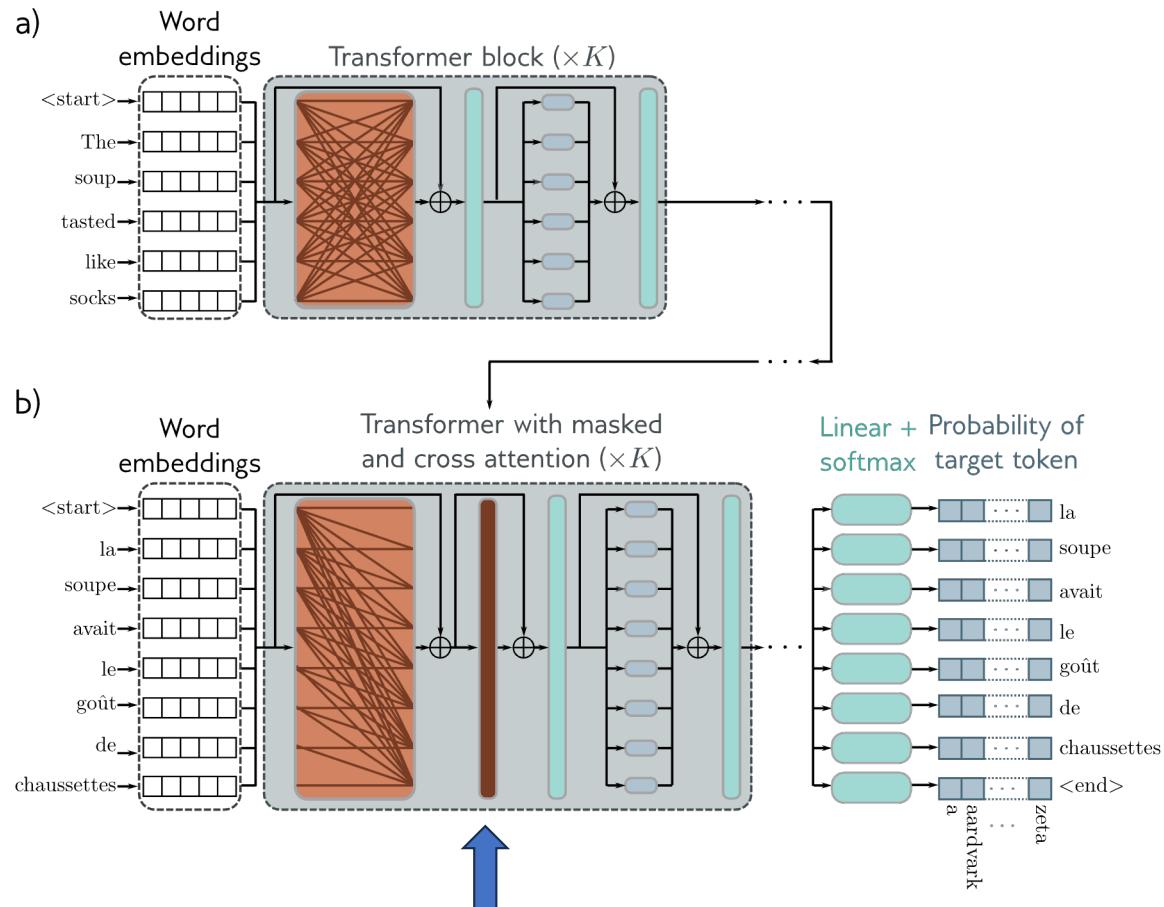


- Extra layer(s) appended to convert output vectors to desired output format
- 3rd Example: Text span prediction -- predict start and end location of answer to a question in passage of Wikipedia, see <https://rajpurkar.github.io/SQuAD-explorer/>

Decoder: Text Generation (Generative AI)

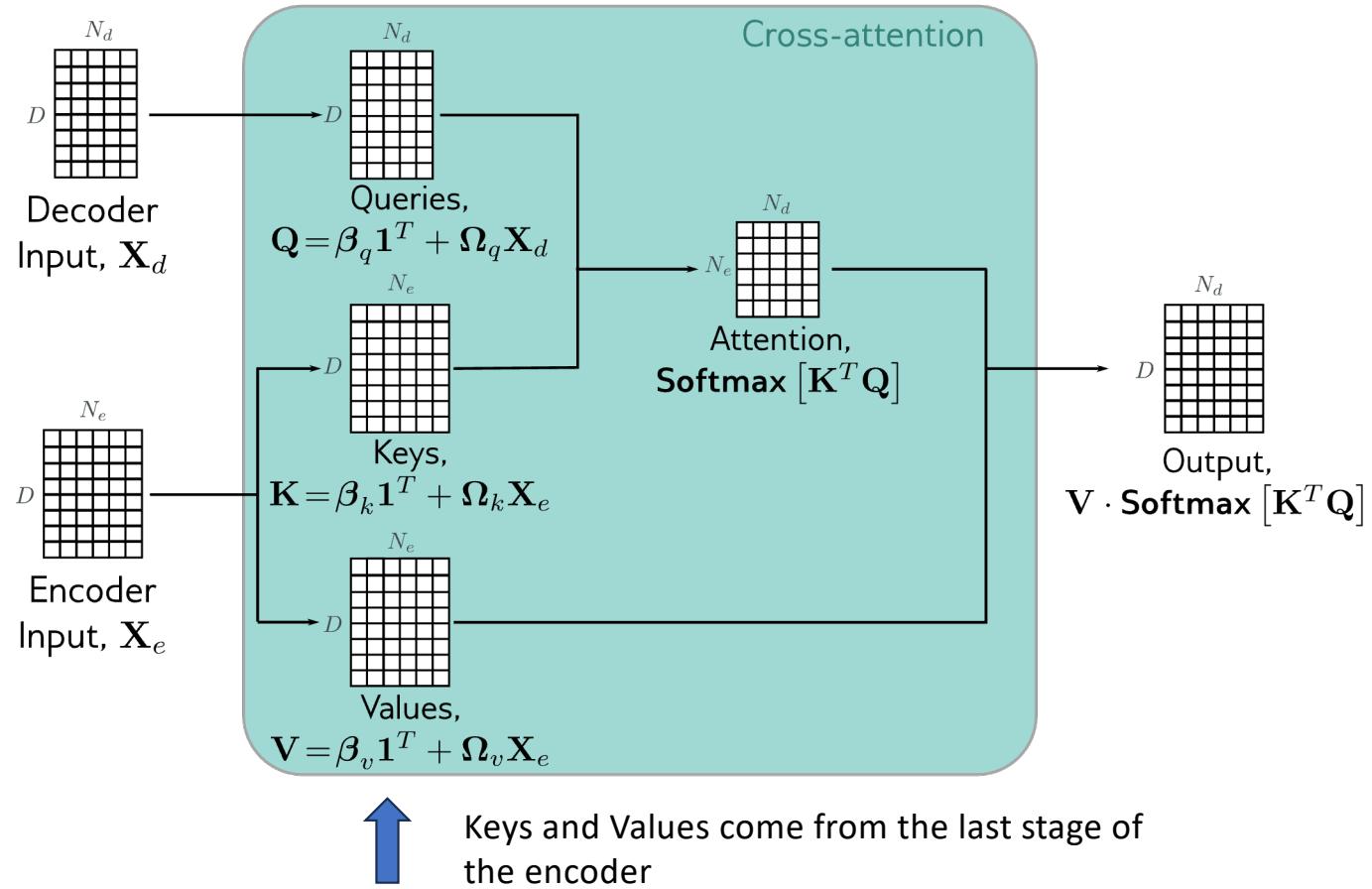


Encoder Decoder Model

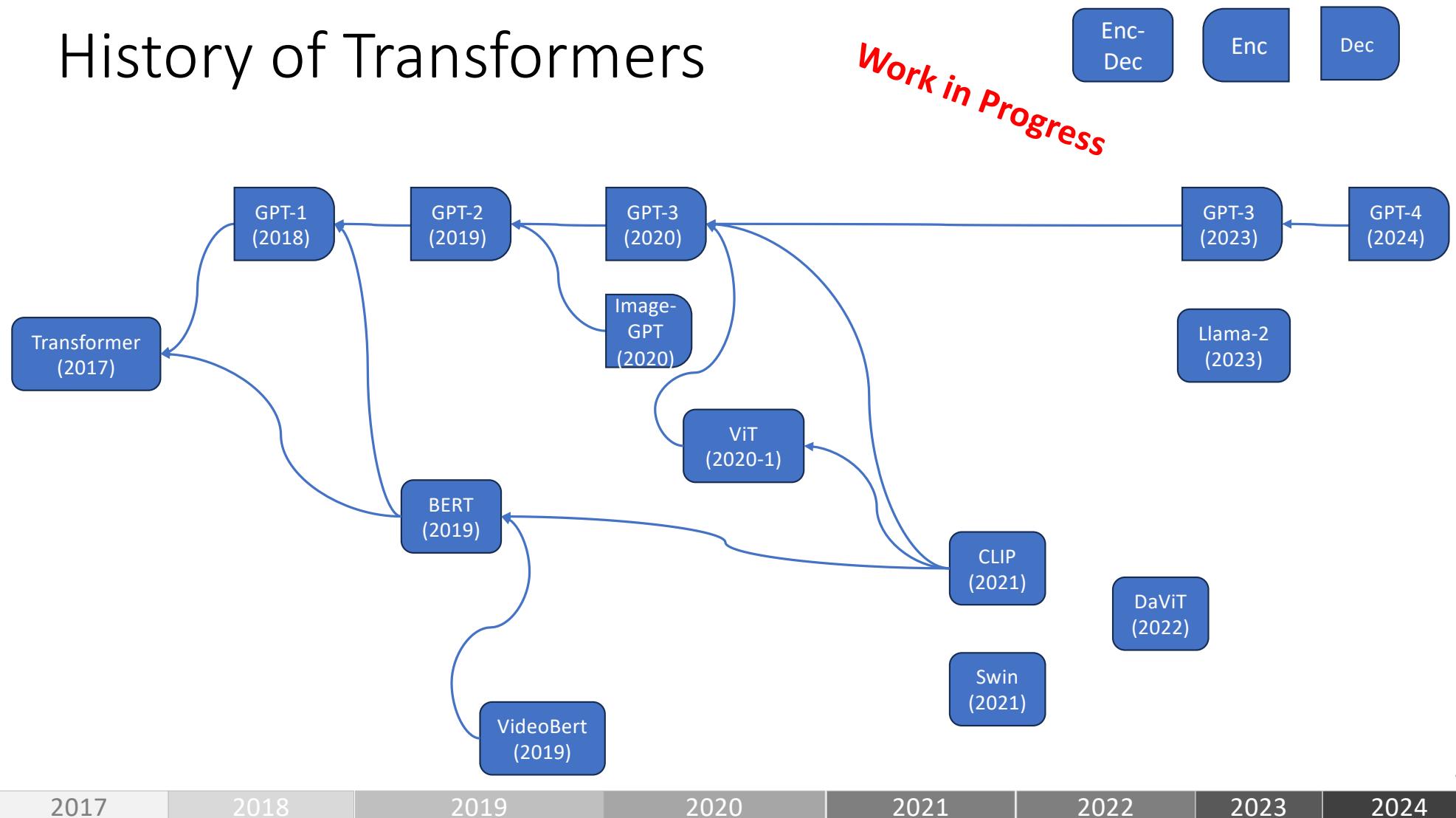


- The transformer layer in the decoder of the encoder-decoder model has an extra stage
- Attends to the input of the encoder with *cross attention* using Keys and Values from the output of the encoder

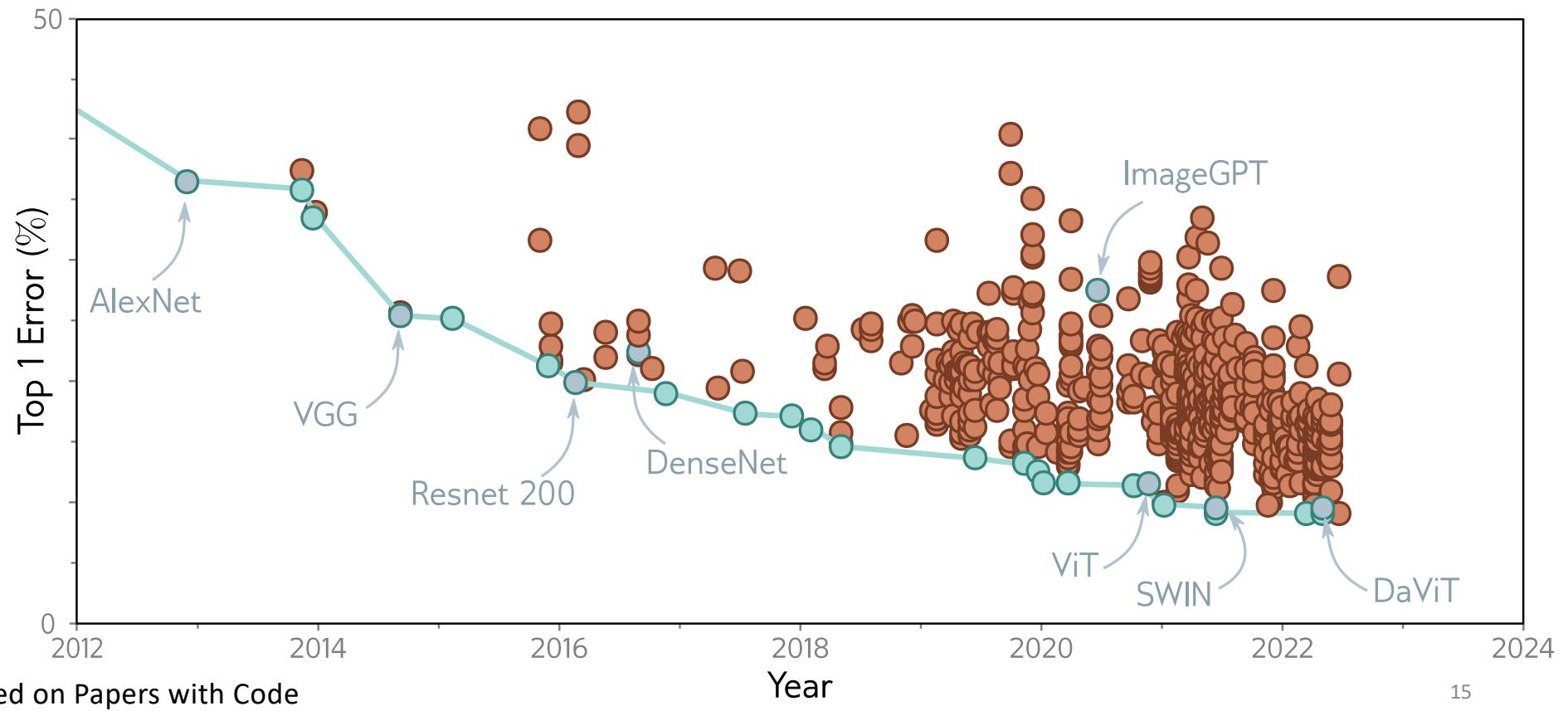
Cross-Attention



History of Transformers

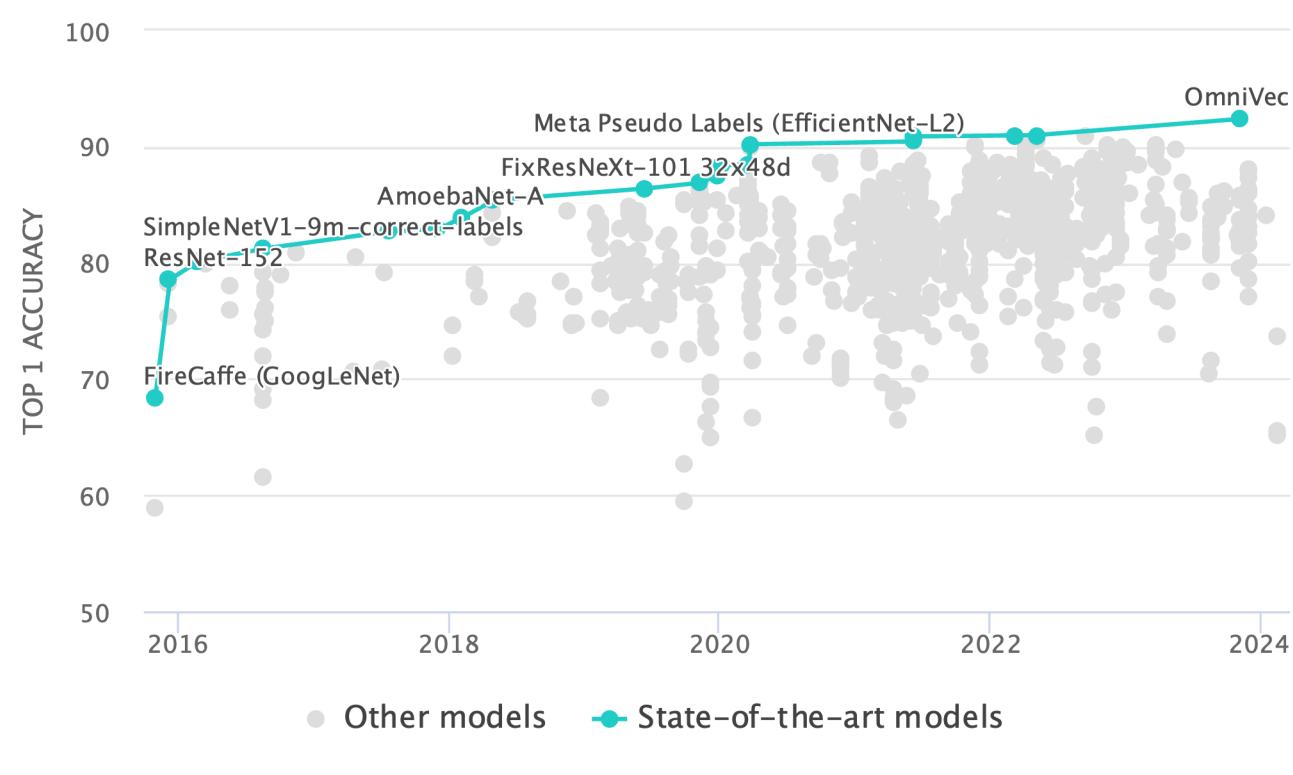


ImageNet History – Top-1 Error



Based on Papers with Code

ImageNet Top-1 Accuracy



<https://paperswithcode.com/sota/image-classification-on-imagenet>

History of Transformers

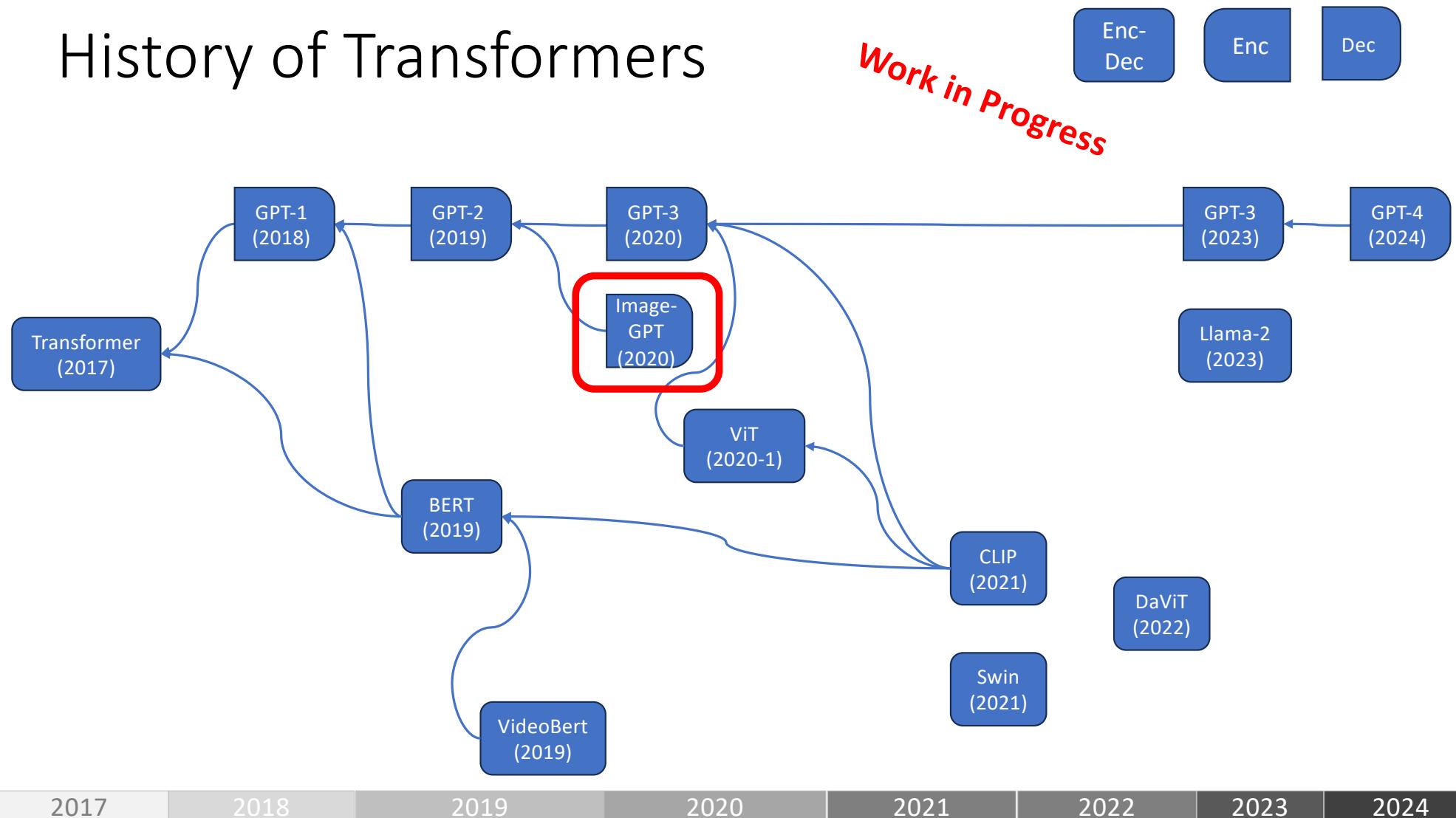


Image GPT – June 2020



- Train GPT-2 scale sequence Transformer to auto-regressively predict pixels, w/o 2D input structure
- Use GPT-2 with only minor changes
- ImageNet Top-1 72% accuracy (not great), trained on ImageNet and web images
- Primary objective is to explore the representation accuracy of internal features

<https://openai.com/research/image-gpt>

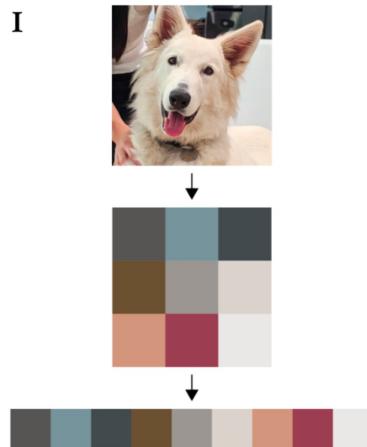
<https://github.com/openai/image-gpt> (deprecated)

https://huggingface.co/docs/transformers/model_doc/imagegpt

M. Chen *et al.*, “Generative Pretraining from Pixels,” OpenAI, Technical Report, Jun. 2020.

Image GPT – Inputs

- Reduced resolution to reduce context size:
 $32 \times 32, 48 \times 48$ or 64×64
- Also reduced color palette from $3 \times 8 = 24$ bit to a 9-bit (512 colors) color palette by clustering (R, G, B) pixels with $k = 512$.



- Reduced resolution to reduce context size:
 $32 \times 32, 48 \times 48$ or 64×64
- Also reduced color palette from $3 \times 8 = 24$ bit to a 9-bit (512 colors) color palette by clustering (R, G, B) pixels with $k = 512$.

<https://openai.com/research/image-gpt>

<https://github.com/openai/image-gpt> (deprecated)

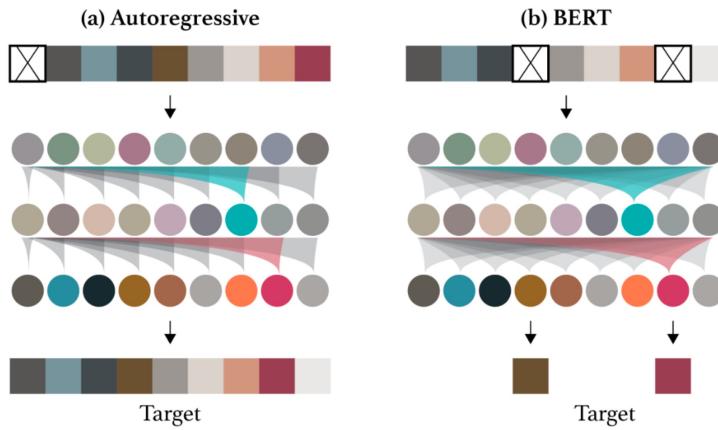
https://huggingface.co/docs/transformers/model_doc/imagegpt

M. Chen *et al.*, “Generative Pretraining from Pixels,” OpenAI, Technical Report, Jun. 2020.

Image GPT – Training Objectives



2



- Tried training with either *Autoregressive* or *BERT* style training objective

<https://openai.com/research/image-gpt>

<https://github.com/openai/image-gpt> (deprecated)

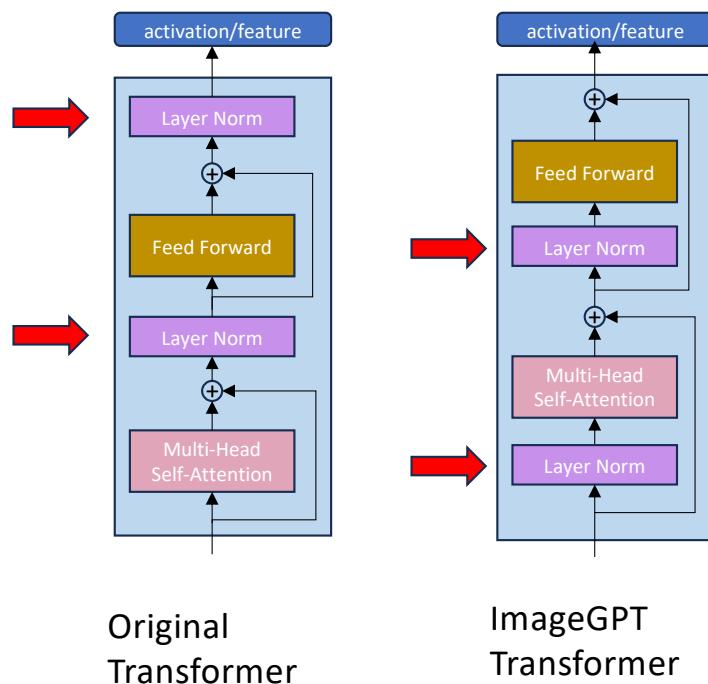
https://huggingface.co/docs/transformers/model_doc/imagegpt

M. Chen *et al.*, “Generative Pretraining from Pixels,” OpenAI, Technical Report, Jun. 2020.

Image GPT – Transformer Layer



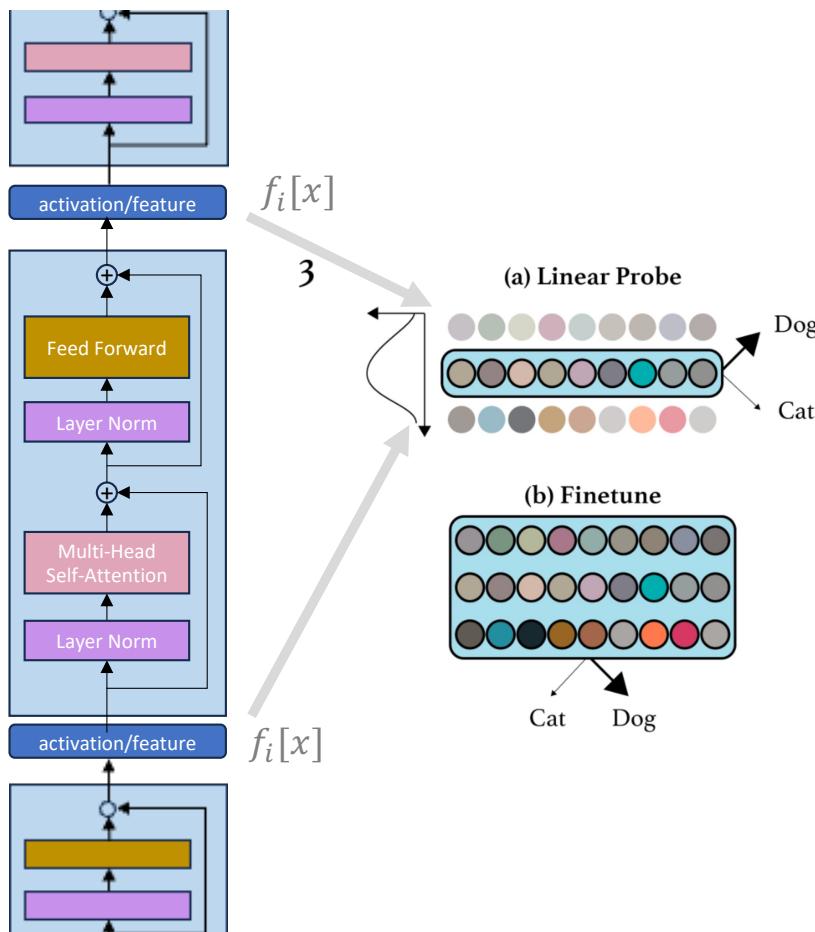
- LayerNorm moved to precede Self-Attention and Feed Forward block
- In the residual path



Original
Transformer

ImageGPT
Transformer

Image GPT – Linear Probes

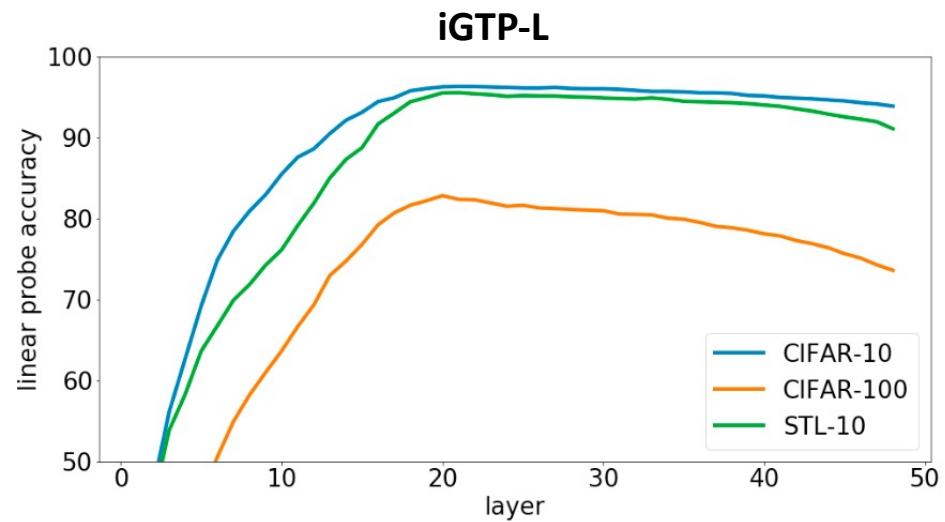


- Use pre-trained model as a “feature extractor”
- Activations after each layer → Features
 - call i^{th} feature: $f_i[x]$
- Good features should linearly separate the classes of transfer tasks
- → linear classifier trained on $(f_i[x], Y)$
- Do this with each feature and see which performs best

Image GPT – Representation Quality



Size	Layers	d	# parms
iGPT-S	24	512	76M
iGPT-M	36	1024	455M
iGPT-L	48	1536	1.4B
iGPT-XL	60	3072	6.8B



- Classification representation quality by feature layer
- Best representation seems to lie in the middle
- As opposed to supervised-training where the best representations lie at the end fo the network

slido



Why is best representation in the middle as opposed to the end of the network like in supervised training?

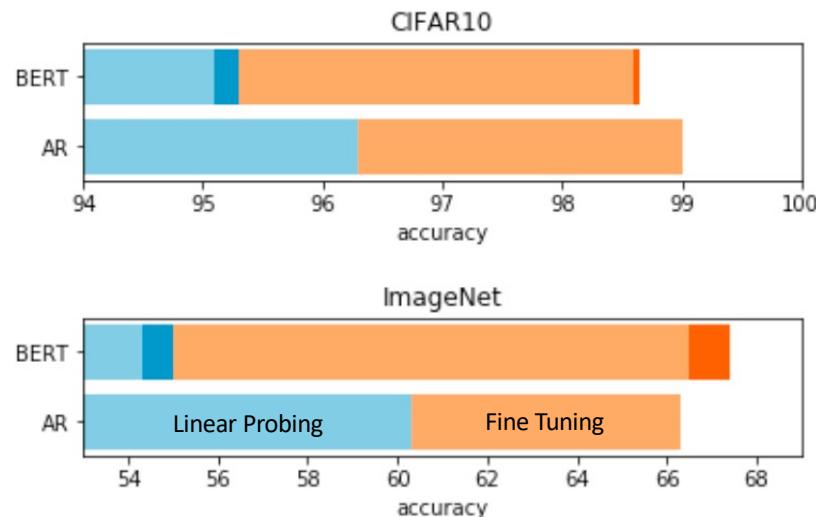
ⓘ Start presenting to display the poll results on this slide.

Image-GPT –

Perhaps generative model operates in two phases:

1. The first phase gathers information from surrounding context in order to build a more global representation.
2. In 2nd phase, contextualized input is used to solve conditional next pixel prediction task

Image GPT – Finetuning for Classification



- Finetuning on the target dataset further improves accuracy
- Finetuning the entire model outperformed finetuning the best linear probe feature

Image GPT – AR Pixel Prediction Results

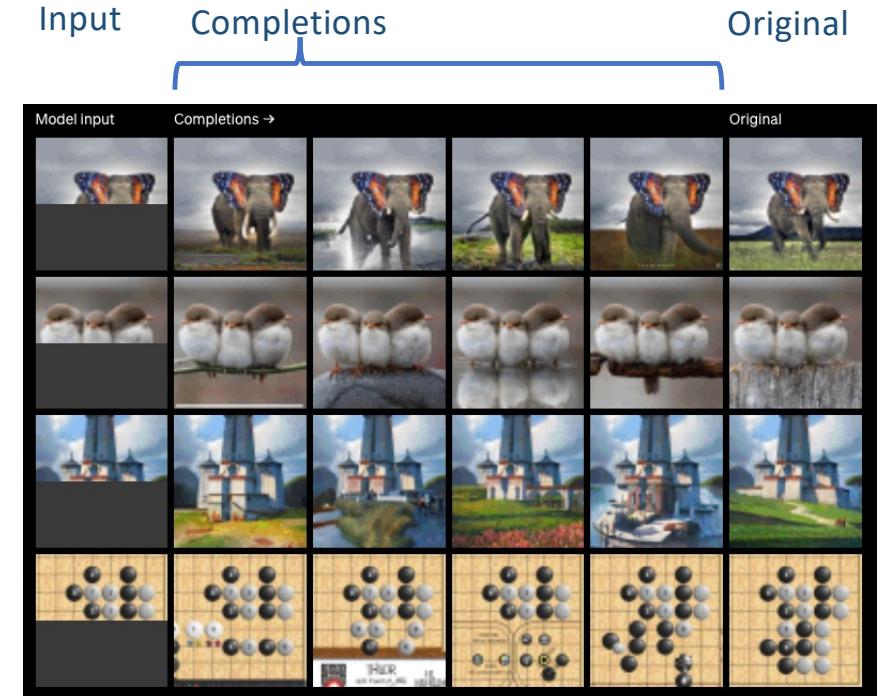
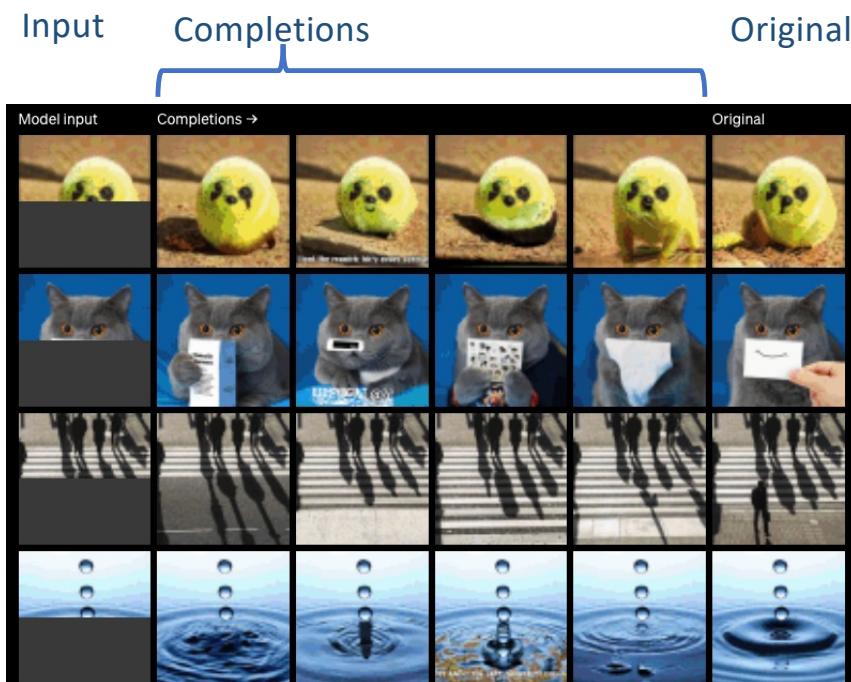


Image GPT – Sampling the Distribution



Image GPT – Pros and Cons

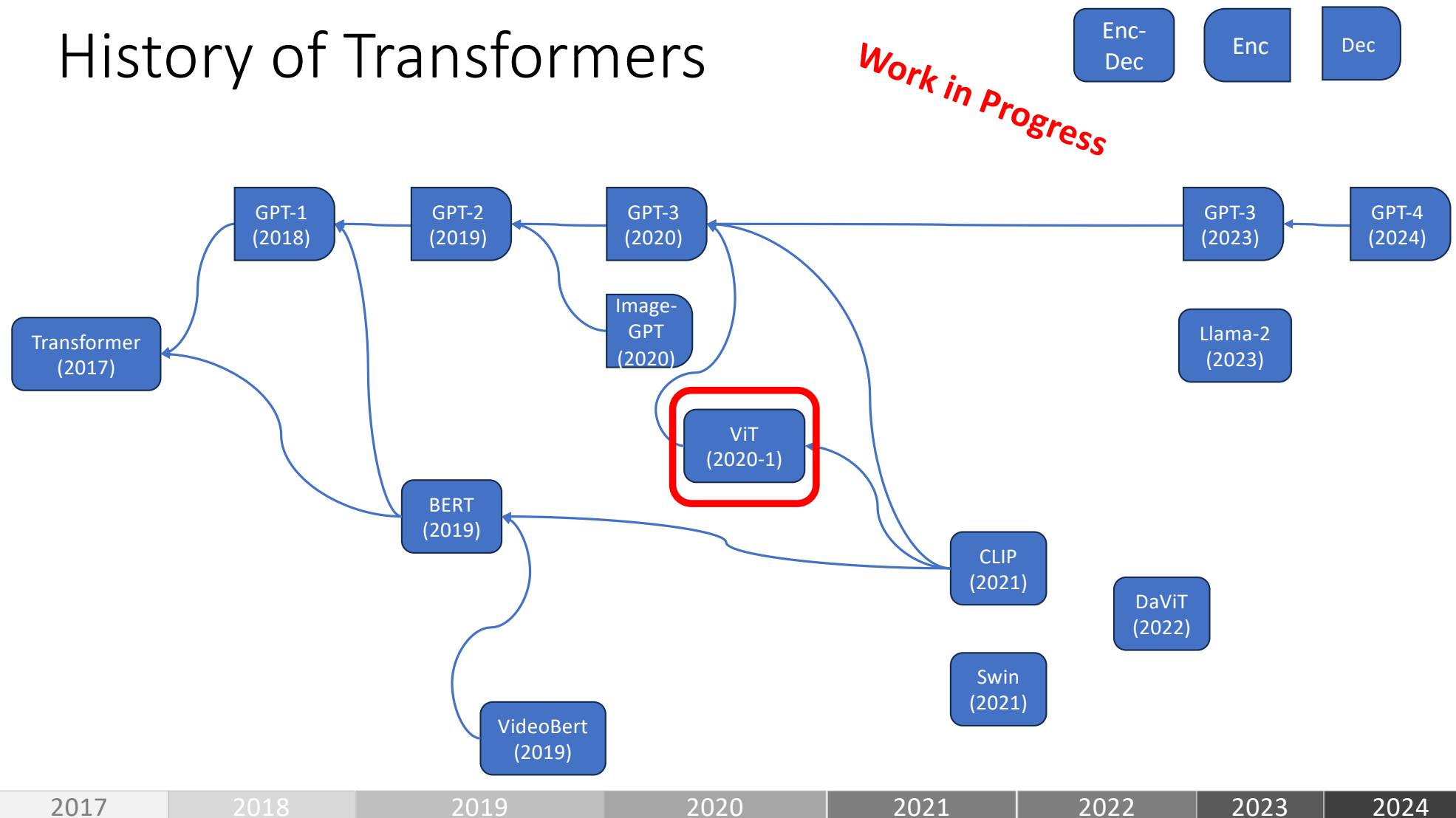
Pro:

- Gave insights into the representational power of Transformers with unsupervised training

Con:

- Worked on downscaled images of size 32x32 to 64x64

History of Transformers



Vision Transformer (ViT) – June 2021

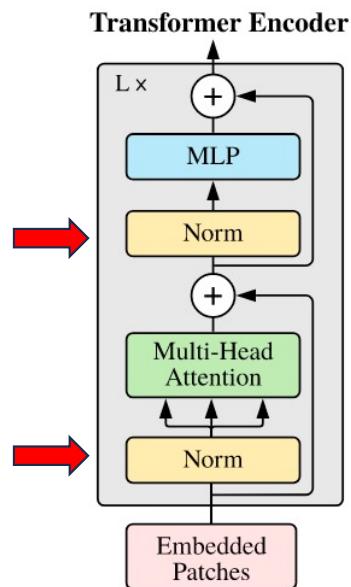
- Overcomes resolution limitation of ImageGPT by using patches
- Follows scalable NLP Transformer architectures to benefit from efficient implementations
- ImageNet Top-1 accuracy: [88.55%](#)
- Performs poorly if just trained on ImageNet
 - → can be expected since Transformers lack the inductive bias of CNNs
- Competitive when pre-trained on very large datasets (e.g. 14M – 300M) images – all supervised at this point

Large scale training trumps inductive bias.

<https://arxiv.org/abs/2010.11929v2>

https://github.com/google-research/vision_transformer

Vision Transformer (ViT) – June 2021



- Uses same Transformer layer as ImageGPT and scalable NLP Transformers

<https://arxiv.org/abs/2010.11929v2>
https://github.com/google-research/vision_transformer

ViT: Putting it all together

1. Divide image into $P \times P$ patches



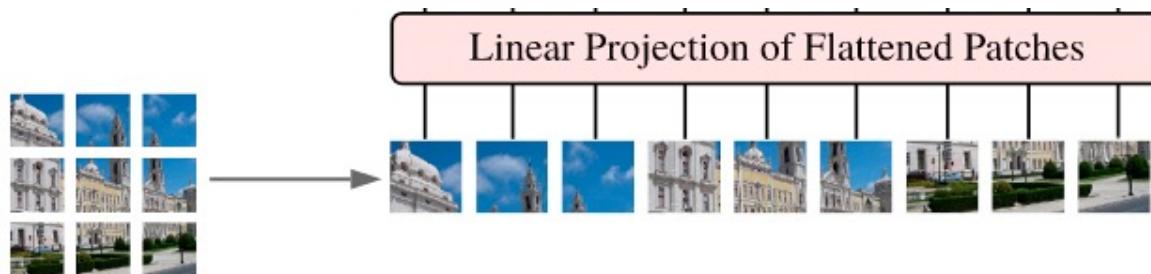
ViT: Putting it all together

1. Divide image into $P \times P$ patches
2. Create sequence of length $N = HW / P^2$



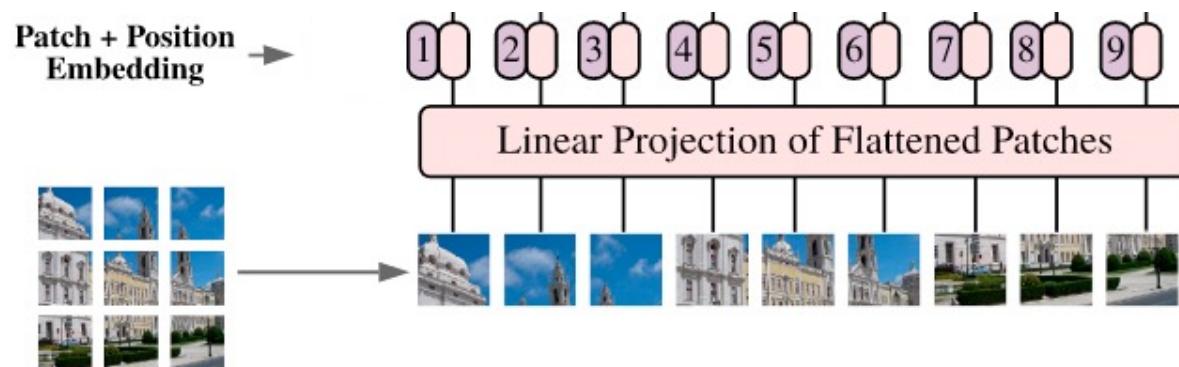
ViT: Putting it all together

1. Divide image into $P \times P$ patches
2. Create sequence of length $N = HW/P^2$
3. Flatten the patches and map to D dimensions with a trainable linear projection



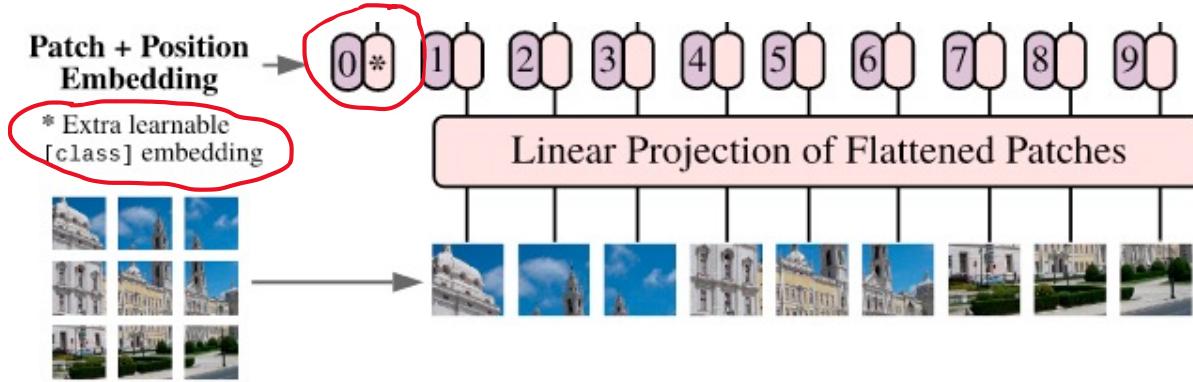
ViT: Putting it all together

1. Divide image into $P \times P$ patches
2. Create sequence of length $N = HW / P^2$
3. Flatten the patches and map to D dimensions with a trainable linear projection
4. Add a learned 1-D position embedding

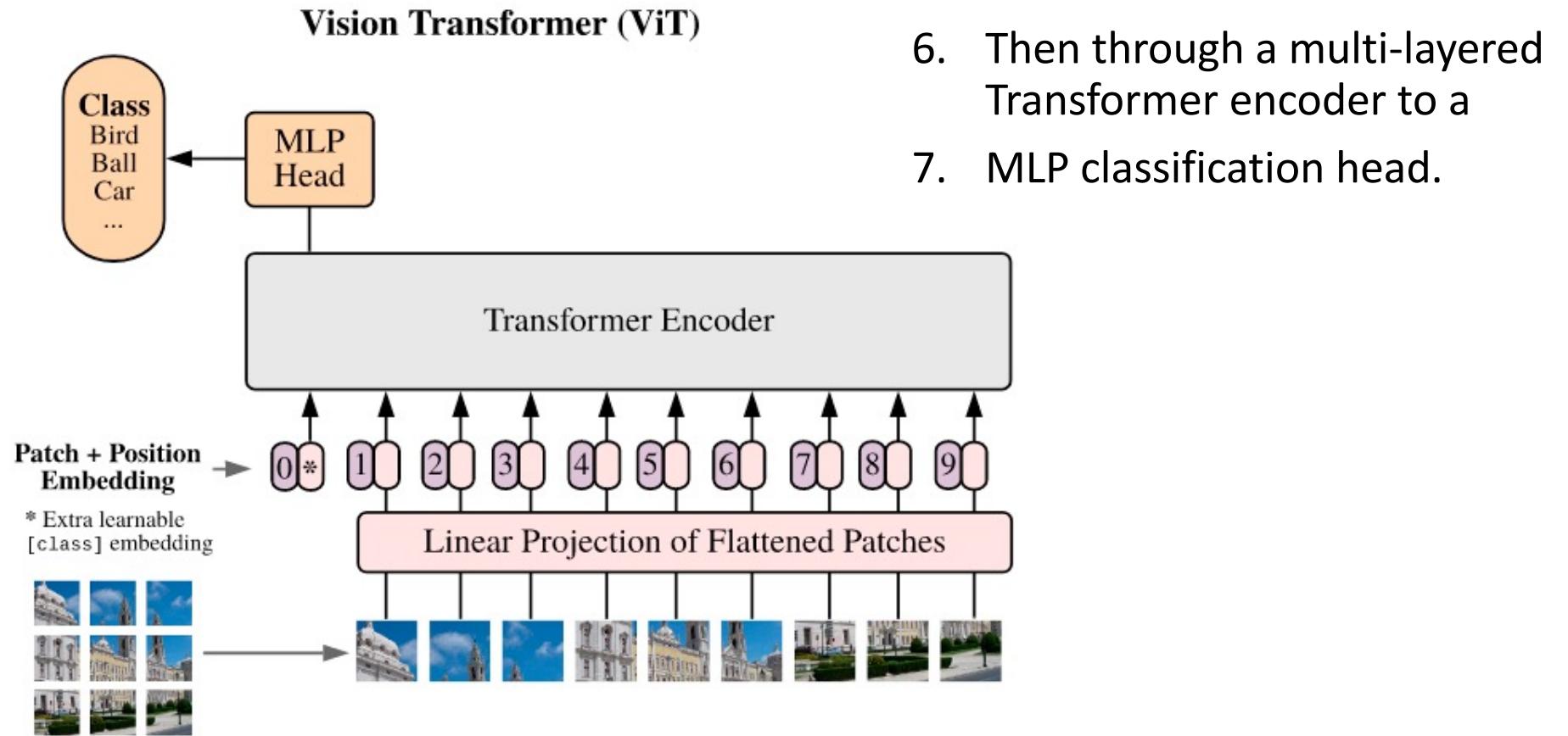


ViT: Putting it all together

1. Divide image into $P \times P$ patches
2. Create sequence of length $N = HW / P^2$
3. Flatten the patches and map to D dimensions with a trainable linear projection
4. Add a learned 1-D position embedding
5. Include a learnable [class] embedding



ViT: Putting it all together



ViT Training Datasets & Model Variants

Dataset	# Classes	# Images
ILSVRC-2012	1K	1.3M
ImageNet-21K	21K	14M
JFT	18K	303M

Model	Layers	Hidden size D	MLP size	Heads	Params	
ViT-Base	12	768	3072	12	86M	Same as BERT
ViT-Large	24	1024	4096	16	307M	Same as BERT
ViT-Huge	32	1280	5120	16	632M	New for ViT

Notation: ViT-L/16 -- "Large" variant with 16×16 input size.

Note: $16 \times 16 \times 3 = 768$

ViT: Image Classification Results

Pre-Trained On

	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	88.55 ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	90.72 ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	99.50 ± 0.06	99.42 ± 0.03	99.15 ± 0.03	99.37 ± 0.06	—
CIFAR-100	94.55 ± 0.04	93.90 ± 0.05	93.25 ± 0.05	93.51 ± 0.08	—
Oxford-IIIT Pets	97.56 ± 0.03	97.32 ± 0.11	94.67 ± 0.15	96.62 ± 0.23	—
Oxford Flowers-102	99.68 ± 0.02	99.74 ± 0.00	99.61 ± 0.02	99.63 ± 0.03	—
VTAB (19 tasks)	77.63 ± 0.23	76.28 ± 0.46	72.72 ± 0.21	76.29 ± 1.70	—
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

ViT: Visualizing Internals

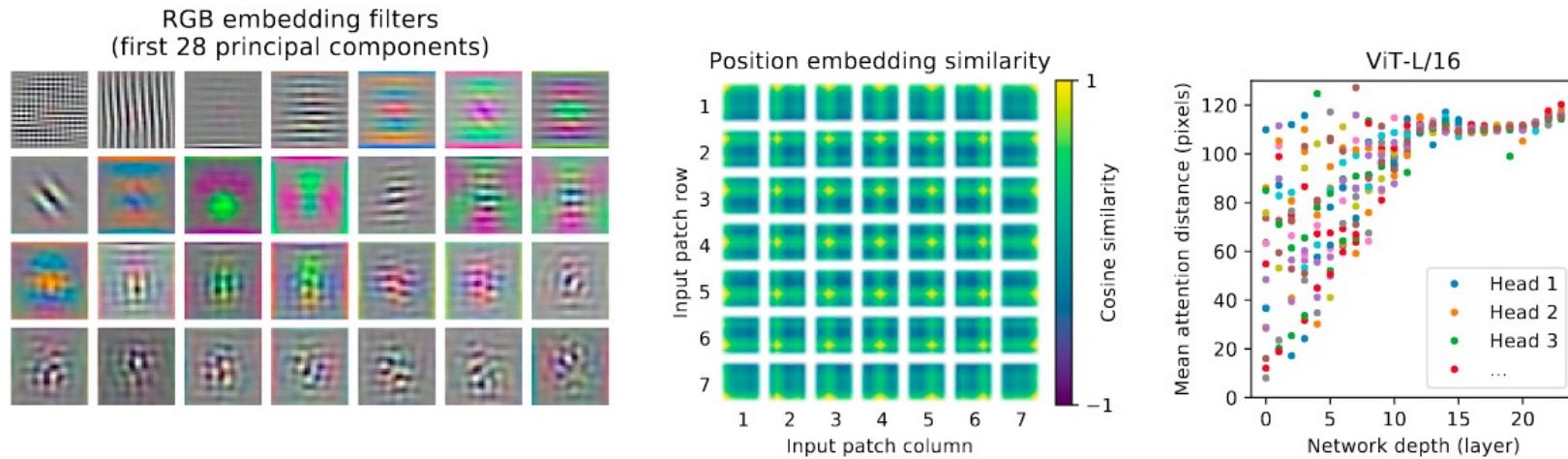


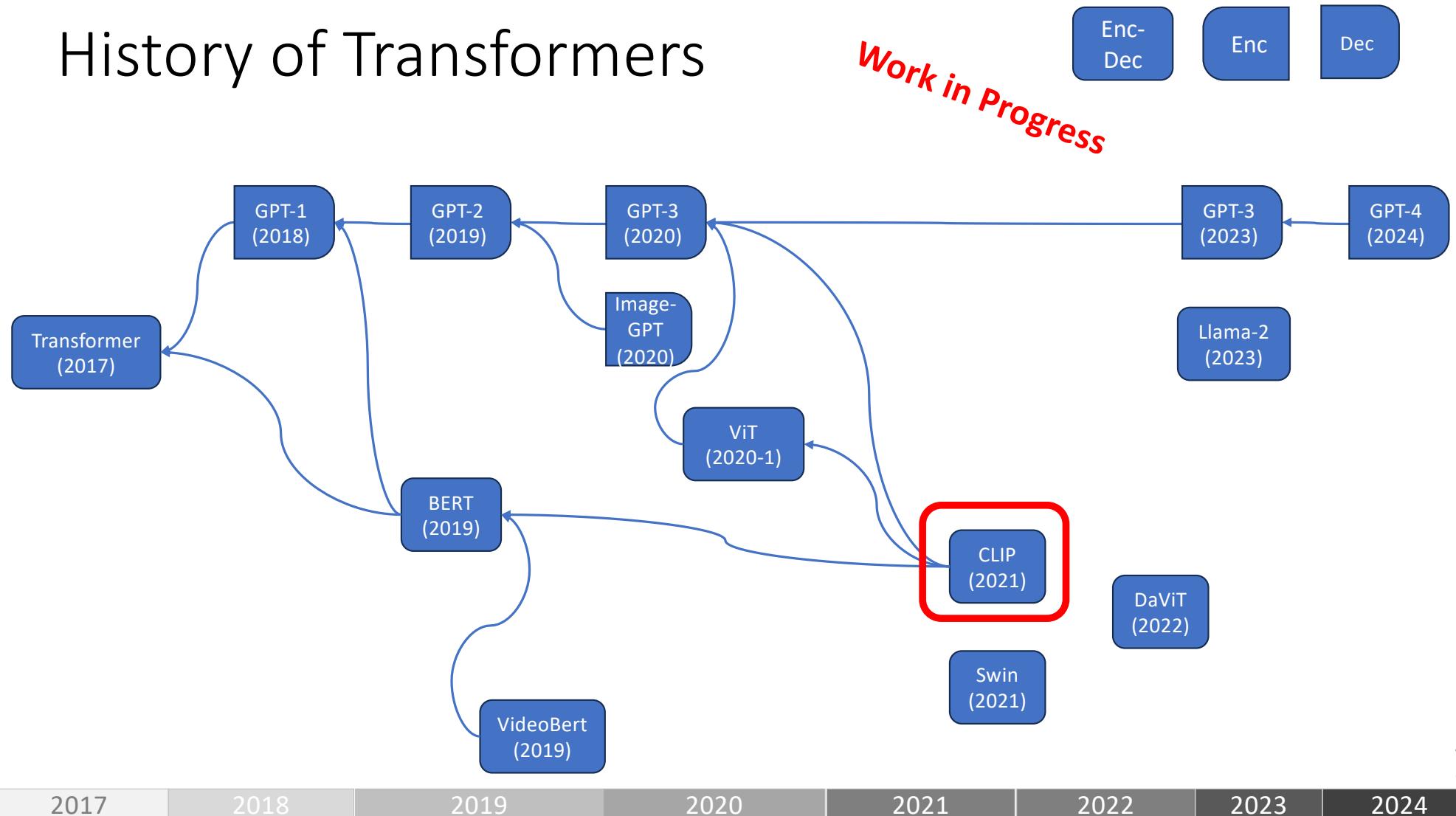
Figure 7: **Left:** Filters of the initial linear embedding of RGB values of ViT-L/32. **Center:** Similarity of position embeddings of ViT-L/32. Tiles show the cosine similarity between the position embedding of the patch with the indicated row and column and the position embeddings of all other patches. **Right:** Size of attended area by head and network depth. Each dot shows the mean attention distance across images for one of 16 heads at one layer. See Appendix D.7 for details.

Scaling Vision Transformers (2022)

- Explore scaling up and down
- Achieves new state-of-the-art on ImageNet top-1: 90.45% *with 2B parameter model*

X. Zhai, A. Kolesnikov, N. Houlsby, and L. Beyer, “Scaling Vision Transformers,” presented at the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 12104–12113. Accessed: Mar. 18, 2024.

History of Transformers



CLIP (2021) – Contrastive Language Image Pretraining

- Learn directly from raw text about images
- Created a new 400m (image, text) pair dataset called WebImageText (WIT) scraped from the internet
- “Simple” pre-training task:
 - Predict which caption goes with which image from scratch on a dataset of 400 million (image, text) pairs
 - Efficient and scalable
 - Learn state-of-the-art image representations from scratch
- Zero-shot transfer to many image classification datasets
- Shows promise for zero-shot transfer for other tasks: e.g. OCR, facial expression recognition, ...

A. Radford *et al.*, “Learning Transferable Visual Models From Natural Language Supervision,” in *Proceedings of the 38th International Conference on Machine Learning*, PMLR, Jul. 2021, pp. 8748–8763.
<https://proceedings.mlr.press/v139/radford21a.html>

CLIP (2021) – Contrastive Language Image Pretraining

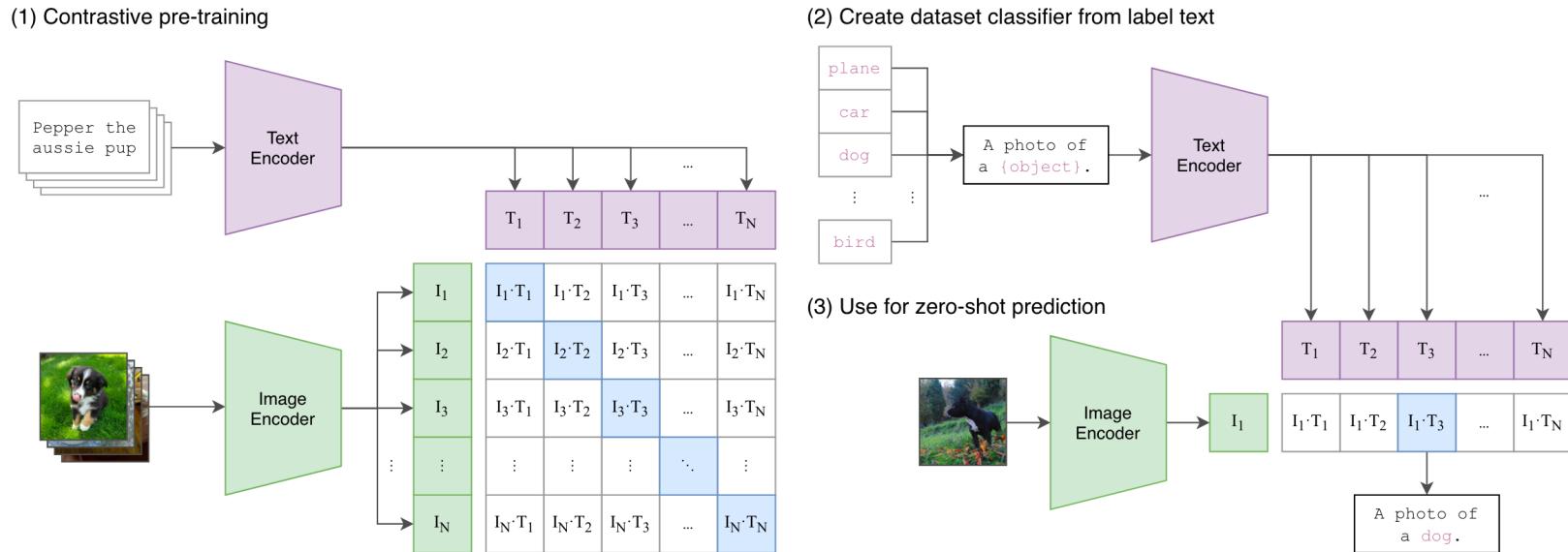
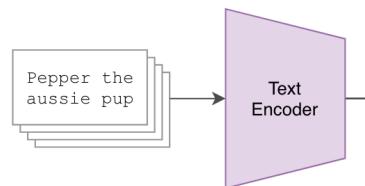


Figure 1. Summary of our approach. While standard image models jointly train an image feature extractor and a linear classifier to predict some label, CLIP jointly trains an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples. At test time the learned text encoder synthesizes a zero-shot linear classifier by embedding the names or descriptions of the target dataset's classes.

CLIP (2021) – Text Encoder



Embedding

- lower-cased byte pair encoding (BPE)
- bracketed with [SOS] and [EOS] tokens

Transformer

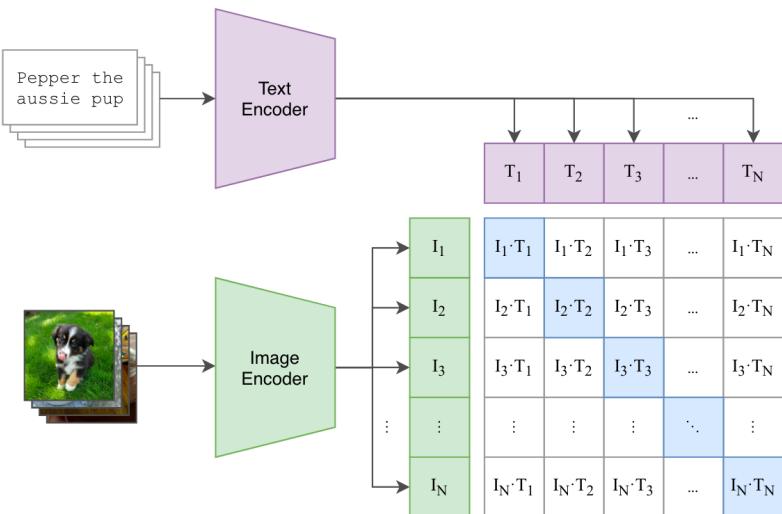
- 12-layer
- 512-wide
- 8 attention heads

CLIP (2021) – Image Encoder

- Trained and compared 5 ResNets and 3 vision transformers
 - ResNet50, ResNet101, RN50x4, x16, x64
 - ViT-B/32, ViT-B/16 and ViT-L/14
- Best model: ViT-L/14@336px
 - e.g. ViT-Large with 336×336 pixel resolution and 14×14 patch resolution
- Found vision transformers ~3x more compute efficient than CLIP ResNets
 - RN50x64 took 18 days to on 592 V100 GPUs
 - ViT took 12 days on 256 V100 GPUs

CLIP (2021) – Contrastive Language Image Pretraining

(1) Contrastive pre-training



```

# image_encoder - ResNet or Vision Transformer
# text_encoder - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l] - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T) #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss = (loss_i + loss_t)/2

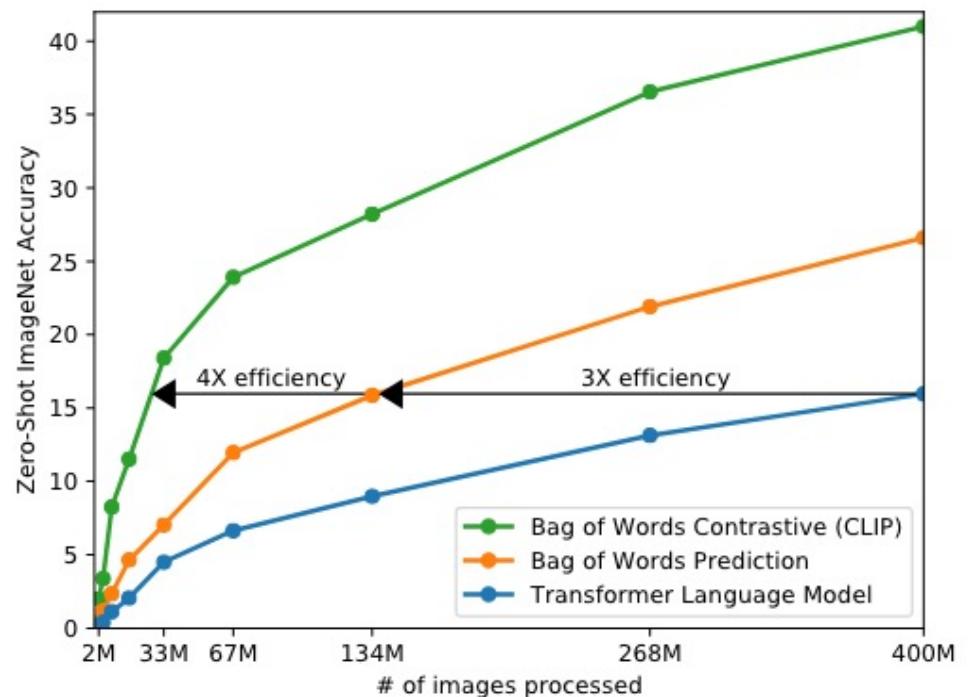
```

Figure 3. Numpy-like pseudocode for the core of an implementation of CLIP.

CLIP (2021) – Contrastive Loss

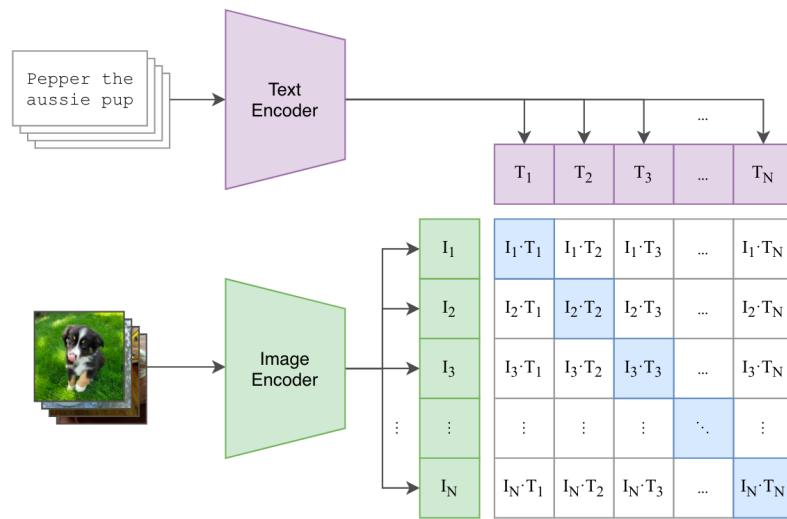
- Initially tried to train to predict caption of image (**blue curve**)
- bag-of-words encoding of same text is 3X more efficient (**orange**) curve
- Contrastive Objective improved another 4X (**green curve**)

Contrastive Loss: Maximize cosine similarity measure between matching (image, text) pairs and simultaneously minimize similarity between non-matching pairs

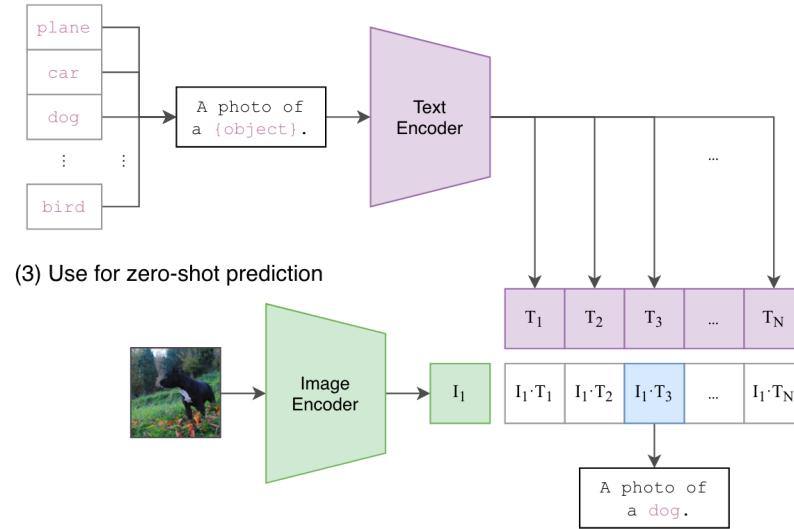


CLIP (2021) – Zero-Shot Image Classification

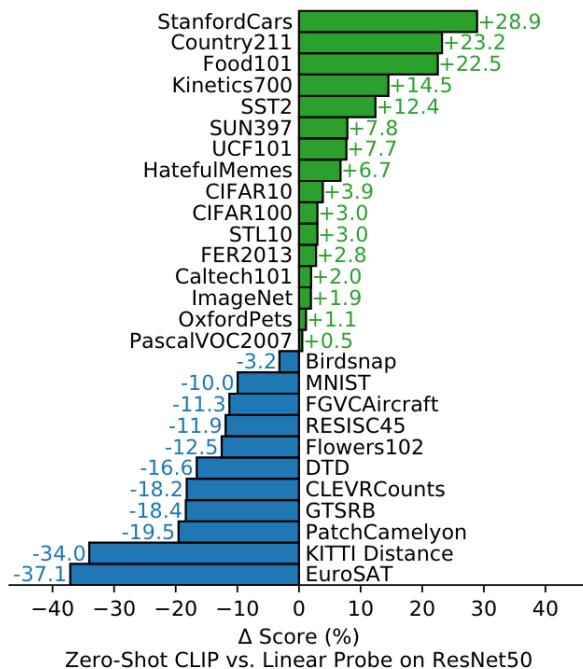
(1) Contrastive pre-training



(2) Create dataset classifier from label text



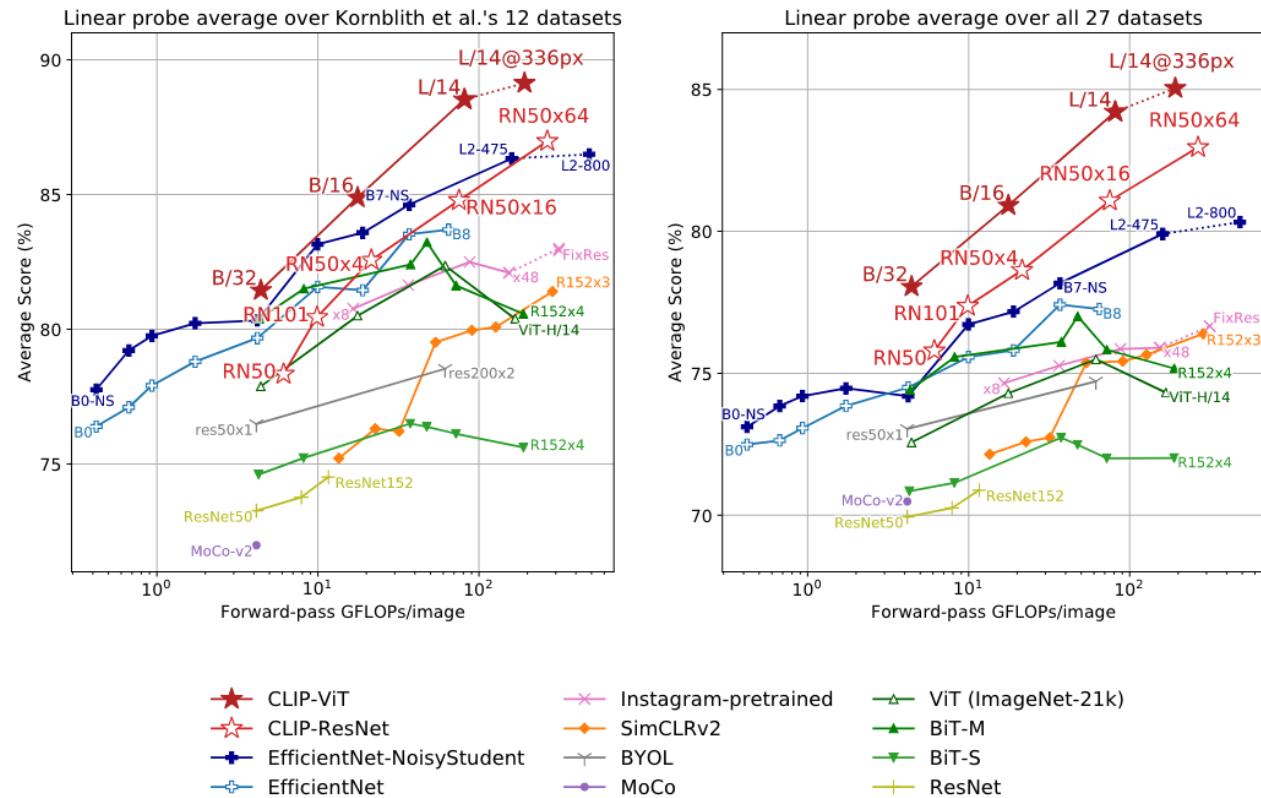
CLIP (2021) – Zero-Shot Image Classification



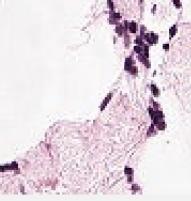
- Evaluated across 27(!!) datasets
- Compared to ResNet50 trained in supervised manner
- Beat ResNet50 on 16 of the 27
- Produced new SoTA on STL10 (99.3%)

Figure 4. Zero-shot CLIP is competitive with a fully supervised baseline. Across a 27 dataset eval suite, a zero-shot CLIP classifier outperforms a fully supervised linear classifier fitted on ResNet50 features on 16 datasets, including ImageNet.

CLIP (2021) – Compute Efficiency

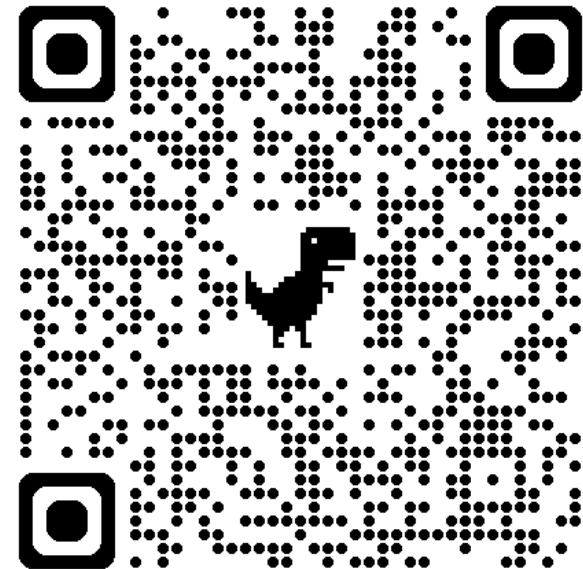


CLIP(2021) – Zero-Shot Classification Examples

<p>Food101 guacamole (90.1%) Ranked 1 out of 101 labels</p>  <p><input checked="" type="checkbox"/> a photo of guacamole, a type of food. ✗ a photo of ceviche, a type of food. ✗ a photo of edamame, a type of food. ✗ a photo of tuna tartare, a type of food. ✗ a photo of hummus, a type of food.</p>	<p>SUN397 television studio (90.2%) Ranked 1 out of 397 labels</p>  <p><input checked="" type="checkbox"/> a photo of a television studio. ✗ a photo of a podium indoor. ✗ a photo of a conference room. ✗ a photo of a lecture room. ✗ a photo of a control room.</p>
<p>Youtube-BB airplane, person (89.0%) Ranked 1 out of 23 labels</p>  <p><input checked="" type="checkbox"/> a photo of a airplane. ✗ a photo of a bird. ✗ a photo of a bear. ✗ a photo of a giraffe. ✗ a photo of a car.</p>	<p>EuroSAT annual crop land (46.5%) Ranked 4 out of 10 labels</p>  <p><input checked="" type="checkbox"/> a centered satellite photo of permanent crop land. ✗ a centered satellite photo of pasture land. ✗ a centered satellite photo of highway or road. <input checked="" type="checkbox"/> a centered satellite photo of annual crop land. ✗ a centered satellite photo of brushland or shrubland.</p>
<p>PatchCamelyon (PCam) healthy lymph node tissue (77.2%) Ranked 2 out of 2 labels</p>  <p><input checked="" type="checkbox"/> this is a photo of lymph node tumor tissue <input checked="" type="checkbox"/> this is a photo of healthy lymph node tissue</p>	<p>ImageNet-A (Adversarial) lynx (47.9%) Ranked 5 out of 200 labels</p>  <p>✗ a photo of a fox squirrel. ✗ a photo of a mongoose. ✗ a photo of a skunk. ✗ a photo of a red fox. <input checked="" type="checkbox"/> a photo of a lynx.</p>

Next Time

- Retrieval Augmented Generation (RAG) and other LLM “cognitive” architecture aspects
- ...



[Link](#)