Deep Generative Models: Transformers for Text

Fall Semester 2024

René Vidal
Director of the Center for Innovation in Data Engineering and Science (IDEAS)
Rachleff University Professor, University of Pennsylvania
Amazon Scholar & Chief Scientist at NORCE

Penn
University of Pennsylvania

Autoregressive Models

- Many kinds of models
  - Markov Chains
  - Hidden Markov Models
  - Markov Random Fields
  - Linear Dynamical Systems
  - Recurrent Neural Networks
  - **Transformers**

- Last lecture
  - **Word Embedding**
  - **Positional Encoding**
  - **Attention Mechanism**
  - **Multi-head Attention**
  - **Attention Visualization**

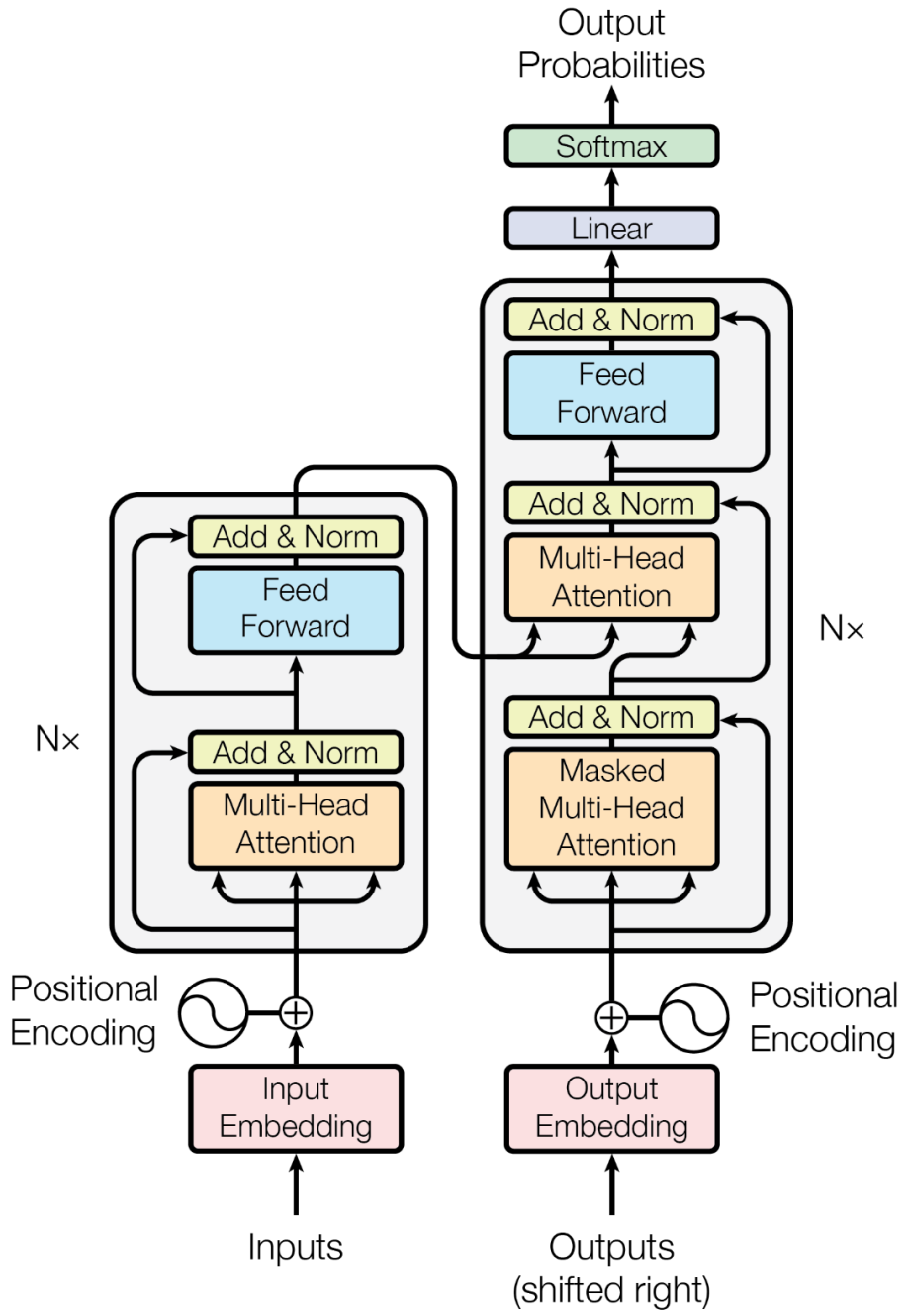


Figure 1: The Transformer - model architecture.

From Last Lecture

- [NeurIPS 2017] Attention is all you need: the Transformer that contains **Encoder Block** and **Decoder Block**.
- The design allows engineers to stack multiple blocks all together in large-scale training, which enables the emergence of foundation models.

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

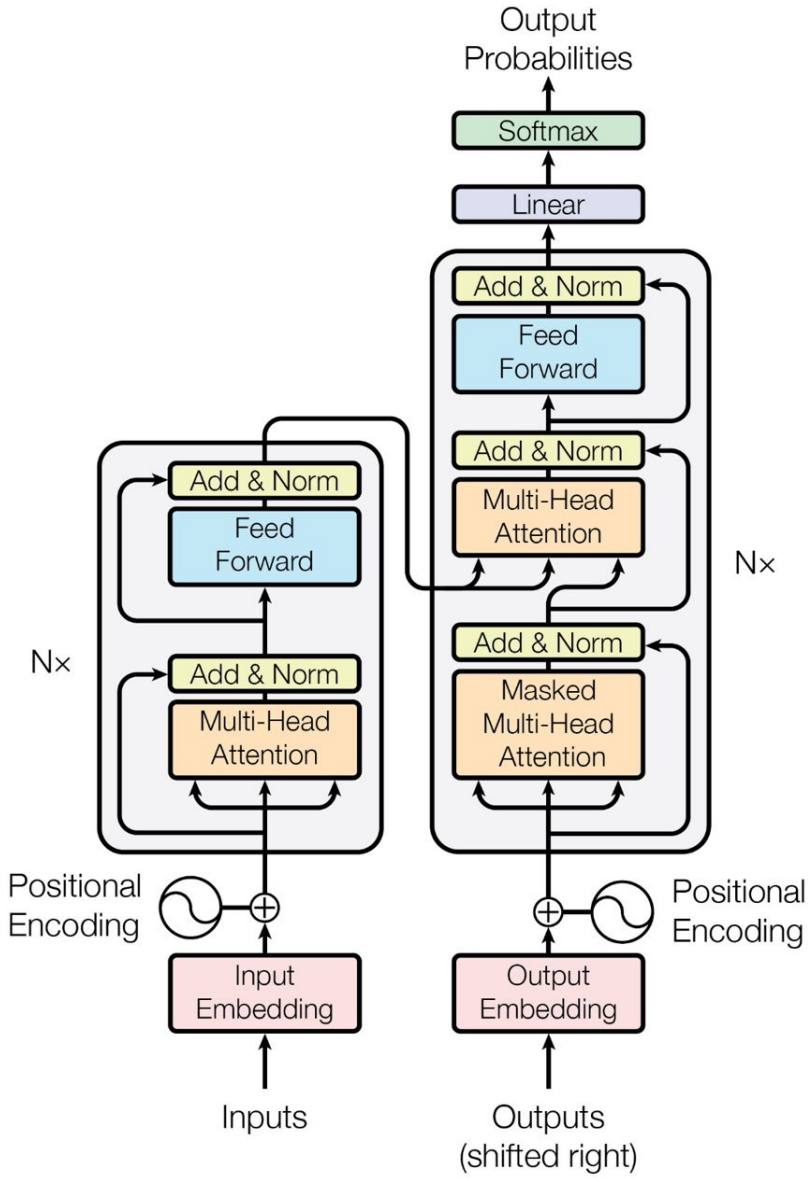
Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

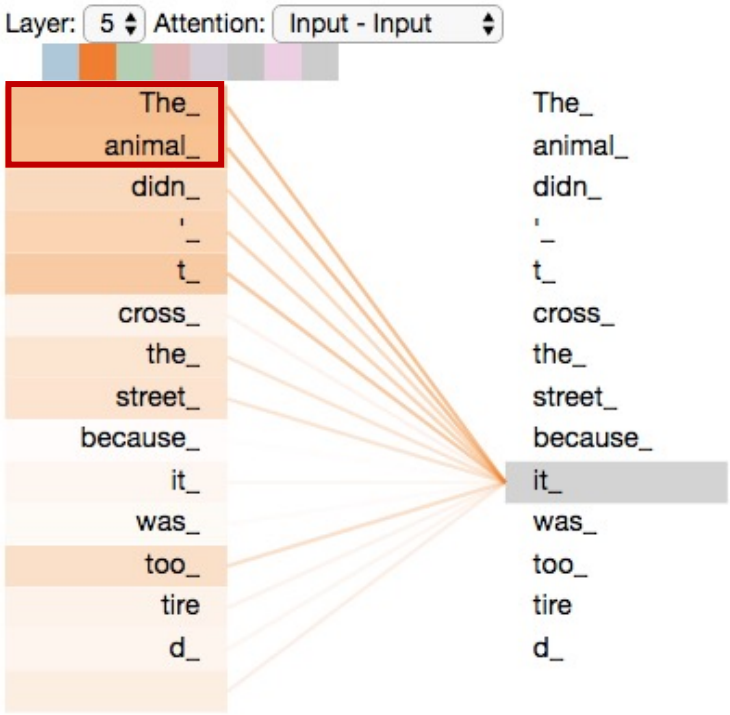
Łukasz Kaiser* Google Brain lukaszkaiser@google.com

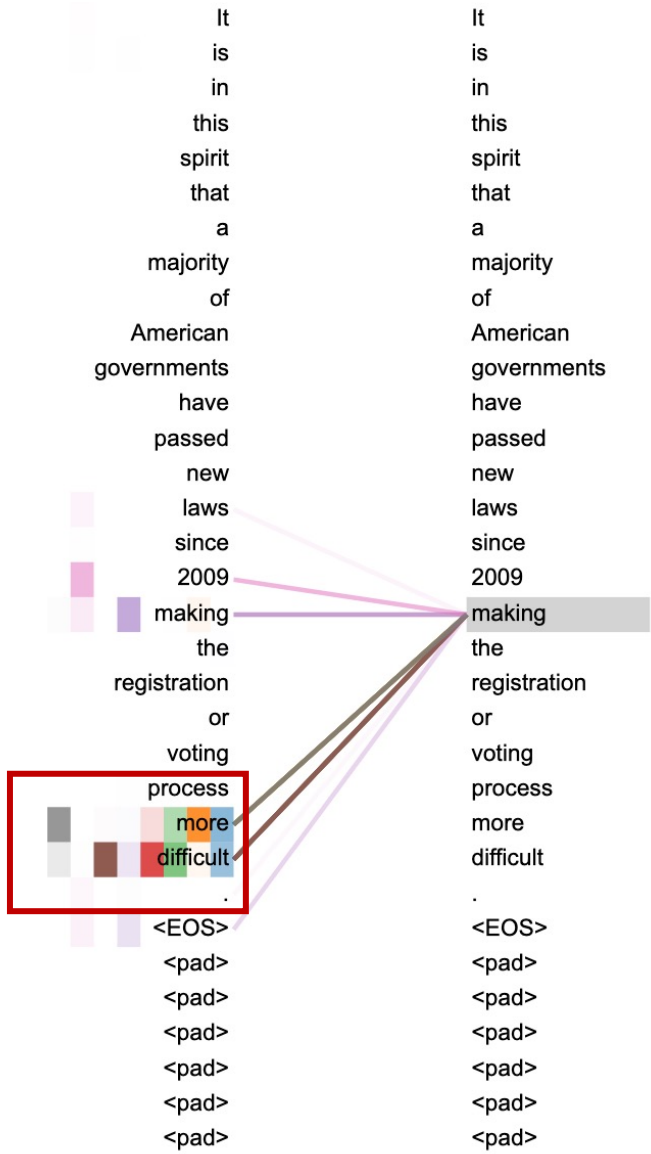
Illia Polosukhin* ‡ illia.polosukhin@gmail.com



Attention Visualization: Long distance dependency

- Earlier we saw the sentence: “The animal didn't cross the street because **it** was too tired.”
- What does "**it**" in this sentence refer to? The visualization of self-attention shows the association of "it" with beginning parts like "The animal".

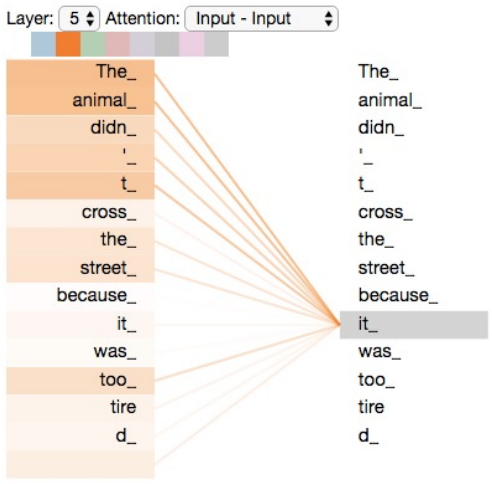


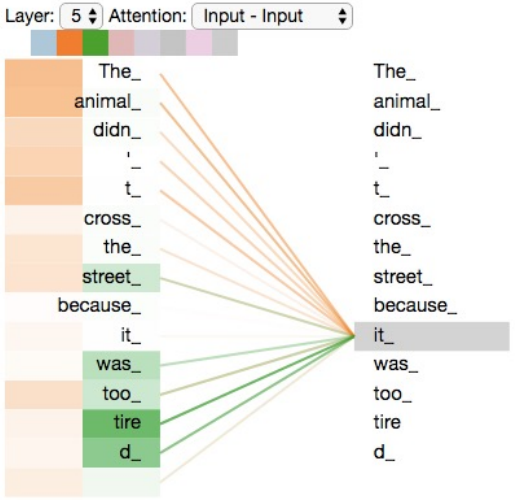


- On the right we see another visualization showing how different words in a longer sentence relate to each other.
- Check out this interactive visualization.

Attention: Attention from Different Heads

- Attention heads can specialize to capture various dependencies, such as syntactic and semantic relationships.
- This allows the model to attend to different types of causalities between words in a sentence.

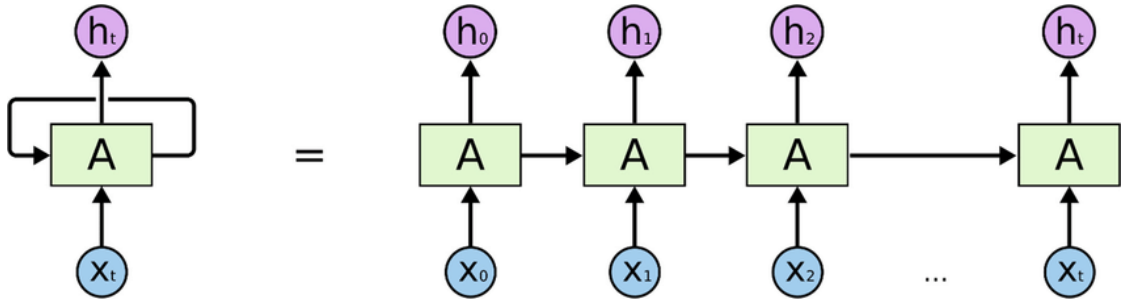




RNNs vs. Transformers

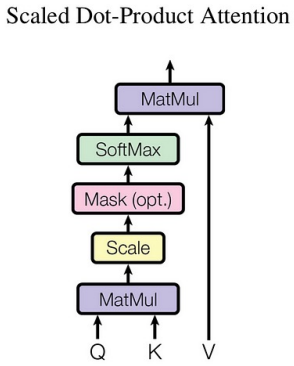
Recurrent Neural Network

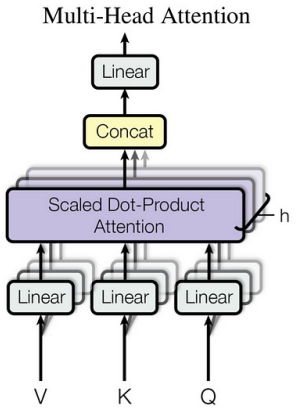
- Handle Sequential Data
- Learn Sequential Dependencies
- Each time step depends on the previous one



Transformers

- Handle Sequential Data
- Learn Sequential Dependencies
- Use self-attention to capture global context





RNNs vs. Transformers

Recurrent Neural Network	Transformers

• (-) Learning long-range dependences is challenging due to recurrent structure
  • Can be aided by specialized architectures like LSTM and GRU
  • Suffer from training issues such as vanishing gradient
	• (+) Attention mechanism better captures long-range dependences
	  • Able to handle both global context and local context
	  • No vanishing gradient issues

• (-) Hard to scale up because each time step depends on the previous one	• (+) Processes tokens in parallel, makes it efficient for training on GPUs
• (+) Usually smaller number of parameters, does not require lots of data to train	• (-) Usually large number of parameters, requires lots of data to train

Evolutions of Transformers

Natural Language Processing:

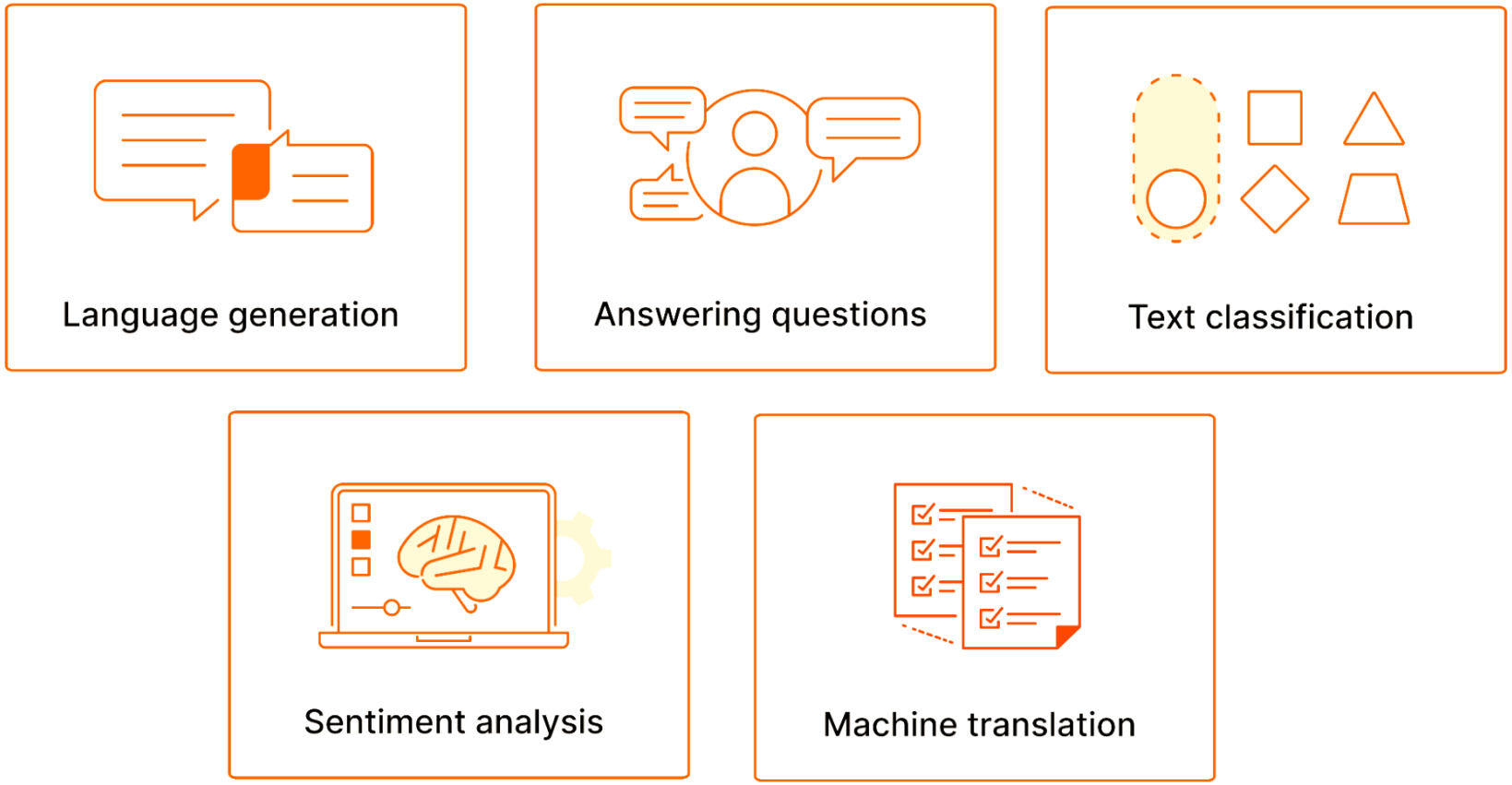
- **BERT (Bidirectional Encoder Representations from Transformers)**
- GPT (Generative Pre-trained Transformer)
- RoBERTa (Robustly Optimized Bert Pre-training)
- T5 (Text-to-Text Transfer Transformer)

When it comes to Vision:

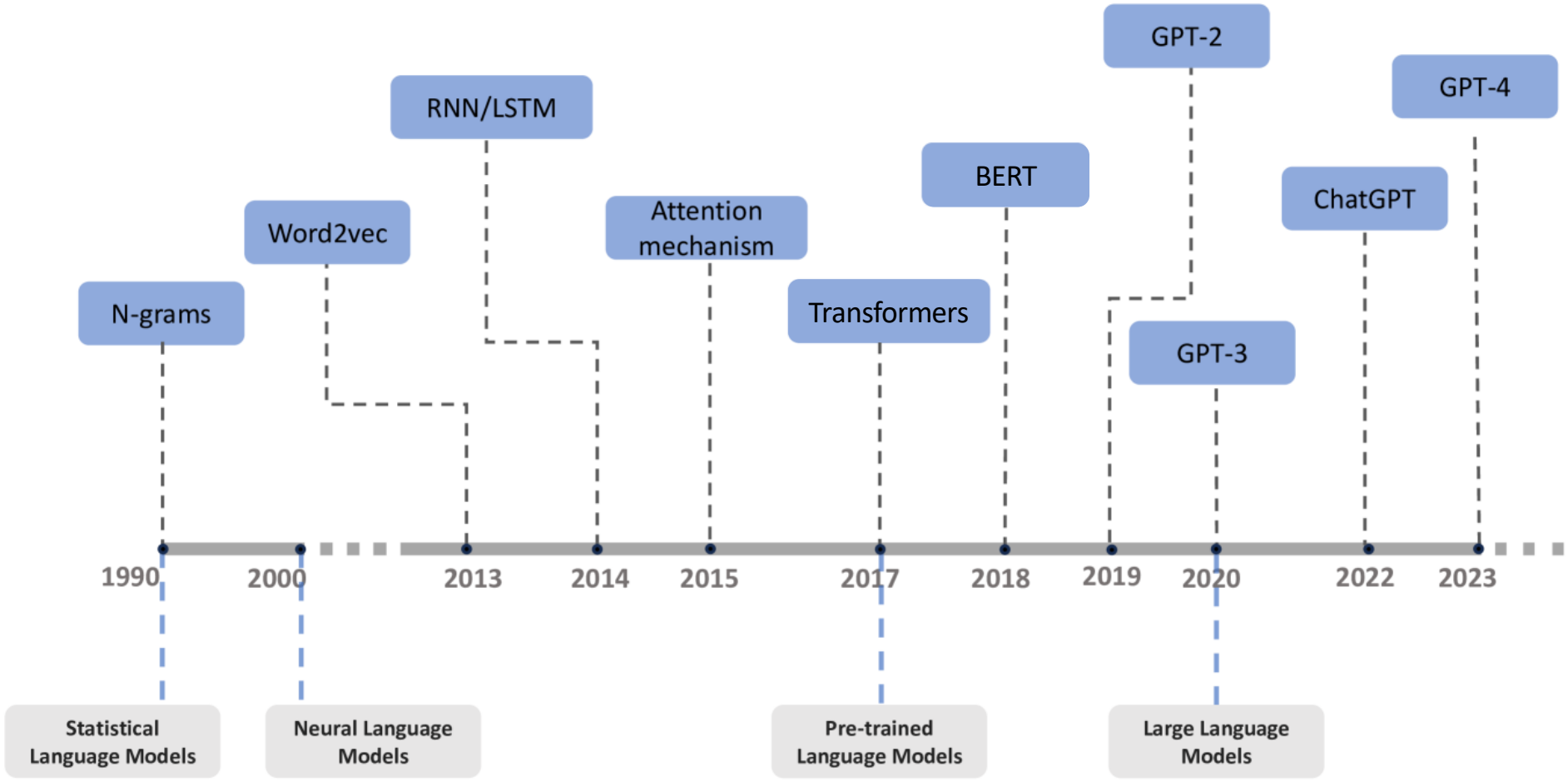
- **Vision Transformer**
- Swin Transformer, Pyramid Vision Transformer

Natural Language Processing Tasks

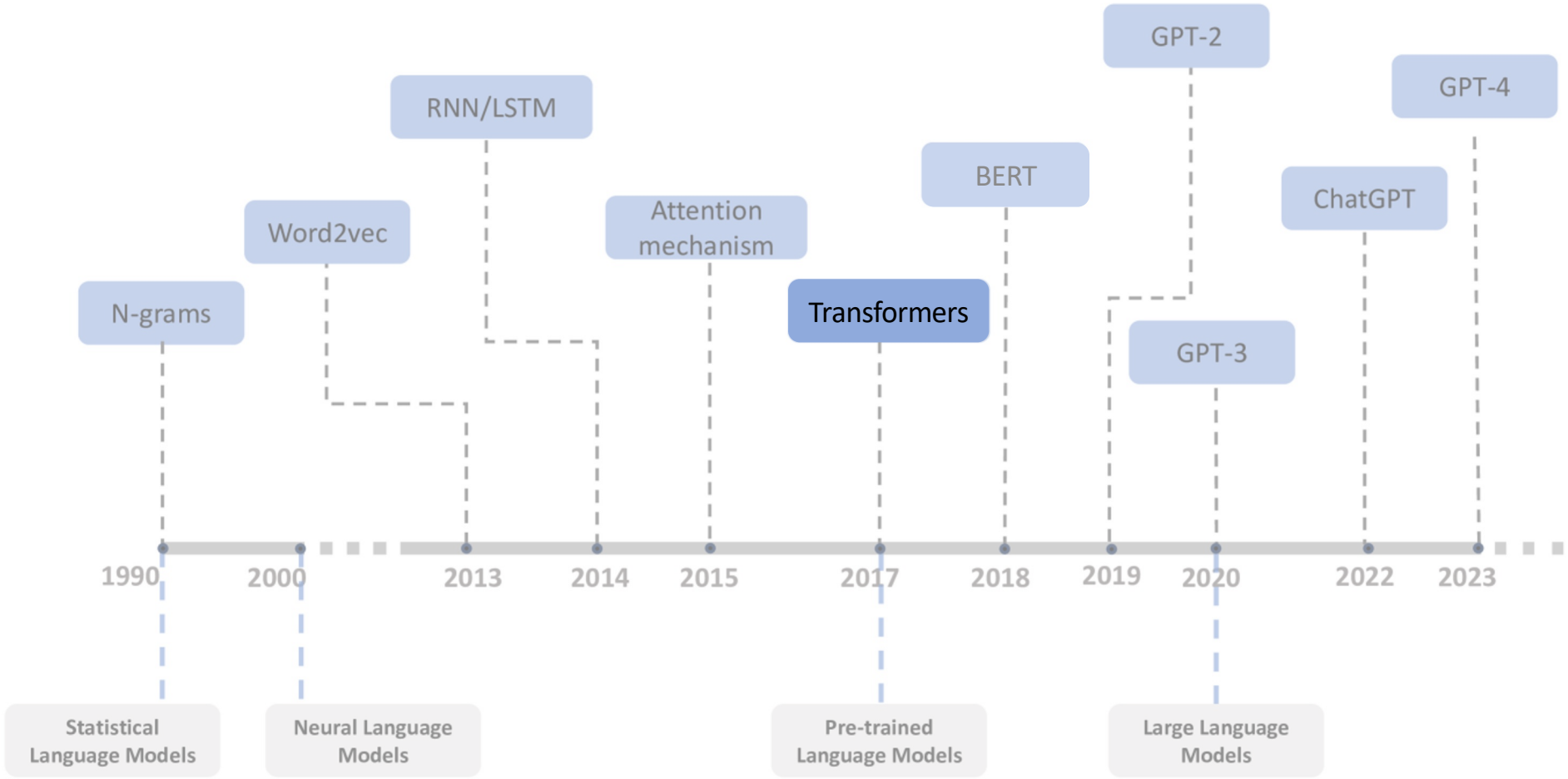
- Natural Language Processing is the process through which AI is taught to understand the rules and syntax of language, programmed to develop complex algorithms to represent those rules and then made to use those algorithms to carry out specific tasks like these.



Natural Language Modeling History

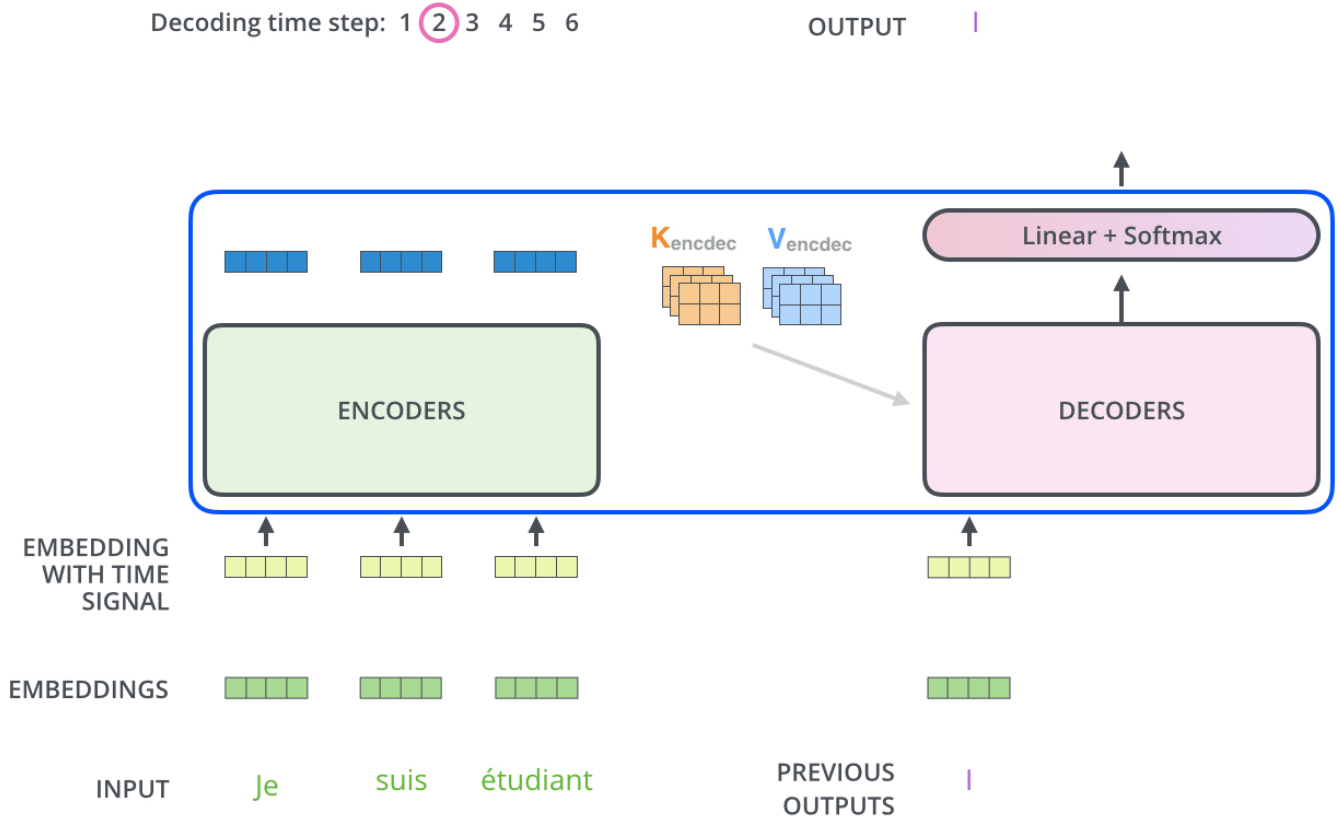


Natural Language Modeling History



Transformer for NLP Tasks

- The Transformer was originally designed as an encoder-decoder model for sequence-to-sequence tasks, like translation.
  - During inference, it generates one word/token at a time and **feeds it back into the model as input for the next word/token.**
  - The process continues until a special end-of-sequence token (<EOS>) is produced or a maximum length is reached.
- Such a paradigm might not be optimal for sentence-level tasks (e.g., sentiment analysis, QA, etc.)
- Can we create a model that incorporates a more comprehensive view of context and serves as a versatile foundation for various NLP tasks?



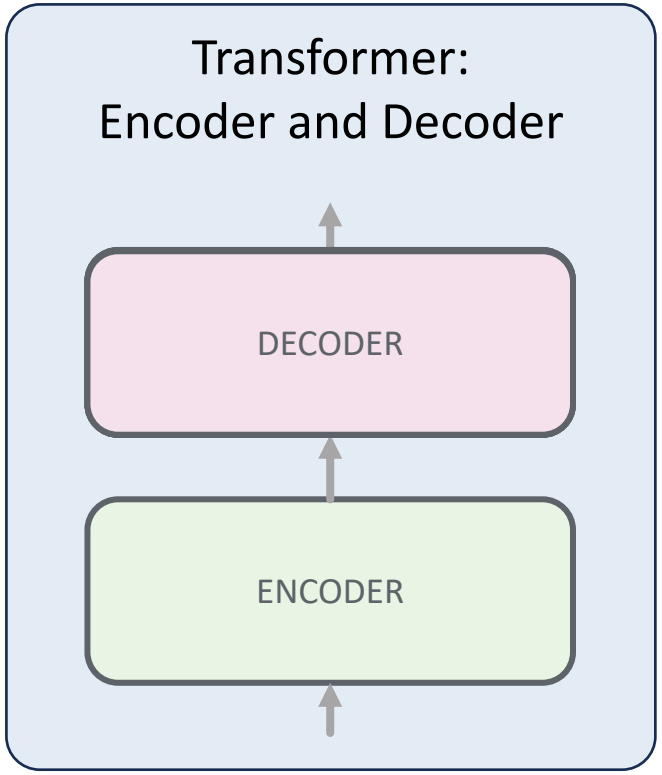
The Need for a General Language Model

- **Goal:** Develop a language model with broad understanding capabilities.
- **Why:** This model can be adapted to various NLP tasks easily, we don't have to retrain a model from scratch every time.
- **How?** This requires language understanding.
  - Pretrain a model that learns universal language patterns.
  - Finetune the language model to learn specific tasks.
- **Pretrained Transformer Models** focuses on the idea of pre-training on vast amounts of generic text data to capture universal language patterns.
  - This allows the models to learn rich contextual representations that can be fine-tuned on specific tasks with minimal data.

"For several years, people have been getting very good results pre-training [Deep Neural Networks] as a language model and then fine-tuning on some downstream NLP tasks (question answering, natural language inference sentiment analysis)"
– BERT author

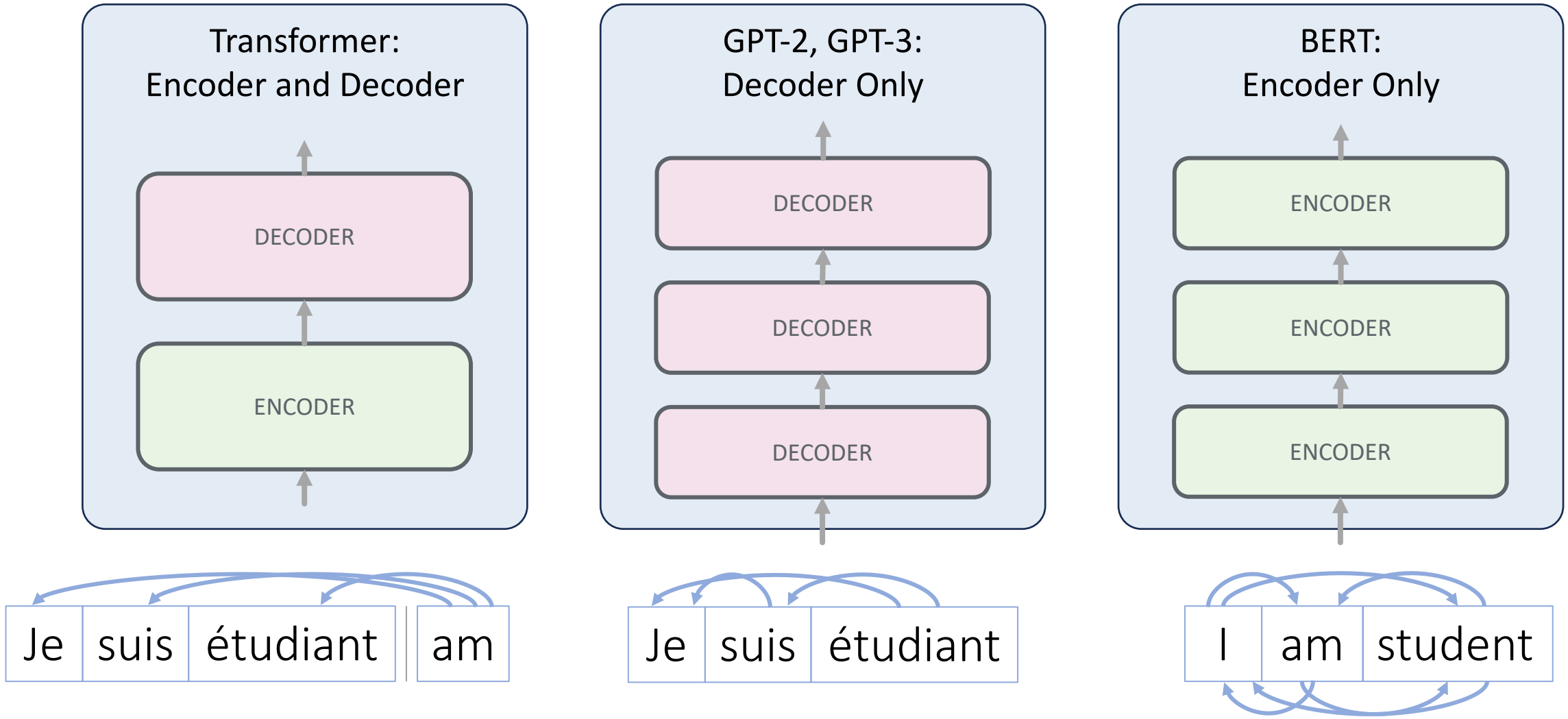
Pretrained Transformer Models

- The transformer architecture inspired the creation of pretrained transformers like BERT and GPT. Both models build directly on the original Transformer architecture but apply it differently.

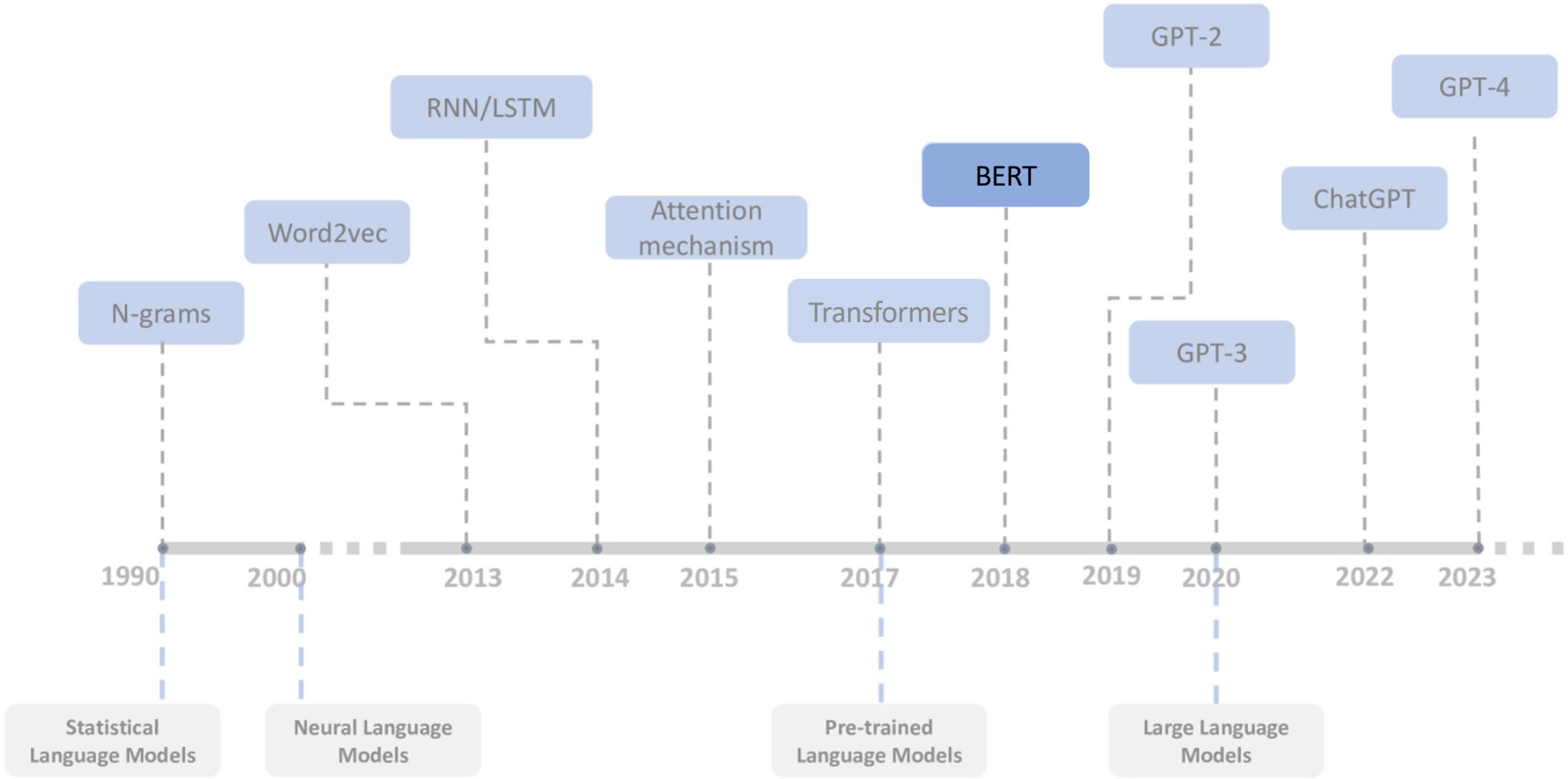


Pretrained Transformer Models

- The transformer architecture inspired the creation of pretrained transformers like BERT and GPT. Both models build directly on the original Transformer architecture but apply it differently.

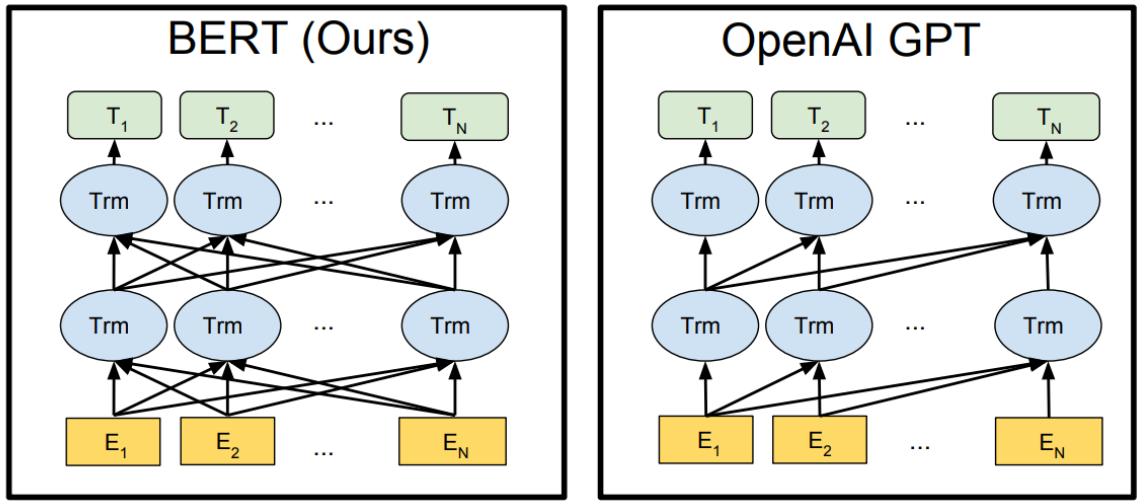


Natural Language Modeling History



Bidirectional Encoder Representations from Transformers (BERT)

- BERT is a transformer-based model whose language model is conditioned on both left and right context (bi-directionally).
  - Models like GPT process text left-to-right (uni-directionally).



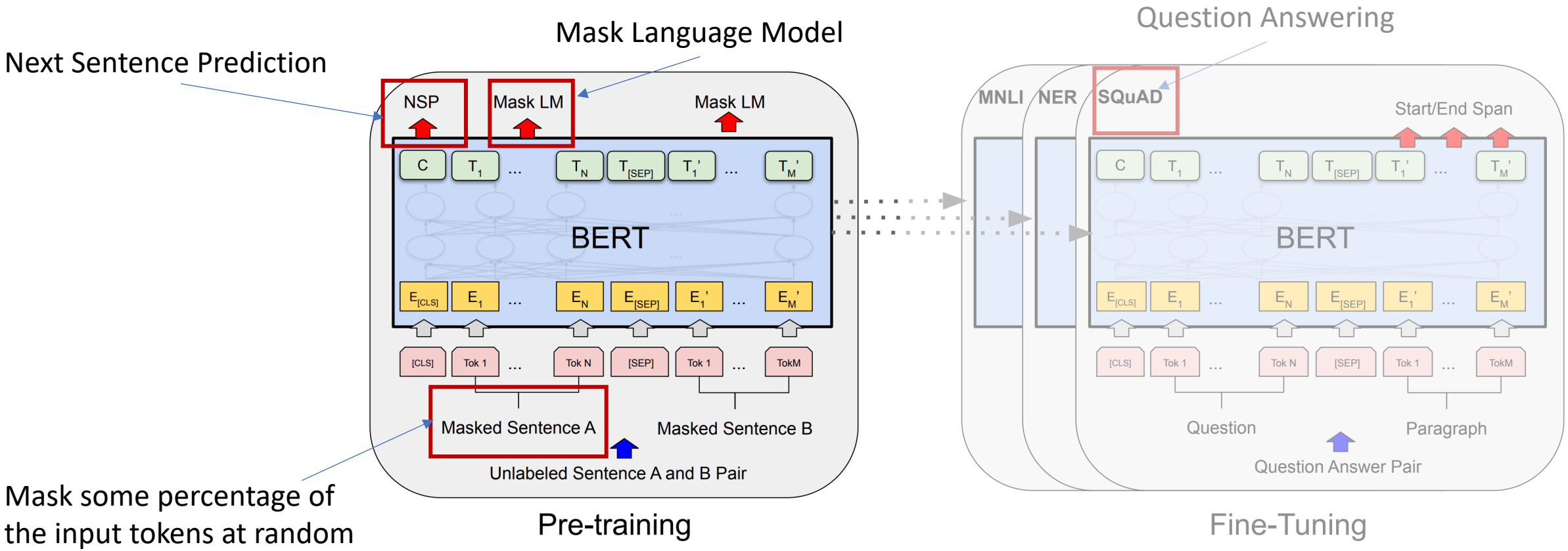
- BERT works on both sentence-level and token-level tasks.

- BERT Training:
  - Pretraining: Understand the language
    - Trained on entire Wikipedia and BookCorpus.
  - Finetuning: Learn specific NLP tasks
    - Can be finetuned easily for downstream tasks.
    - Targeted at multi-task objective.

	$BERT_{BASE}$	$BERT_{LARGE}$	Transformer
Layers	12	24	6
Feedforward networks (hidden units)	768	1024	512
Attention heads	12	16	8

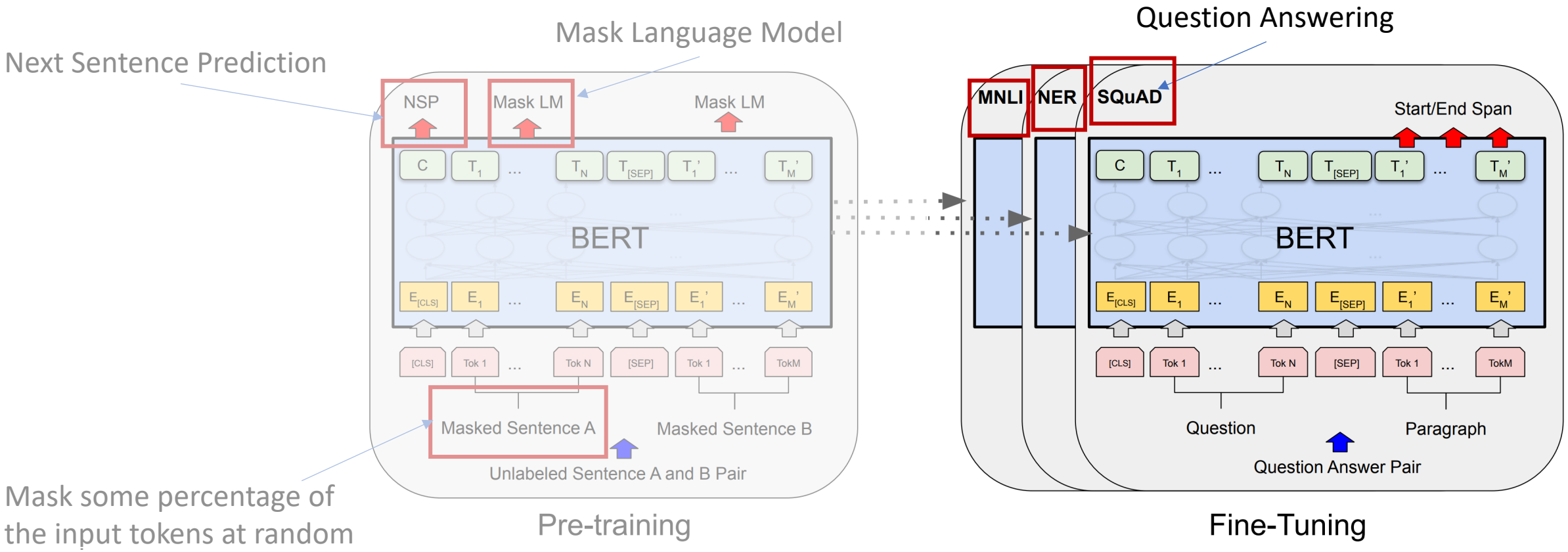
Bidirectional Encoder Representations from Transformers (BERT)

- **Pre-training:** BERT is pre-trained using two main objectives:
  - **Masked Language Modeling (MLM):** Randomly masks words in a sentence and trains the model to predict them, encouraging it to learn context from both directions.
  - **Next Sentence Prediction (NSP):** Trains the model to understand the relationship between two sentences, useful for tasks like question answering and sentence coherence.



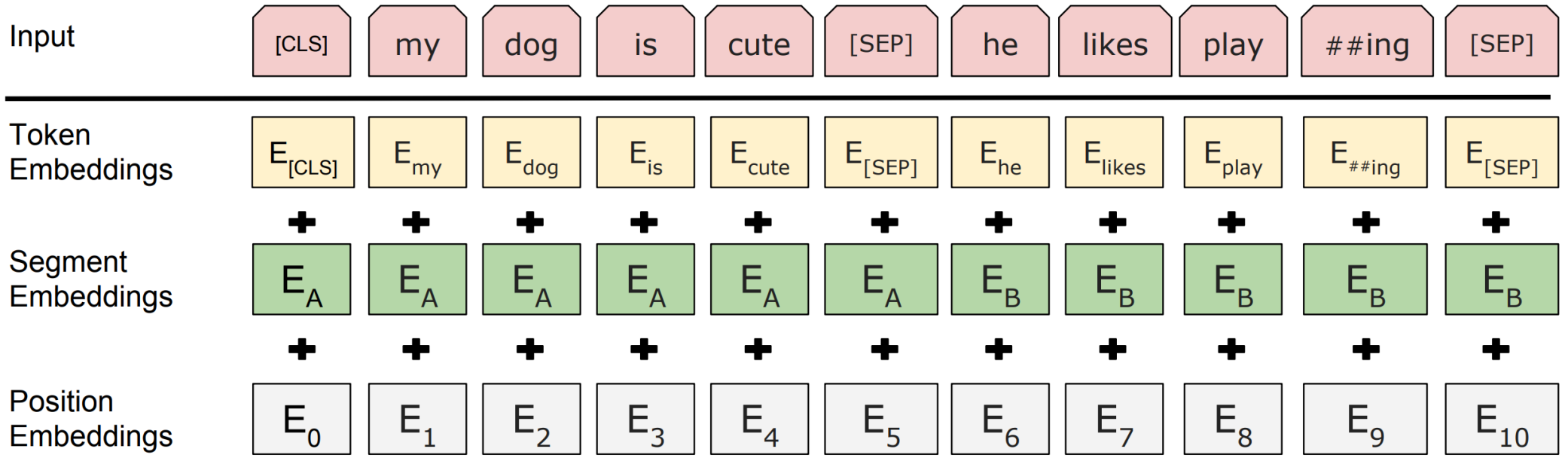
Bidirectional Encoder Representations from Transformers (BERT)

- **Fine-tuning:** Once pre-trained, BERT is fine-tuned on specific NLP tasks with minimal adjustments, making it highly versatile for different applications.
  - Apart from output layers, the same architectures are used in both pre-training and fine-tuning.



Bidirectional Encoder Representations from Transformers (BERT)

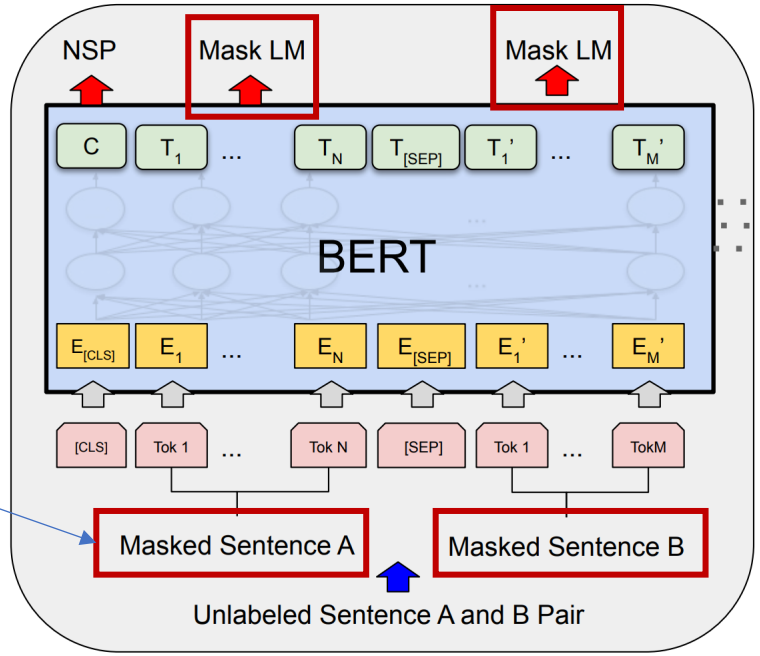
- BERT's input representation combines three types of embeddings: token embeddings (words), segment embeddings (sentence distinction), and position embeddings (word order within the sentence).
- Each input sequence begins with a special [CLS] token, whose final hidden state is used for classification tasks.
- The [SEP] token marks the end of each sentence or segment.



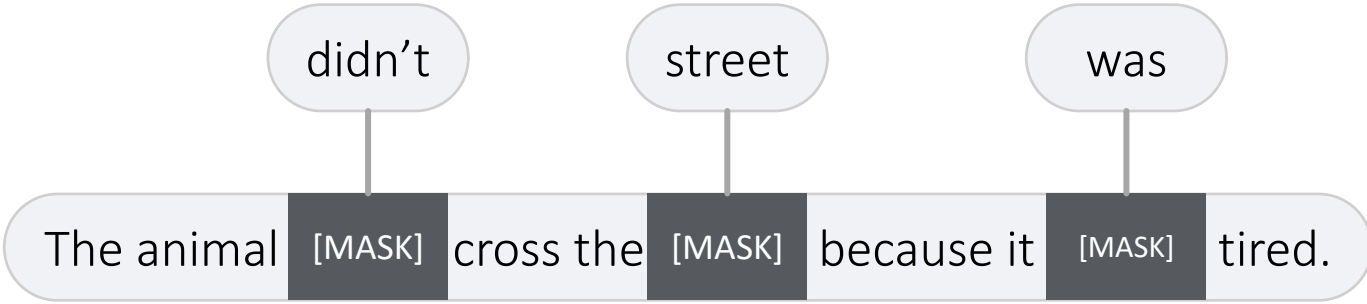
BERT Pre-training Stage: Masked Language Modeling (MLM)

- BERT masks a portion of the input words and trains to predict these masked words using context from the sequence.
  - Typically, 15% of the tokens are selected for masking. Among the candidates: 80% chance to be masked, 10% chance to be altered, 10% chance remain the same.
- The task is reconstructing the token sequence given the masked one.

Mask some percentage of the input tokens at random

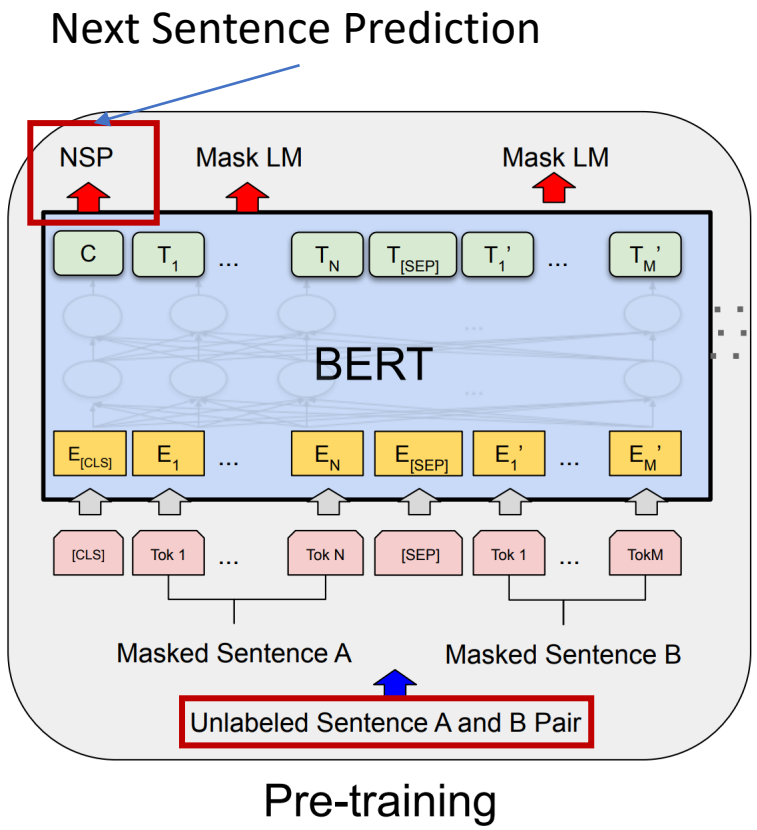


Pre-training



BERT Pre-training Stage: **N**ext **S**entence **P**rediction (**NSP**)

- NSP helps BERT understand relationships between sentences, which is essential for tasks like question answering and natural language inference.
- Given pair of sentences, it predicts whether the second sentence naturally follows the first. It learns to classify each pair as “Is Next” or “Not Next”.



Sentence A	She studied hard for the final exam.
Sentence B	Thus, she passed with flying colors.

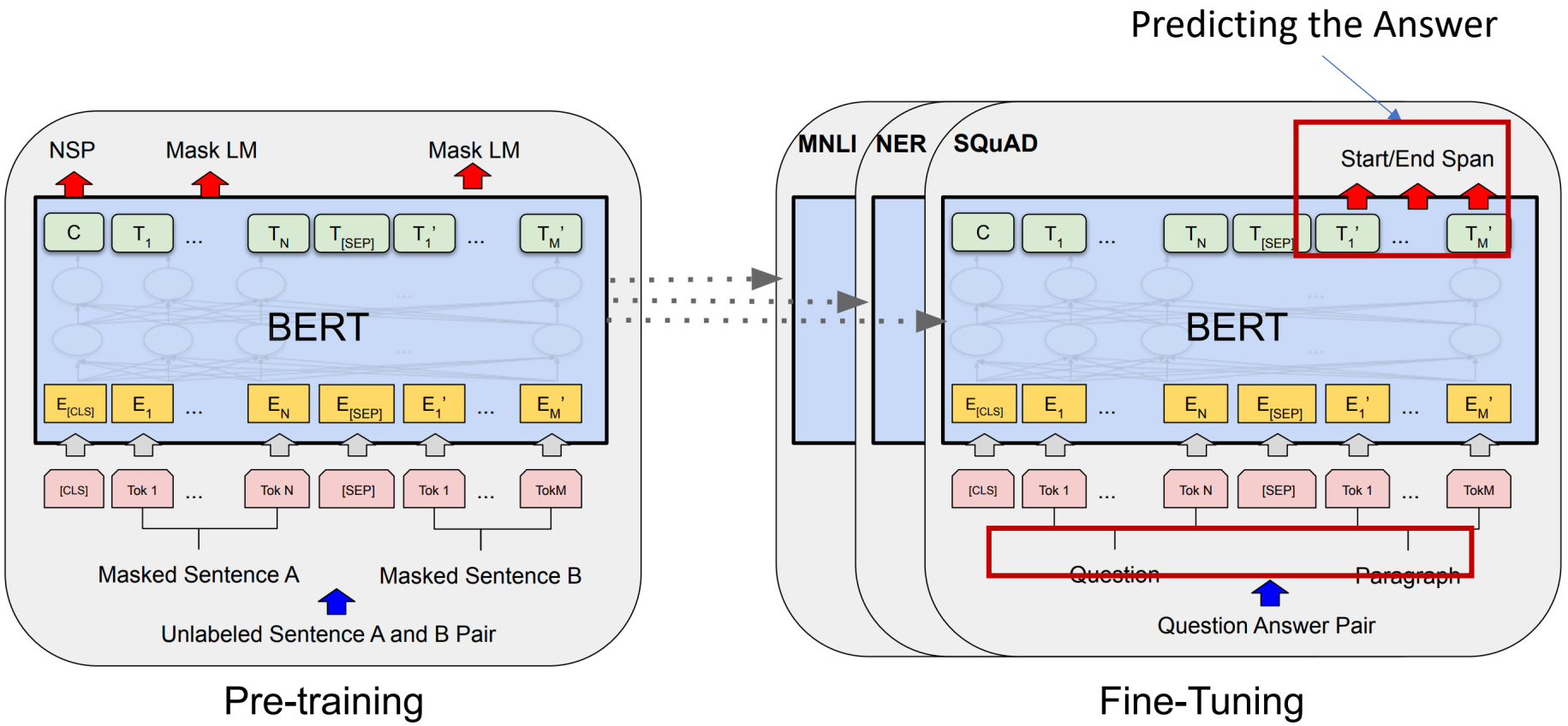
IsNext

Sentence A	The children were playing in the park.
Sentence B	He then started cooking the soup.

NotNext

BERT Fine-tuning Stage

- Fine-tuning adapts BERT's general language understanding (learned during pre-training) to specific NLP tasks like **sentiment analysis**, **question answering**, and **named entity recognition (NER)**.
  - For instance, given a QA dataset that consists of training samples (Question, Paragraph, Answer), we naturally utilize the pre-trained weights to fine-tune BERT.



Deep Generative Models: Transformers for Vision

Fall Semester 2024

René Vidal
Director of the Center for Innovation in Data Engineering and Science (IDEAS)
Rachleff University Professor, University of Pennsylvania
Amazon Scholar & Chief Scientist at NORCE

Penn
University of Pennsylvania

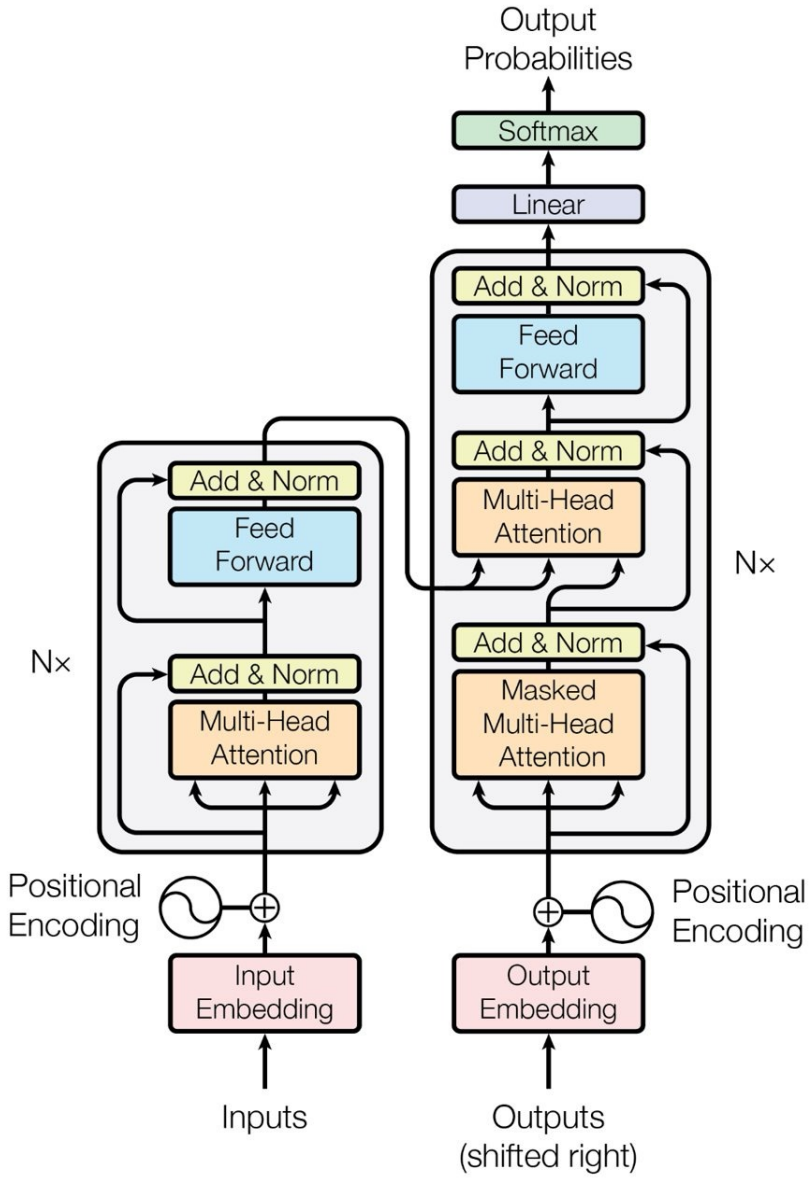
Last Two Lectures -> Today's Lecture

Natural Language Processing:

- Attention is all you need: Enc-Dec Transformer
- BERT (Bidirectional Encoder Representations from Transformers)
- GPT (Generative Pre-trained Transformer)
- RoBERTa (Robustly Optimized Bert Pre-training)
- T5 (Text-to-Text Transfer Transformer)

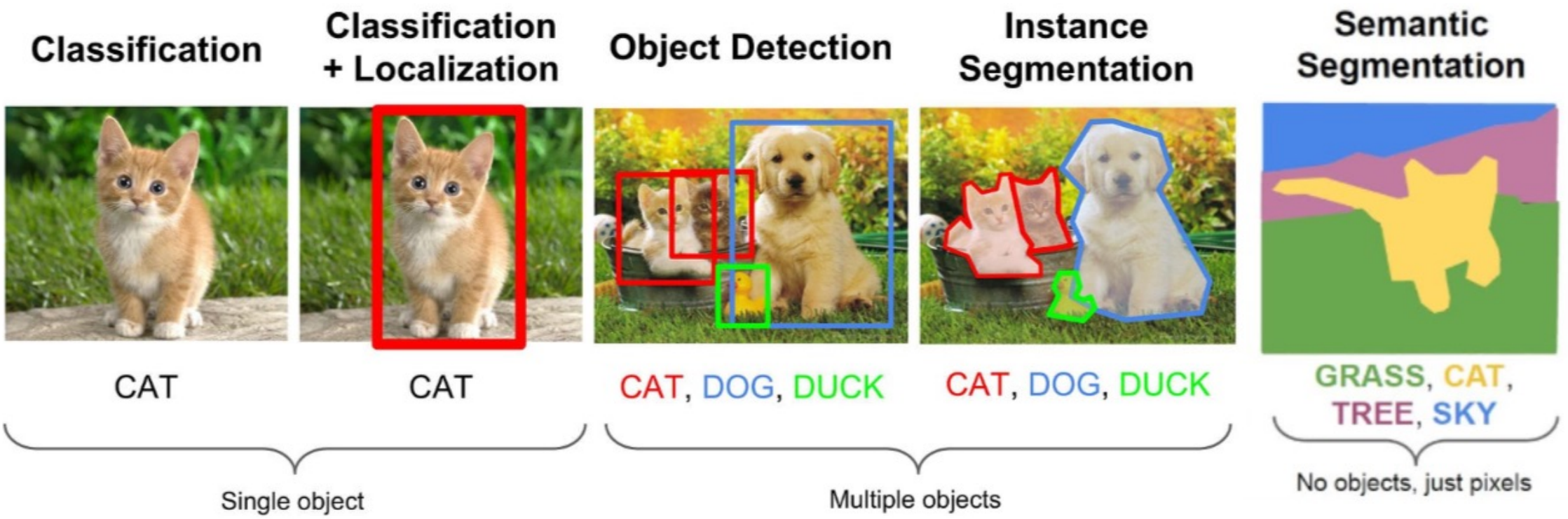
Computer Vision:

- Generative Pretraining from Pixels
- Vision Transformer
- Swin Transformer, Pyramid Vision Transformer

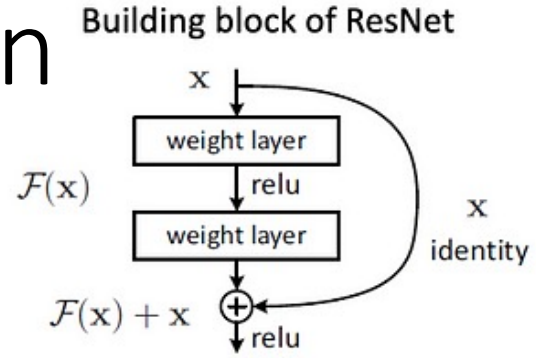


Computer Vision Tasks

- Computer Vision is the field of AI that enables machines to interpret and make decisions based on visual data. It uses a variety of algorithms to recognize, classify, and understand images or videos.
- Some key tasks in Computer Vision are:



Neural Networks in Vision



1998
LeNet

2010
Pascal VOC

ImageNet
2009

AlexNet
(deepCNN)
2012

2014
VGG
R-CNN
GoogLeNet
(Inception)

Fast R-CNN
ResNet
2015

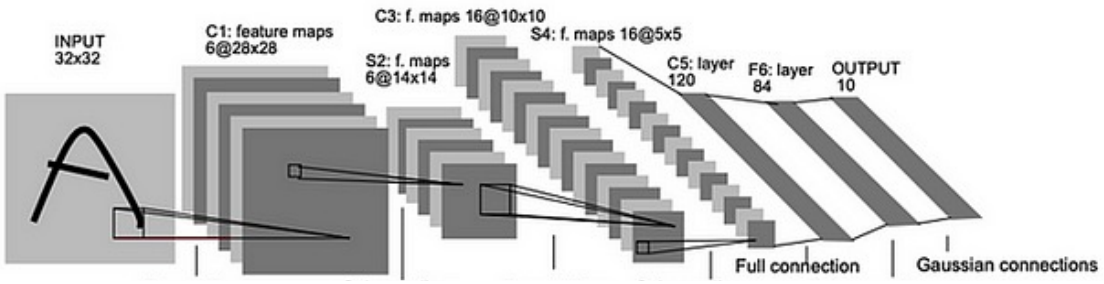
2016
SSD
YOLO
Inception-v4

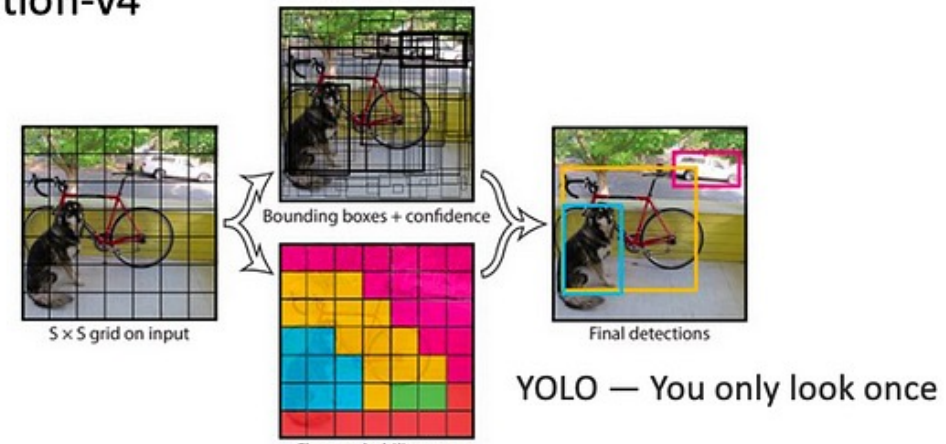
RevNet
YOLOv2
Transformer
2017

2018
YOLOv3
BERT

2020
Image GPT

Vision
Transformer (ViT)
2021

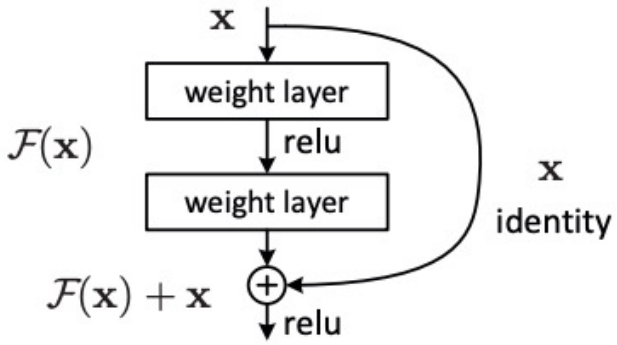




YOLO — You only look once

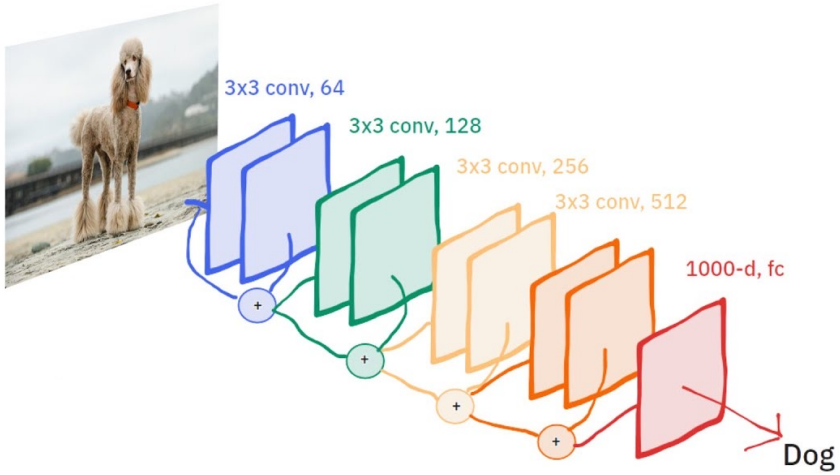
Pre-2019: Convolution Structures Dominated

- In large-scale image recognition (e.g., ImageNet competitions), convolutional residual learning (e.g., ResNet and ResNeXt) architectures were still state of the art up to 2019.



Deep Residual Learning for Image Recognition

Kaiming He Xiangyu Zhang Shaoqing Ren Jian Sun
Microsoft Research
{kahe, v-xiangz, v-shren, jiansun}@microsoft.com

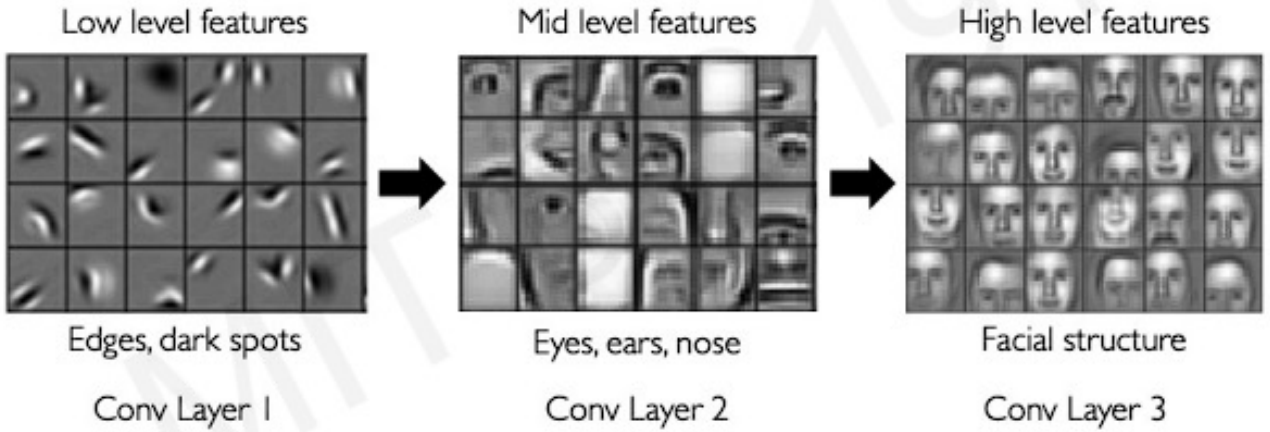


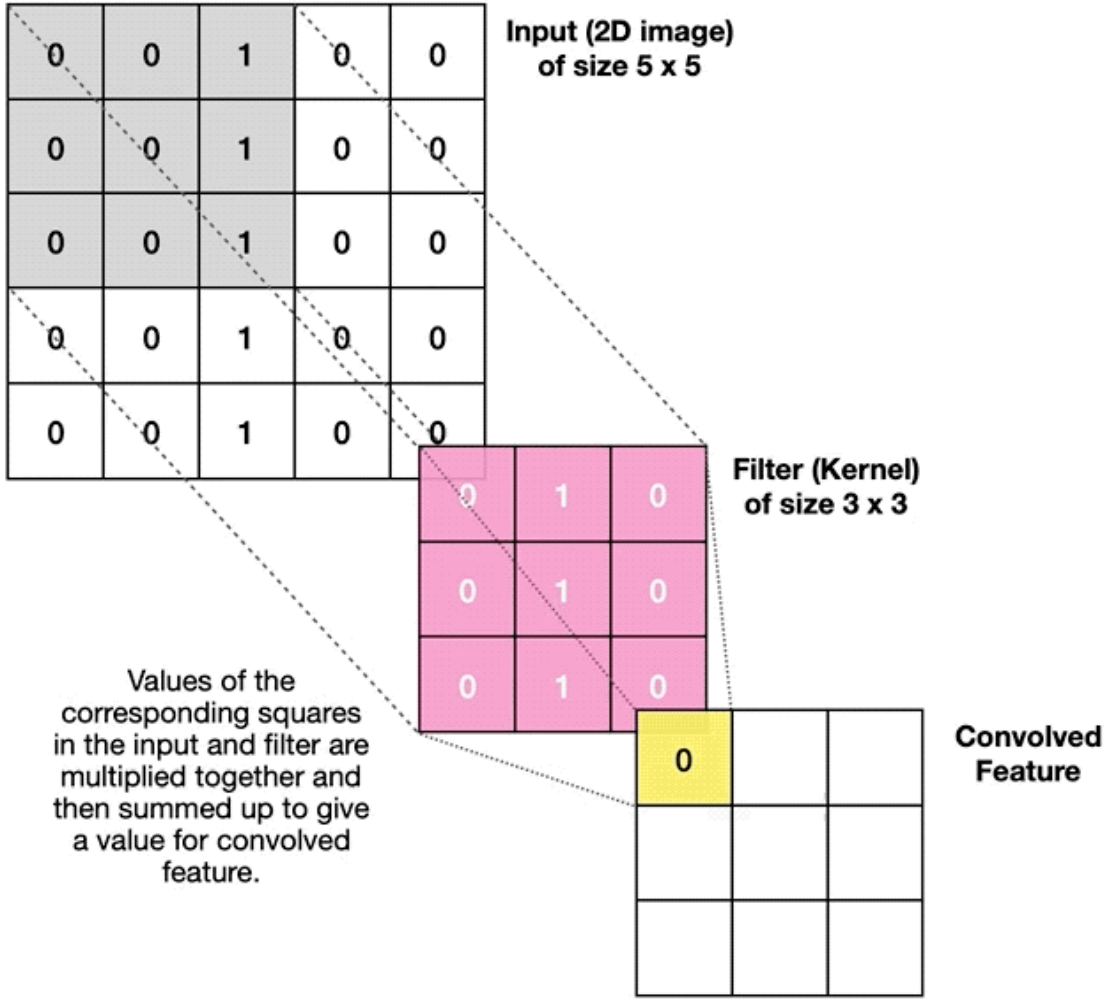
Aggregated Residual Transformations for Deep Neural Networks

Saining Xie[1] Ross Girshick[2] Piotr Dollár[2] Zhuowen Tu[1] Kaiming He[2]
[1]UC San Diego [2]Facebook AI Research
{s9xie,ztu}@ucsd.edu {rbg,pdollar,kaiminghe}@fb.com

Pre-2019: Convolution Structures Dominated

- In large-scale image recognition (e.g., ImageNet competitions), convolutional residual learning (e.g., ResNet and ResNeXt) architectures were still state of the art up to 2019.
  - A convolution operation involves sliding a filter or kernel across the image. Each position results in a weighted sum of the pixel values covered by the filter, producing a convolved feature map, highlighting various features such as edges, textures, and patterns.





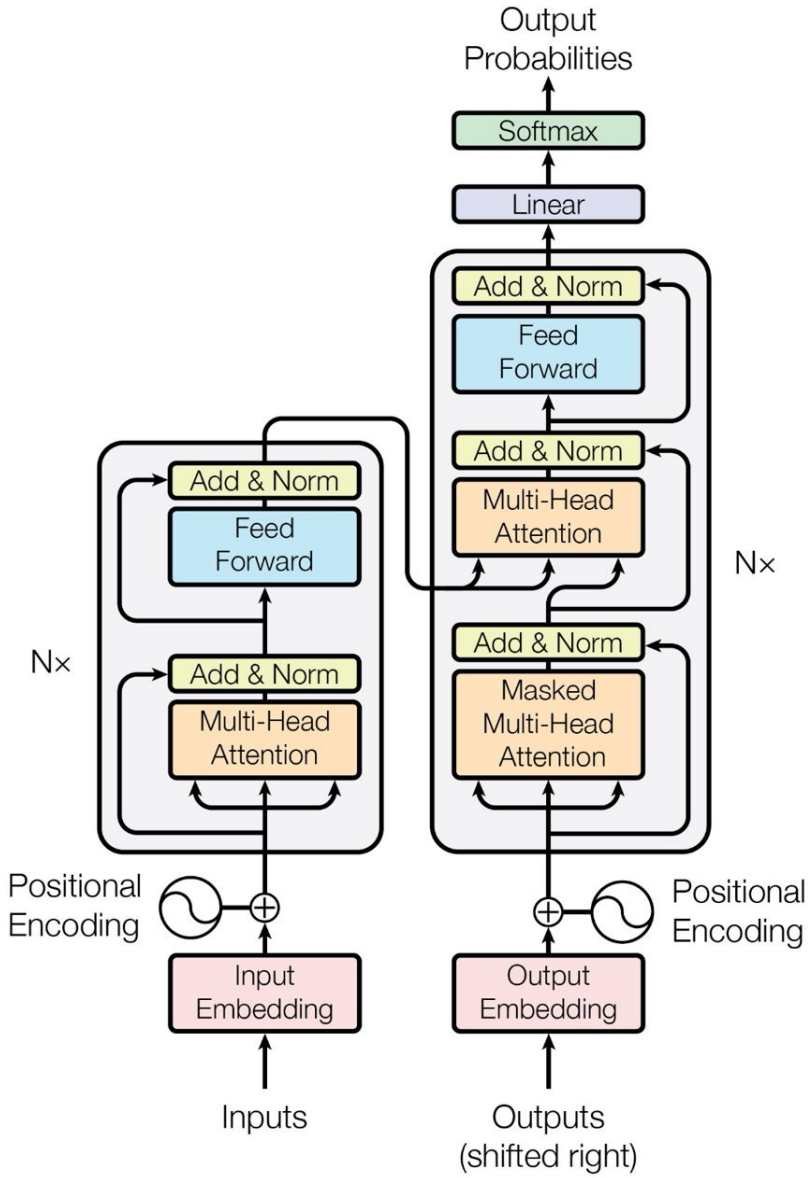
Vision Transformer: Transformer for the CV domain

- Self-attention-based architectures, in particular Transformers, have become the model of choice in natural language processing (NLP).
- The learning paradigm with Foundation Models emerges: Researchers now get the **pretraining** on a large text corpus and then **fine-tune/inference** on a smaller task-specific dataset.
- Transformers' **computational efficiency and scalability** make it a suitable choice for such pretraining. We can now train NLP models of unprecedented size, with over 100B parameters (e.g., GPT-4, LLaMA).

- But how about the computer vision (CV) community? Can we apply the same success story in the CV domain?

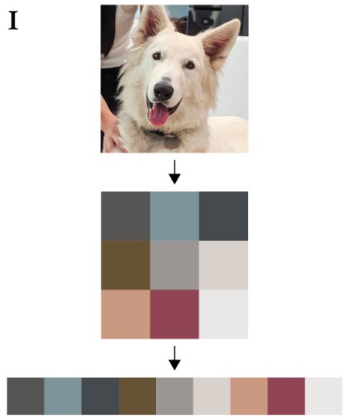
Transformers for CV: Transformer Overview

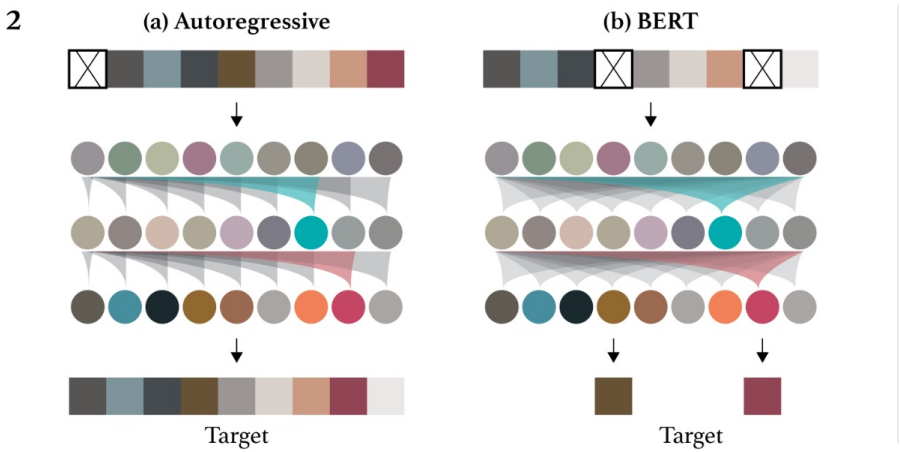
- Input Tokens
  - How do we tokenize an image in a manner similar to text tokenization?
    - In language tasks, we use words or subwords as tokens. What could be the equivalent for images?
- Input Embedding
  - How do we embed these image tokens?
    - In NLP, embedding tables work for discrete word tokens, but pixels are continuous. How can we effectively embed image pixels?
- Positional encoding
- Multi-head attention
- Add & Norm
- Feed Forward Net

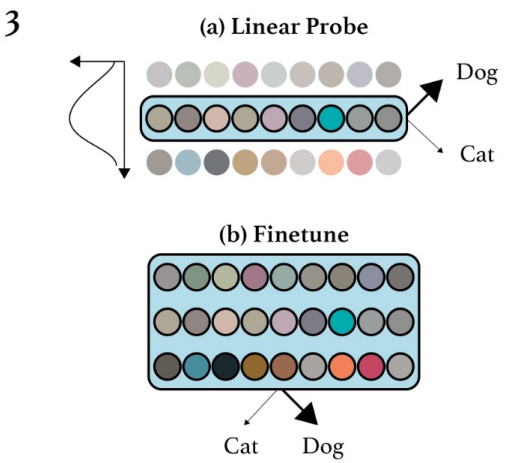


ImageGPT: Generative Pretraining from Pixels [2020]

- Treat color value from each pixel as a discrete token!
  - Typically represented as a 24-bit value ([0-255] per color channel) (vocab size of ~16.7M).
  - Reduction: We may not need to store that many colors?
    - A 9-bit representation ([0-8] per color channel) reduces vocabulary size to 512.
- However, Transformers have quadratic complexity $O(n^2)$ w.r.t. token length.
  - For a 256x256 image, we would have 65,536 tokens (BERT max length was 512).
- Solution: just use lower resolution images (maximum size of 64x64).
- Trained on a similar objective to BERT (predict the next/masked pixels).



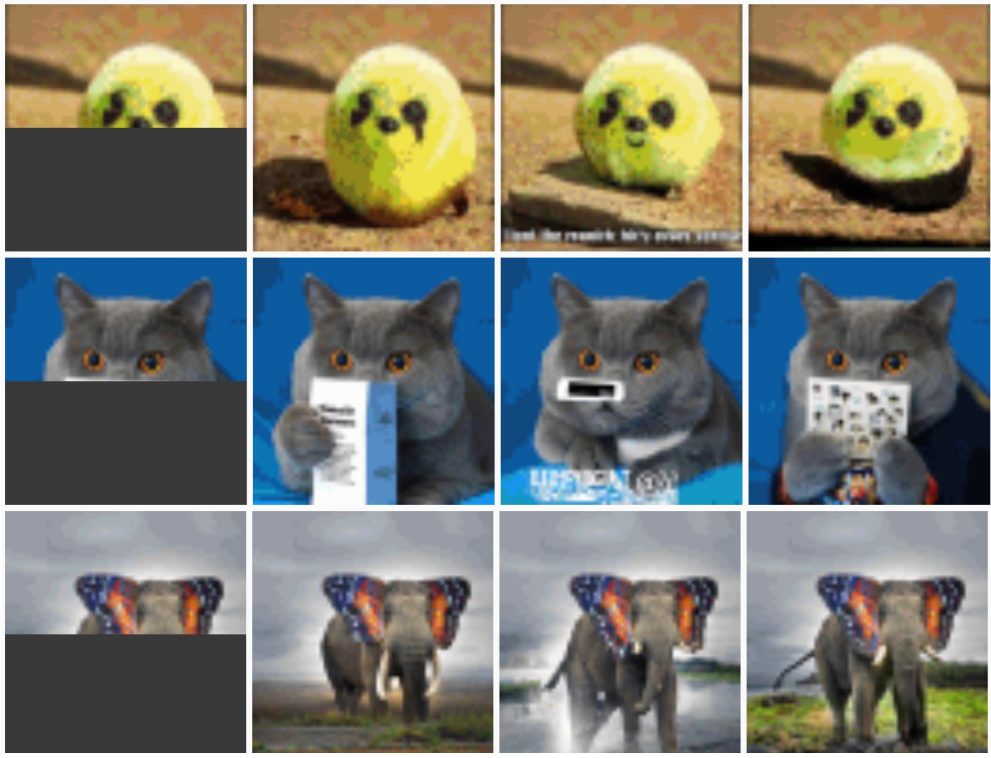




ImageGPT: Generative Pretraining from Pixels [2020]

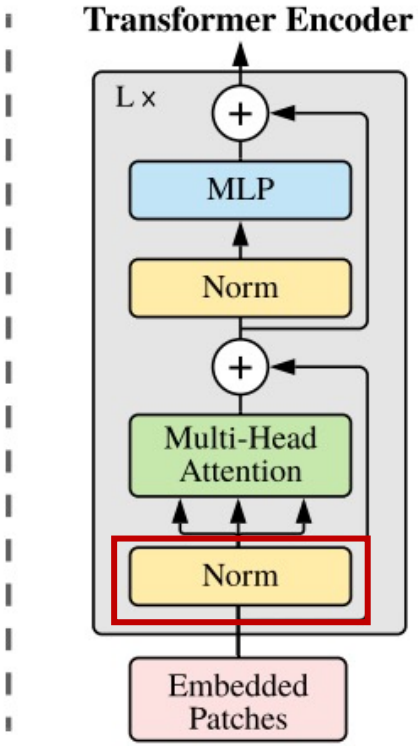
- **Model Variants:**
  - iGPT-S, iGPT-M, iGPT-L:
    - Parameters: 76M, 455M, 1.4B respectively.
    - Trained on ImageNet.
  - iGPT-XL:
    - 6.8B parameters, trained on ImageNet + additional web images.
- **Key Outcomes:**
  - Good image representations.
    - Was SOTA on semi-supervised classification.
  - Good image generations.
    - Shown to be effective at modeling visual information.
- **Training complexity:**
  - **iGPT-L** was trained for roughly **2500** V100-days.
  - ResNet equivalent model trained in **70** V100-days.
  - And this is just for **64x64** resolution images!

		Pre-trained on ImageNet		
Evaluation	**Model**	**Accuracy**	**w/o labels**	**w/ labels**
CIFAR-10	ResNet-152[50]	94.0		✔
Linear Probe				
	SimCLR[12]	95.3	✔	
	iGPT-L 32×32	**96.3**	✔	✔
CIFAR-100	ResNet-152	78.0		✔



Vision Transformer: An Image is Worth 16x16 Words

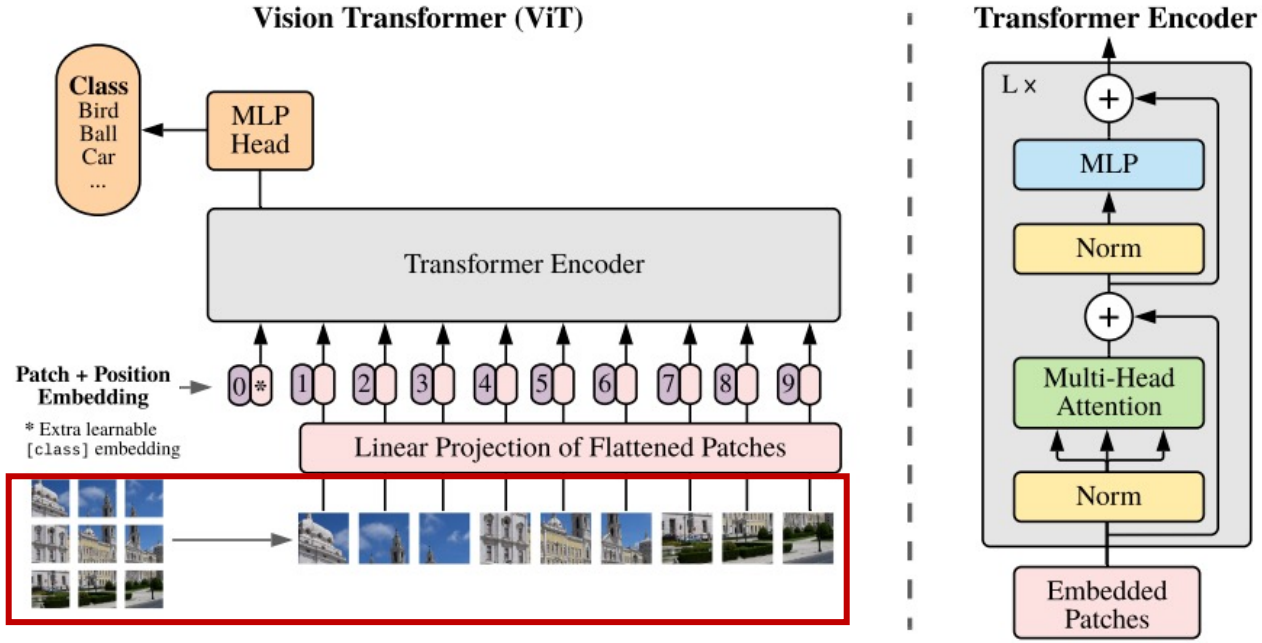
- Rather than quantizing pixels, **Vision Transformer** splits an image into **patches (16x16 pixels)**, which are flattened and linearly embedded to form tokens.
- Adds learnable positional embeddings to retain spatial information of patches.
- Adds a [CLS] token as an additional input to the transformer encoder.
  - After processing, the representation of the [class] token is used for image classification.



Vision Transformer (ViT)

- To handle 2D images, ViT reshapes the image of shape (H, W, C) into a sequence of flattened 2D patches of shape (P^2, C).
- (H, W) is the resolution of the original image, C is the number of channels, P is the width of each image patch.
- We then get $N = \frac{H*W}{P^2}$: the number of patches (as the input sequence length).

image to
patch sequence

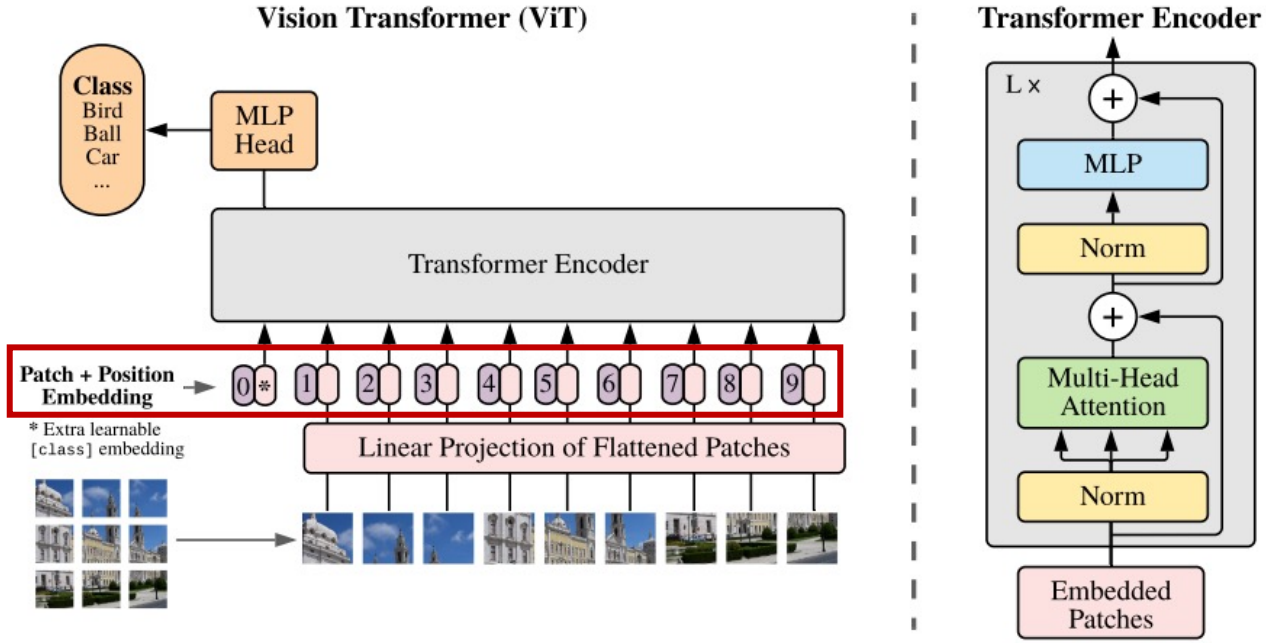


Vision Transformer (ViT)

- **Position embeddings (E_{pos})** are added to the **patch embeddings (x_pE)** to retain positional information. ViT uses learnable 1D position embeddings.

- Similar to BERT's [class] token, ViT prepends a learnable embedding for image class to the beginning of the embedding sequence, whose state serves as the task representation for image classification.

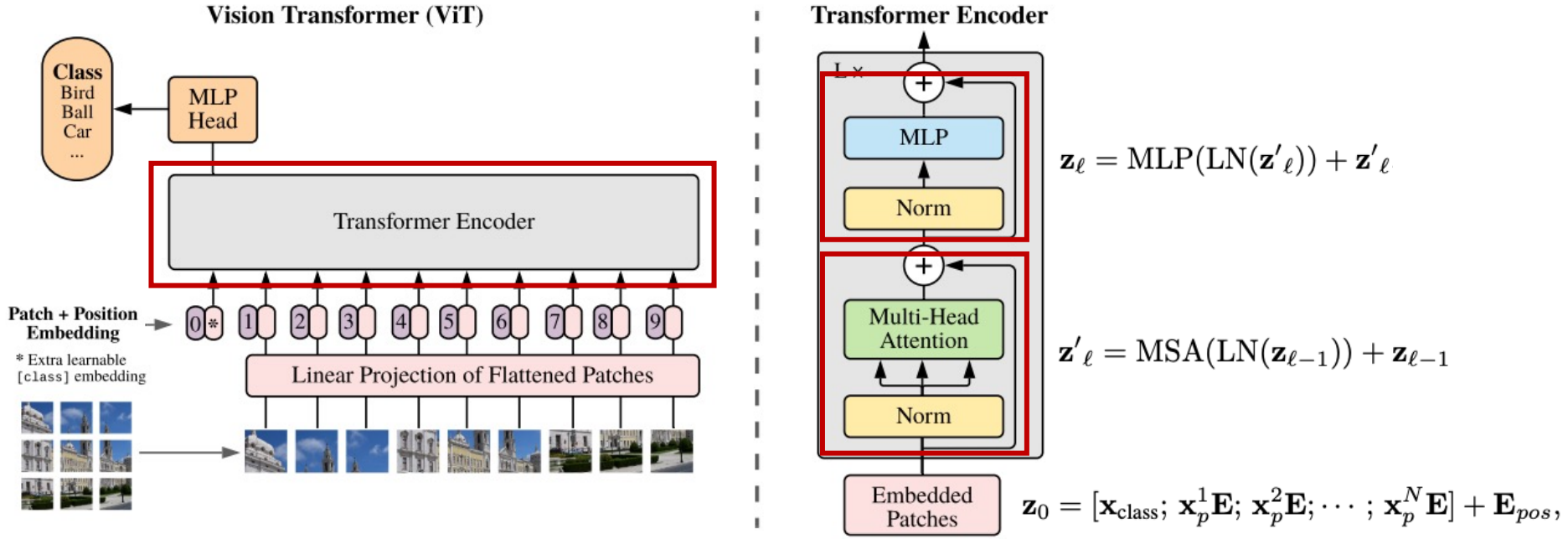
positional and image class embedding



Vision Transformer (ViT)

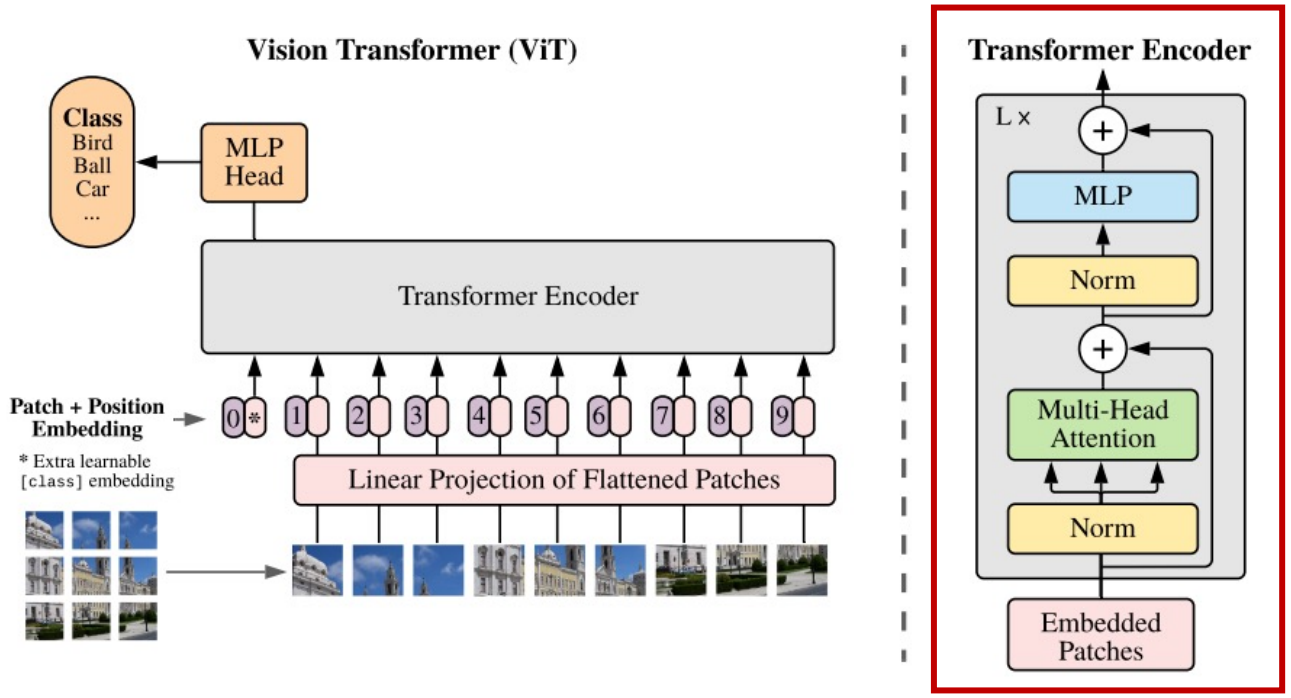
- Transformer encoder consists of alternating layers of Multi-Head Self-Attention (MSA) and MLP blocks.
- LayerNorm (LN) is applied before every block, and residual connections after every block.
  - Pre-Norm configurations tend to help improve the gradient flow during training.
  - This approach has been adopted in updated official implementation of the Transformer.

Recall the Transformer architecture from last week.



Vision Transformer (ViT)

- ViT has less built-in image-specific assumptions compared to CNNs.
  - CNNs use local receptive fields and shared weights, making them better suited for capturing spatial patterns.
  - ViT operates on image patches without assuming local structure or spatial hierarchies.
    - It relies on self-attention to capture relationships, making it more flexible but less biased towards spatial locality. Each patch can attend to every other patch, providing a global context from the start.
    - However, ViT requires more data or pre-training to learn spatial relationships effectively due to the lack of locality bias.



Less image-specific inductive bias

ViT Scalability and Attention Visualization

- A popular choice of the transformer encoder is Bidirectional Encoder Representations from Transformers (BERT). The “Base” and “Large” models are adopted from BERT.

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

Scaled-up compared to previous CNNs

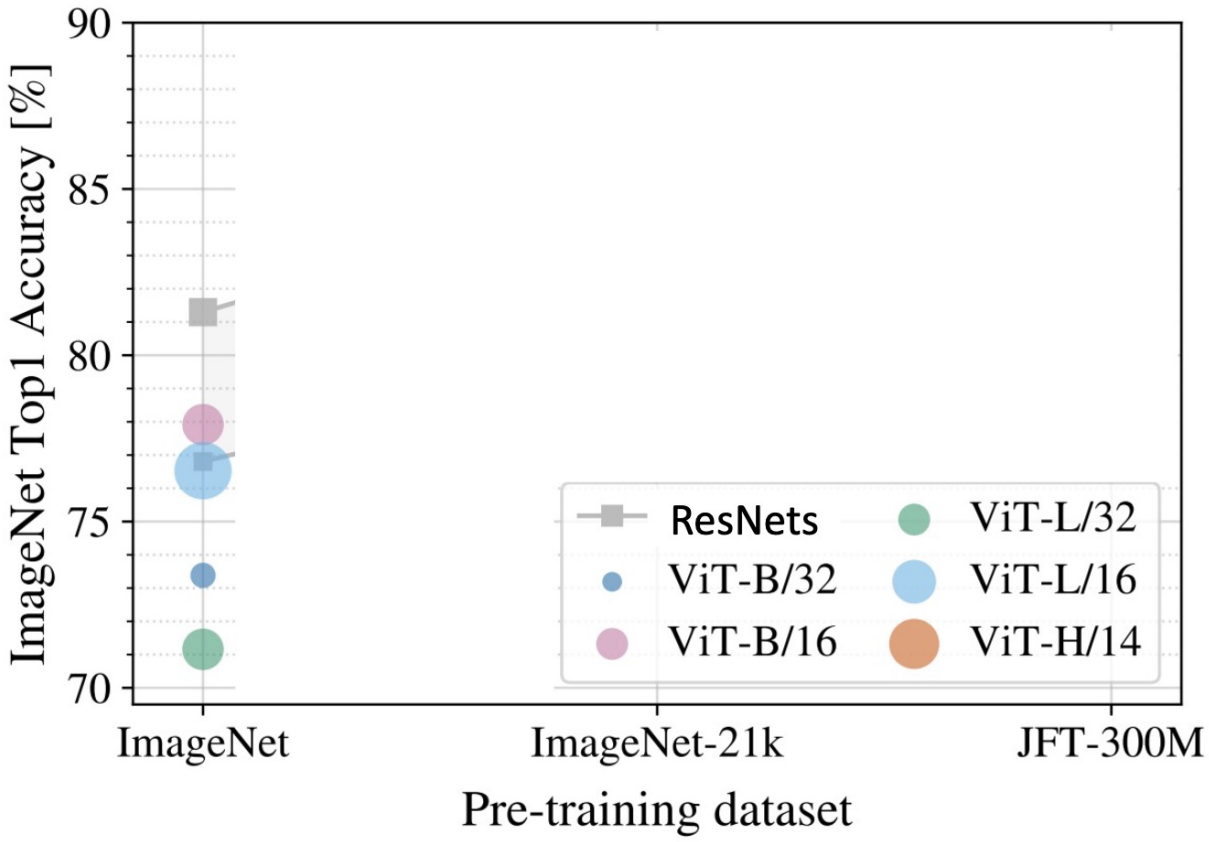
- Visualization of attention values from the output token to the input space.

ViT Results on ImageNet

- A popular choice of the transformer encoder is Bidirectional Encoder Representations from Transformers (BERT). The “Base” and “Large” models are adopted from BERT.

ImageNet dataset has 1k categories, 1.2M Images.

When trained on ImageNet, ViT models perform worse than ResNets



B = Base
L = Large
H = Huge

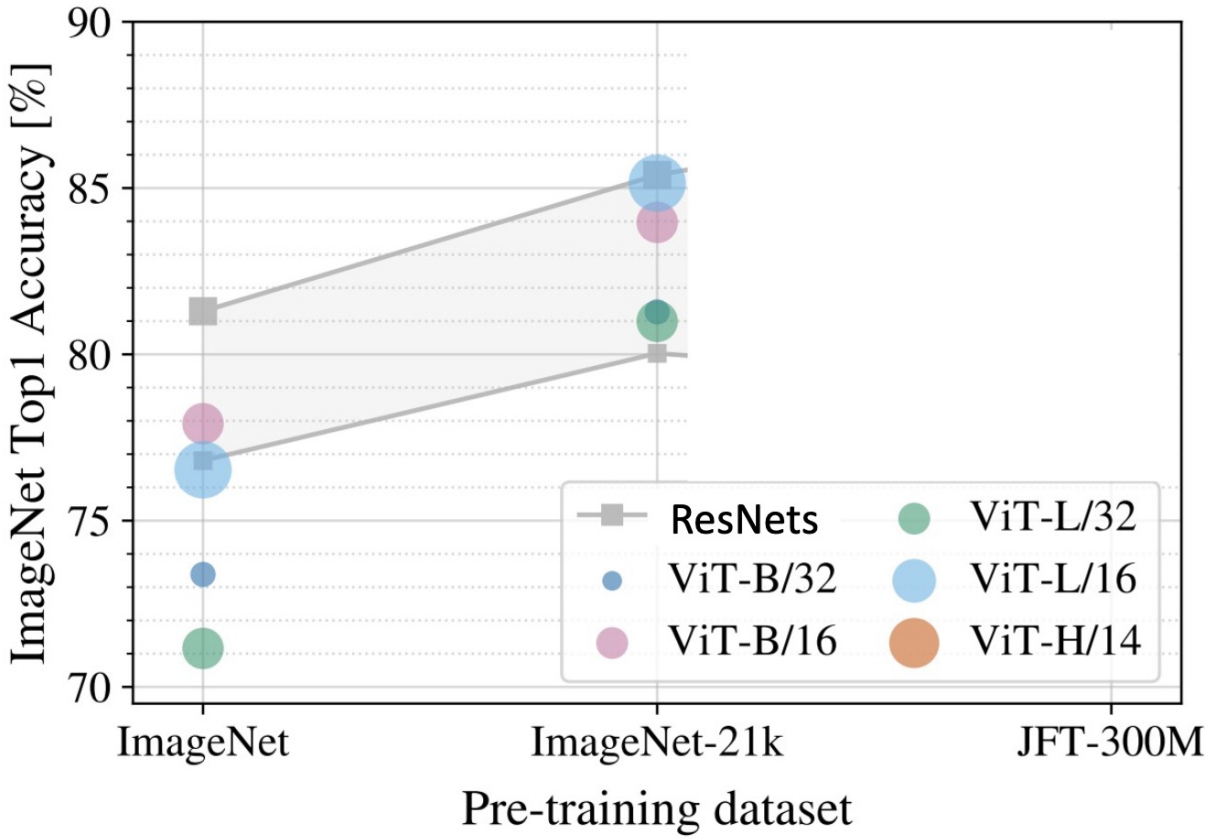
/32, /16, /14 is patch size; smaller patch size is a bigger model (more patches)

ViT Results on ImageNet

- A popular choice of the transformer encoder is Bidirectional Encoder Representations from Transformers (BERT). The “Base” and “Large” models are adopted from BERT.

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

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



B = Base
L = Large
H = Huge

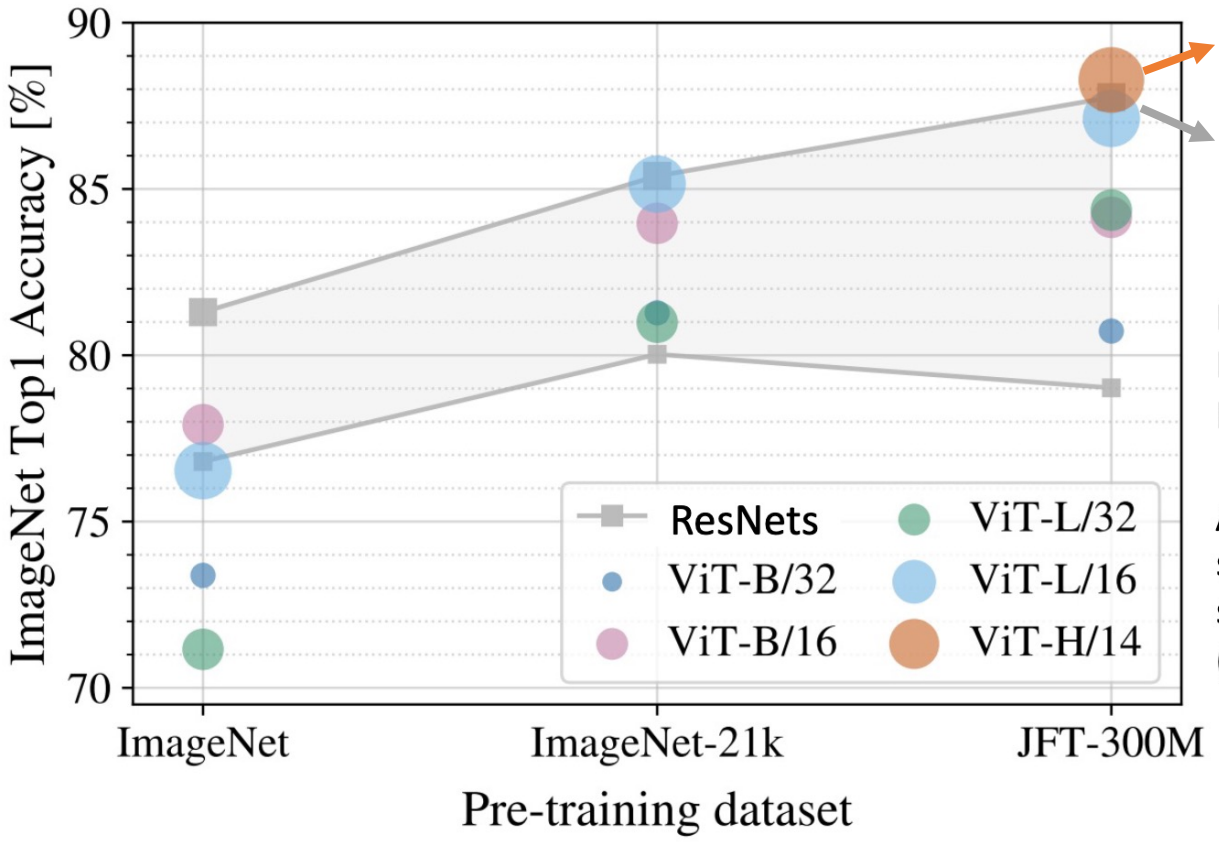
/32, /16, /14 is patch size; smaller patch size is a bigger model (more patches)

ViT Results on ImageNet

- A popular choice of the transformer encoder is Bidirectional Encoder Representations from Transformers (BERT). The “Base” and “Large” models are adopted from BERT.

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 (~$33K)

ResNet: 9.9k TPU-v3 core days of training (~$133k)

B = Base
L = Large
H = Huge

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