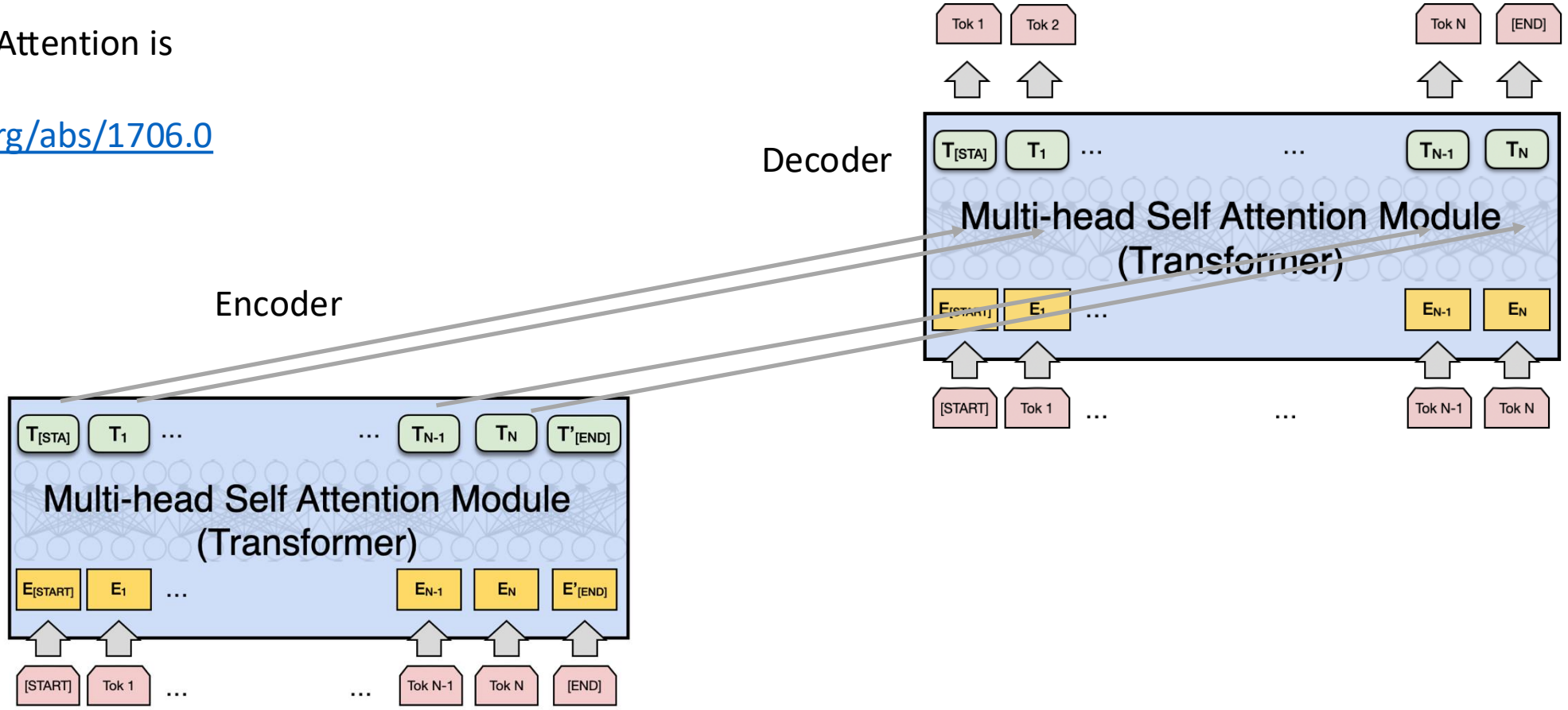
Deep Learning for Vision & Language

Transformers for NLP and Beyond

Attention is All you Need (no RNNs)

Vaswani et al. Attention is all you need https://arxiv.org/abs/1706.03762

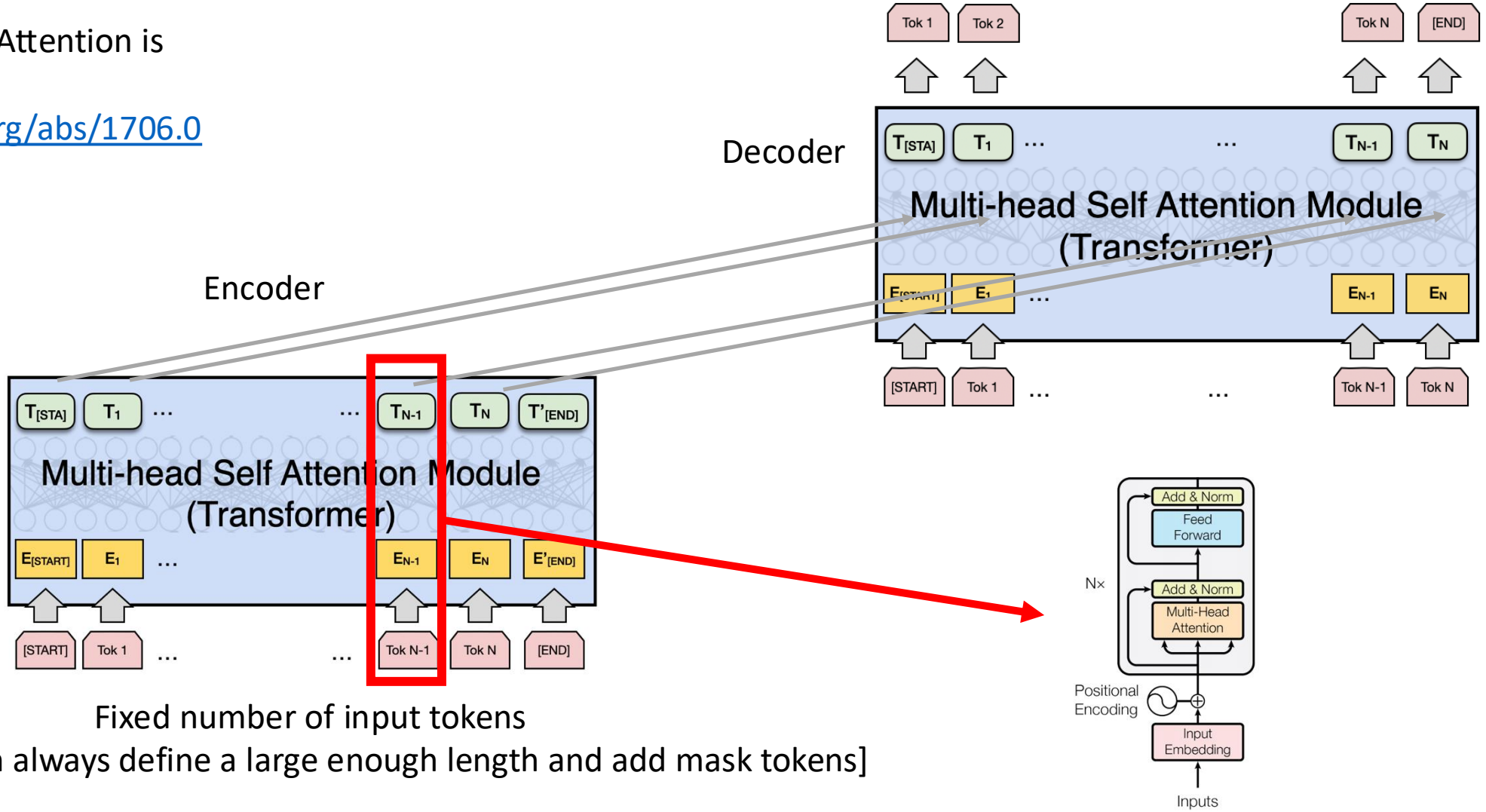


Fixed number of input tokens

[but hey! we can always define a large enough length and add mask tokens]

Attention is All you Need (no RNNs)

Vaswani et al. Attention is all you need https://arxiv.org/abs/1706.03762

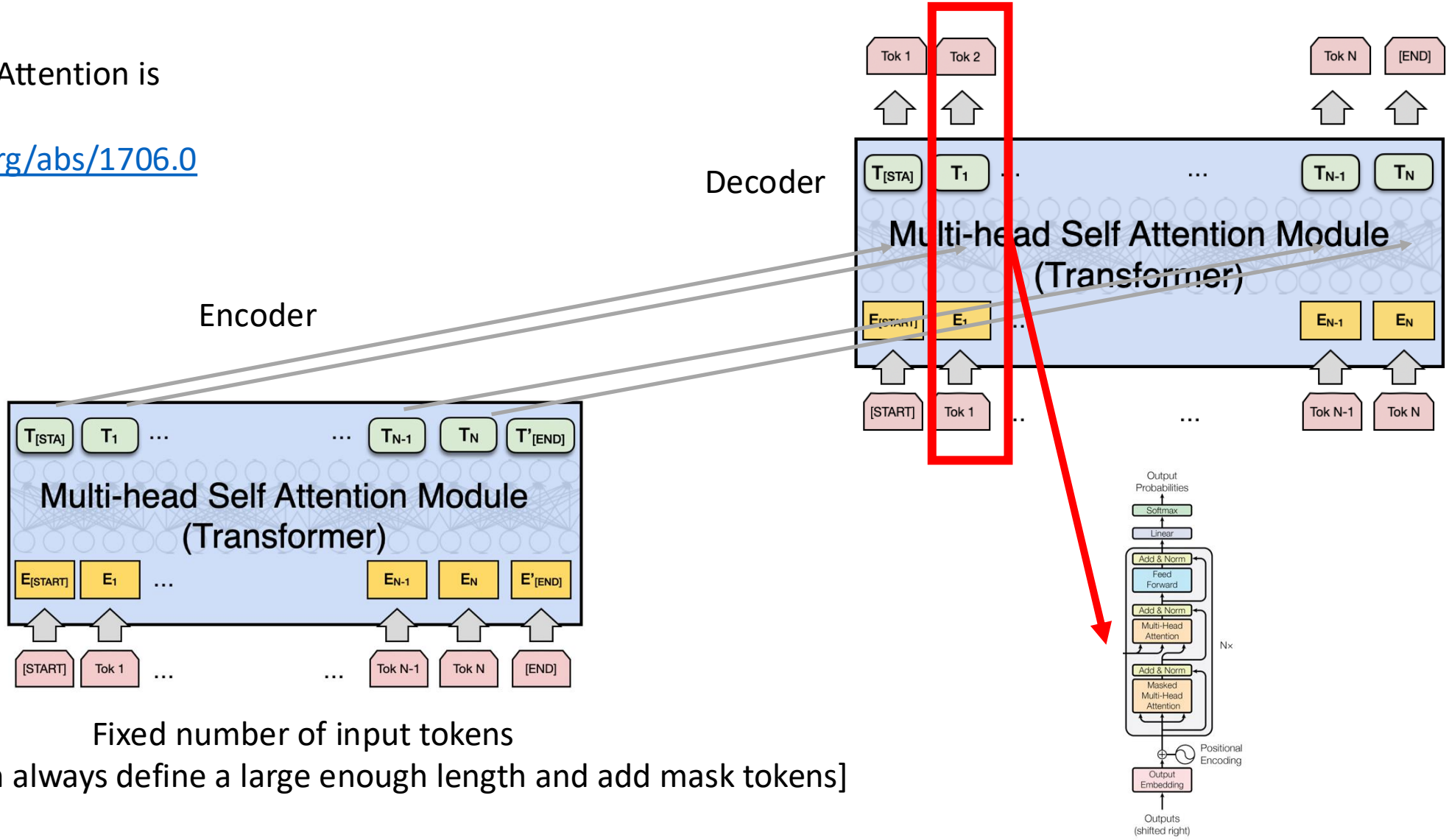


Fixed number of input tokens

[but hey! we can always define a large enough length and add mask tokens]

Attention is All you Need (no RNNs)

Vaswani et al. Attention is all you need https://arxiv.org/abs/1706.03762

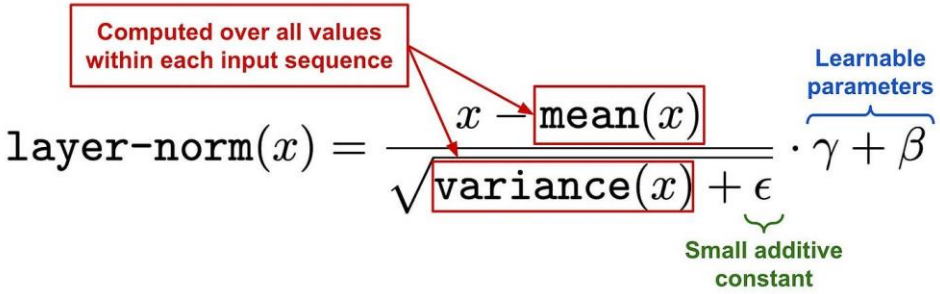


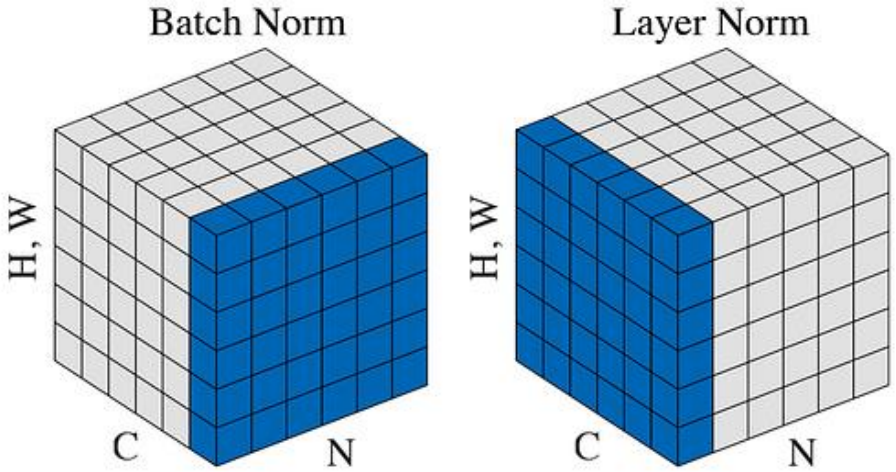
Fixed number of input tokens

[but hey! we can always define a large enough length and add mask tokens]

We can also draw this as in the paper:

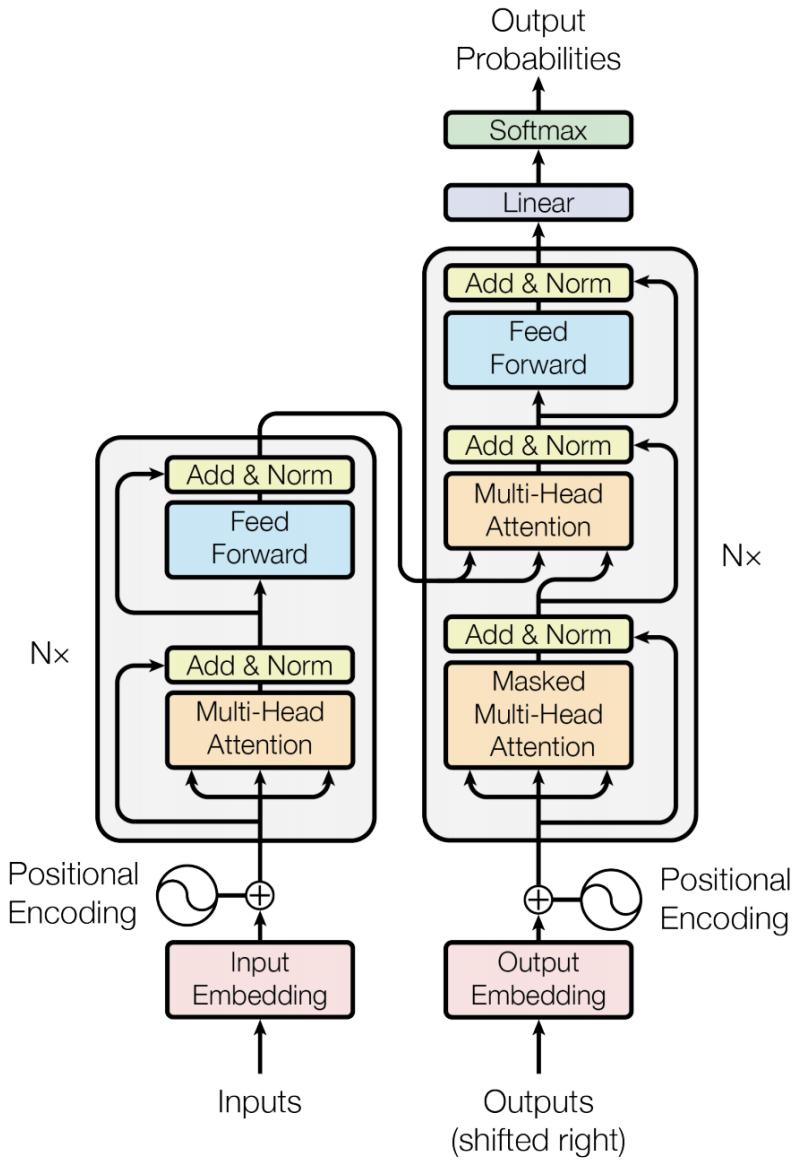
Vaswani et al. Attention is all you need https://arxiv.org/abs/1706.03762





across different samples

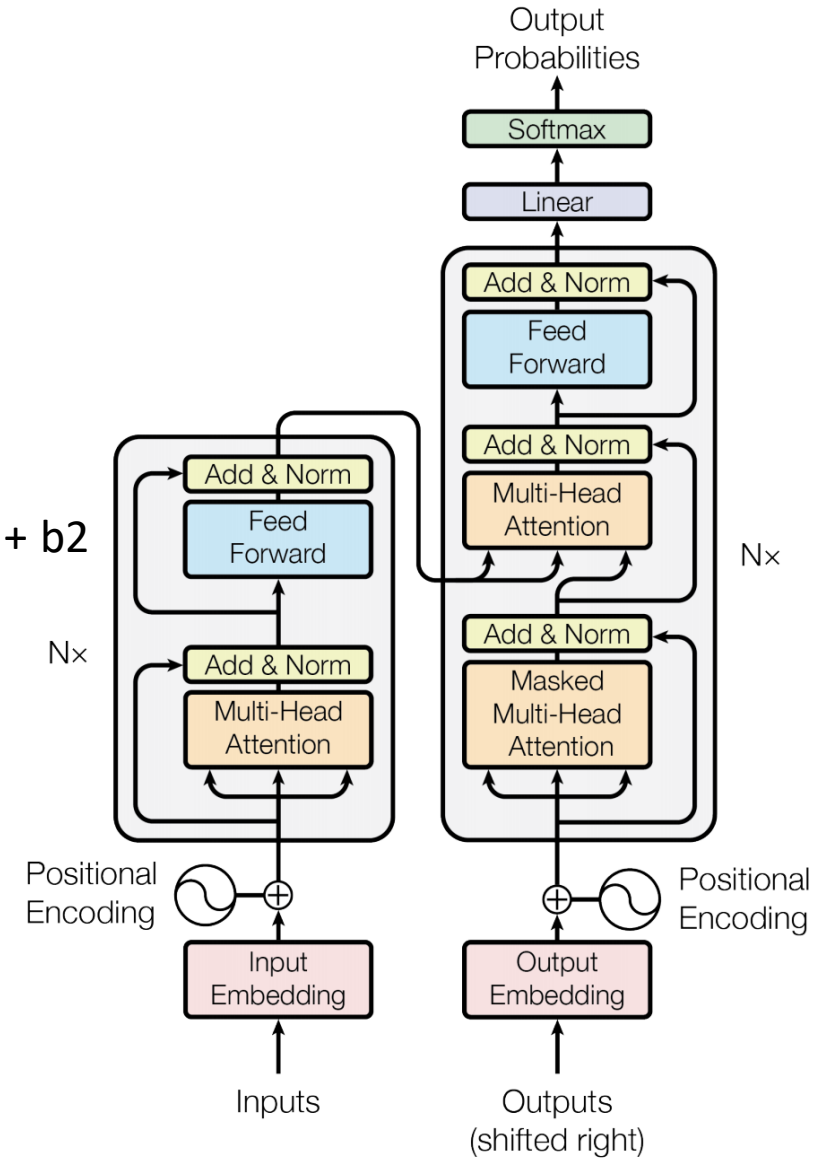
across different hidden dimensions



We can also draw this as in the paper:

Vaswani et al. Attention is all you need https://arxiv.org/abs/1706.03762

FFN(x) = W2 * ReLU(W1 * x + b1) + b2



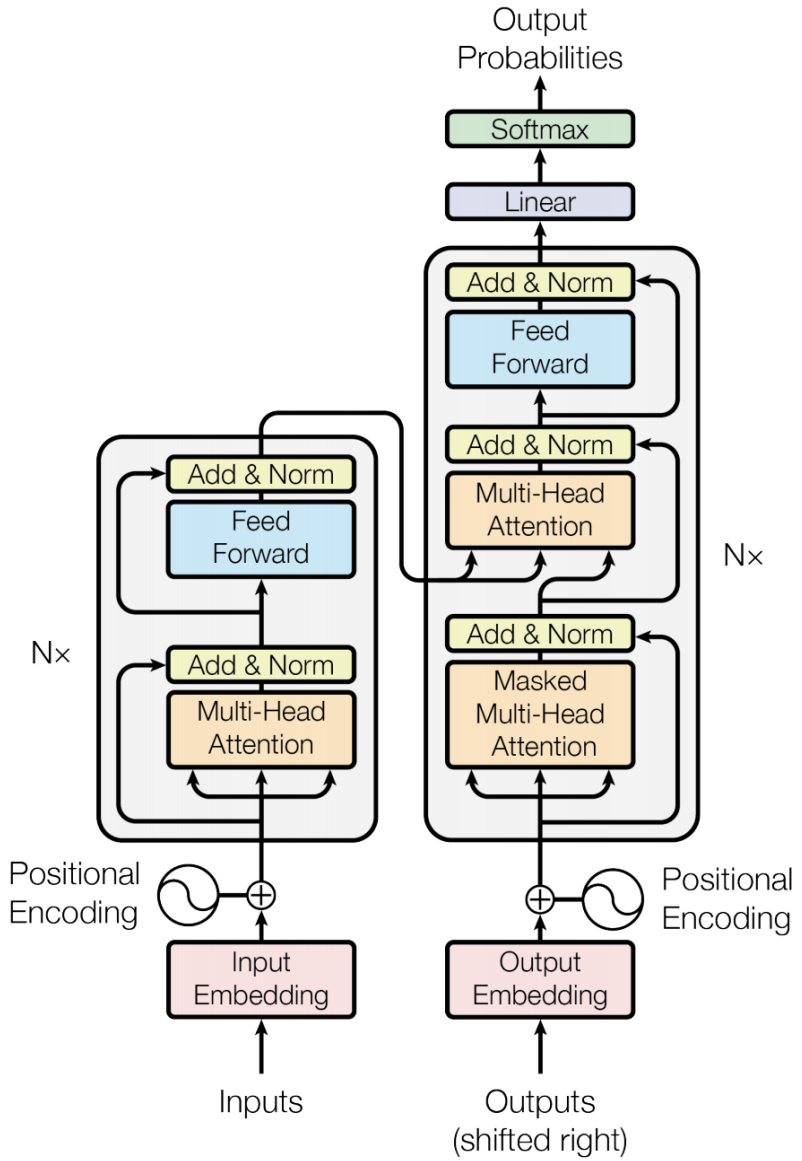
We can lose track of position since we are aggregating across all locations

Vaswani et al. Attention is all you need https://arxiv.org/abs/1706.03762

"The cat sat on the mat."

“mat sat on the cat."

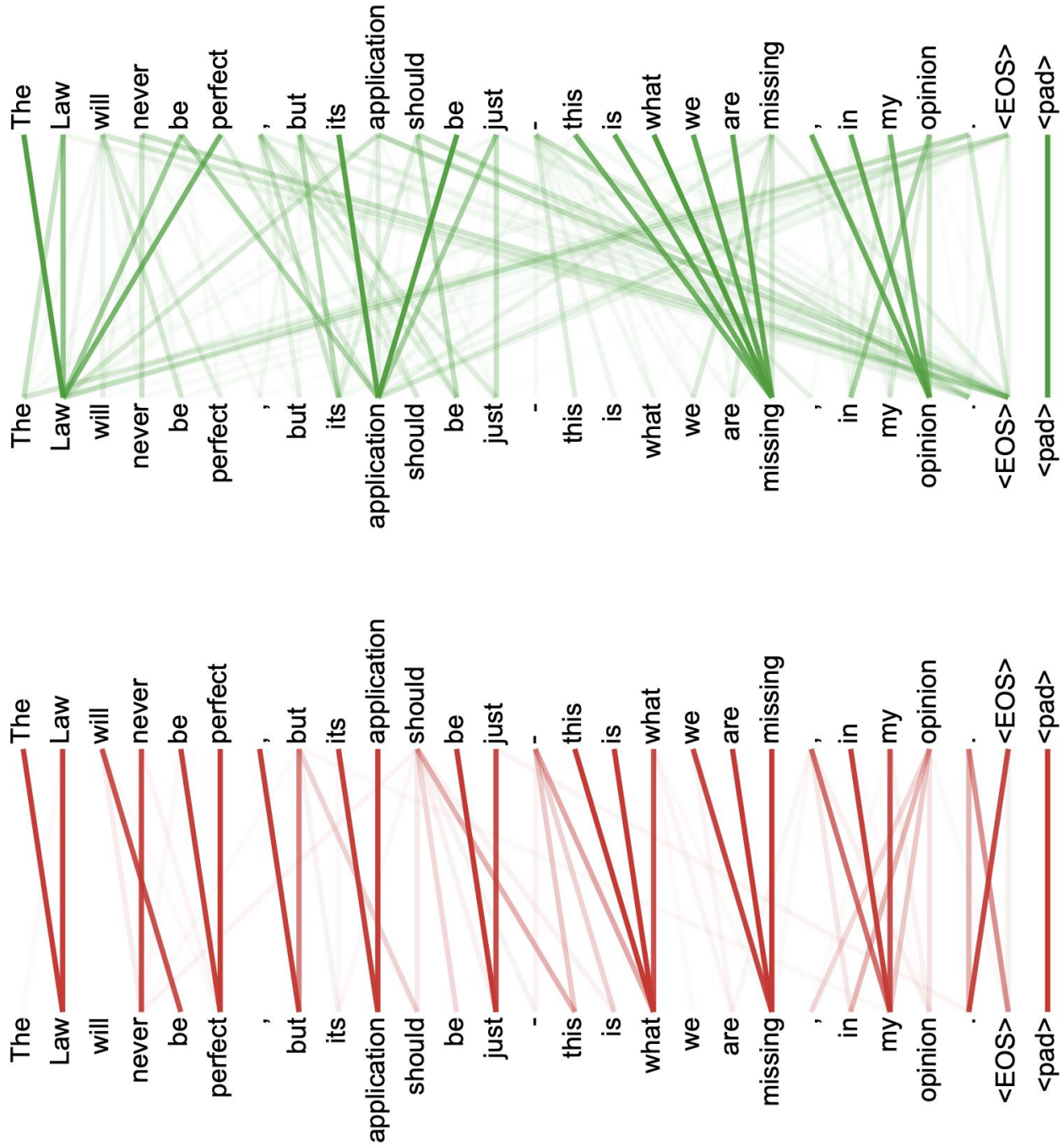
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$



Other Positional Encodings

- **Learnable Absolute Positional Encodings:** Initialize them with random weights and learn them. However fixed for each position. BERT Models use this.
- **Relative Positional Encodings:** Encode distances between tokens instead of absolute positions. Used in moderns such as T5 from Google.
- **RoPE: Rotationary Positional Encodings:** Position information is encoded through rotation in multidimensional space. Models such as LlaMA use this.
- NoPE? No Positional Encodings? Let's hope the model learns through attention weights?

Multi-headed attention weights are harder to interpret obviously

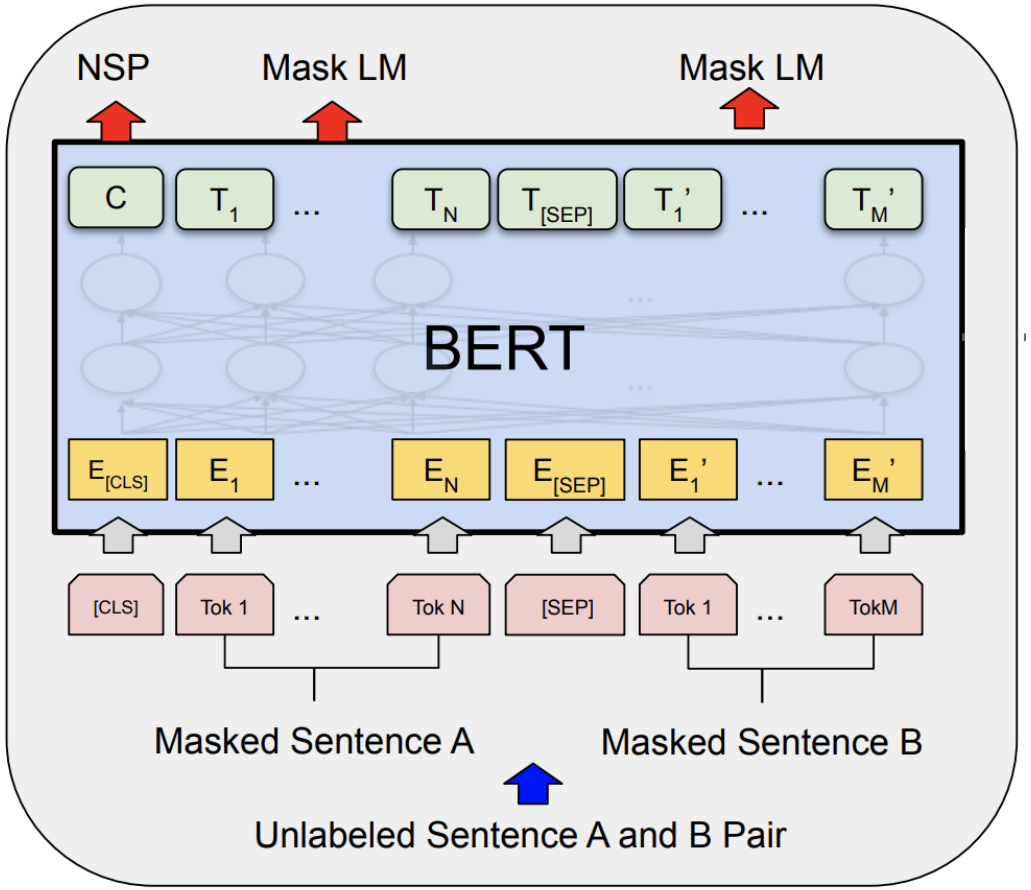


The BERT Encoder Model (October, 2018)

Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding . https://arxiv.org/abs/1810.04805

Important things to know

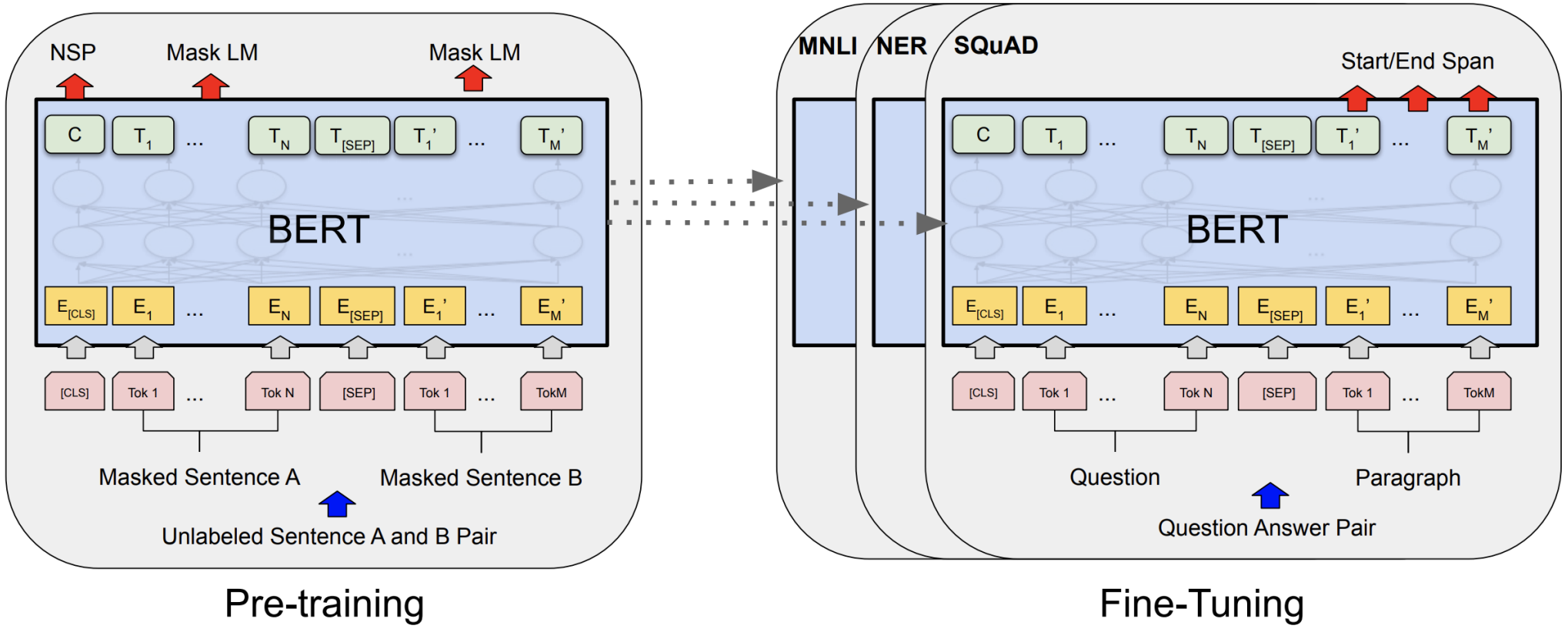
- No decoder
- Train the model to fill-in-the-blank by masking some of the input tokens and trying to recover the full sentence.
- The input is not one sentence but two sentences separated by a [SEP] token.
- Also try to predict whether these two input sentences are consecutive or not.



Pre-training

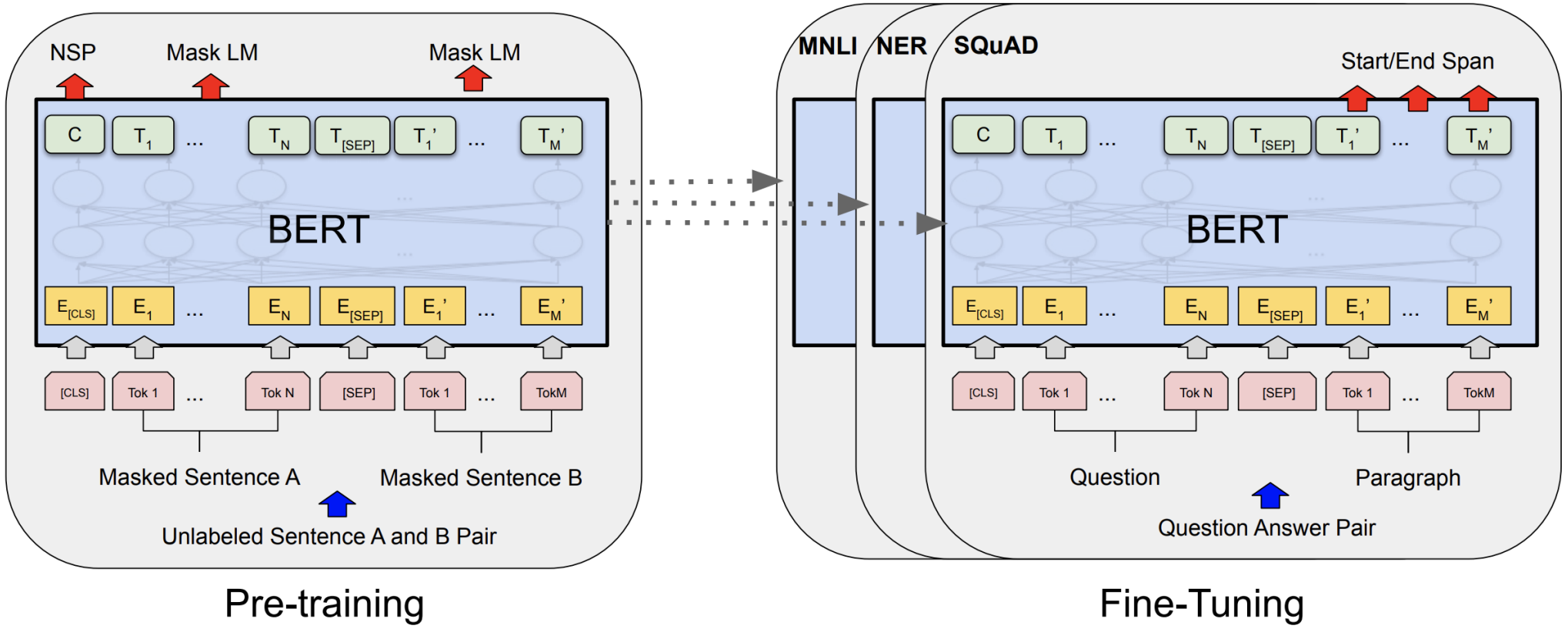
The BERT Encoder Model

Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding . https://arxiv.org/abs/1810.04805

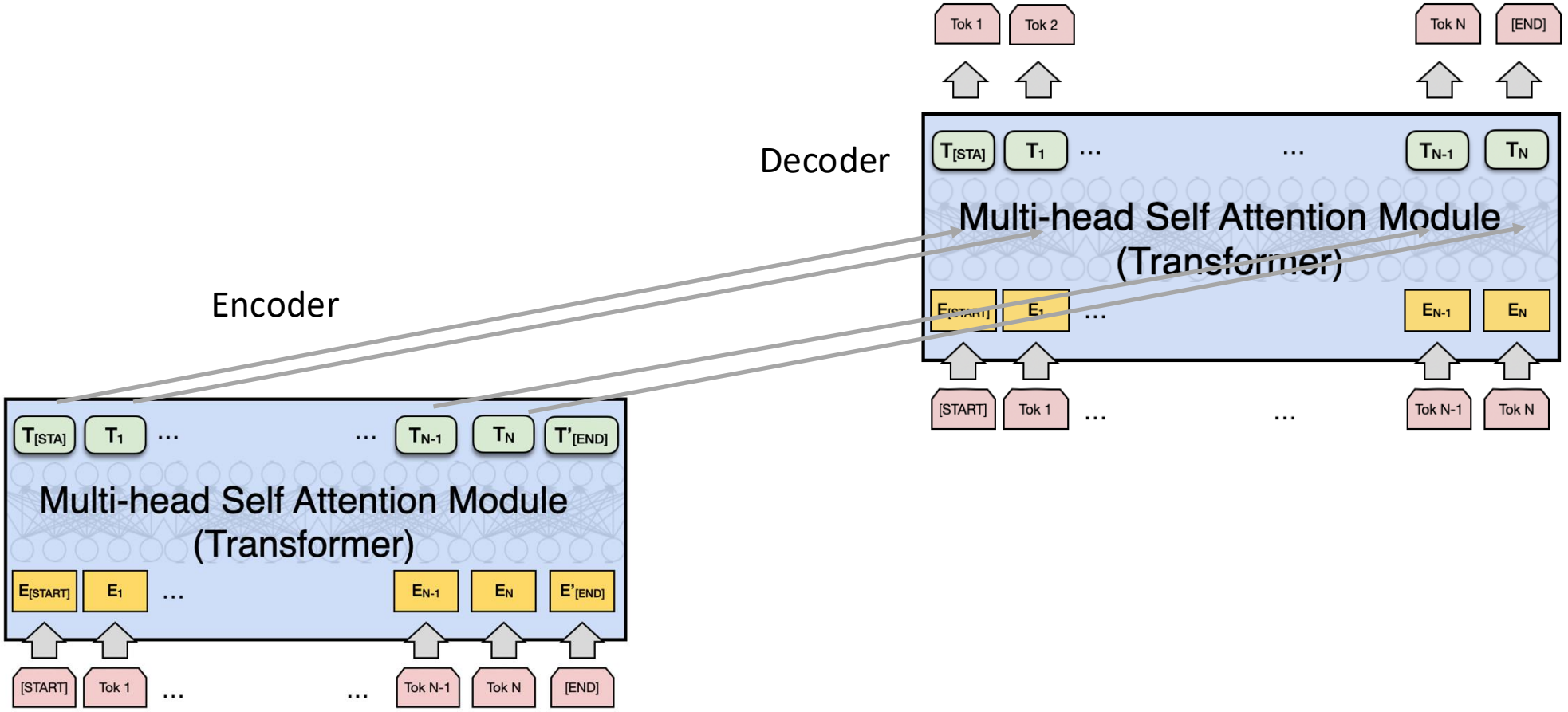


The BERT Encoder-only Model

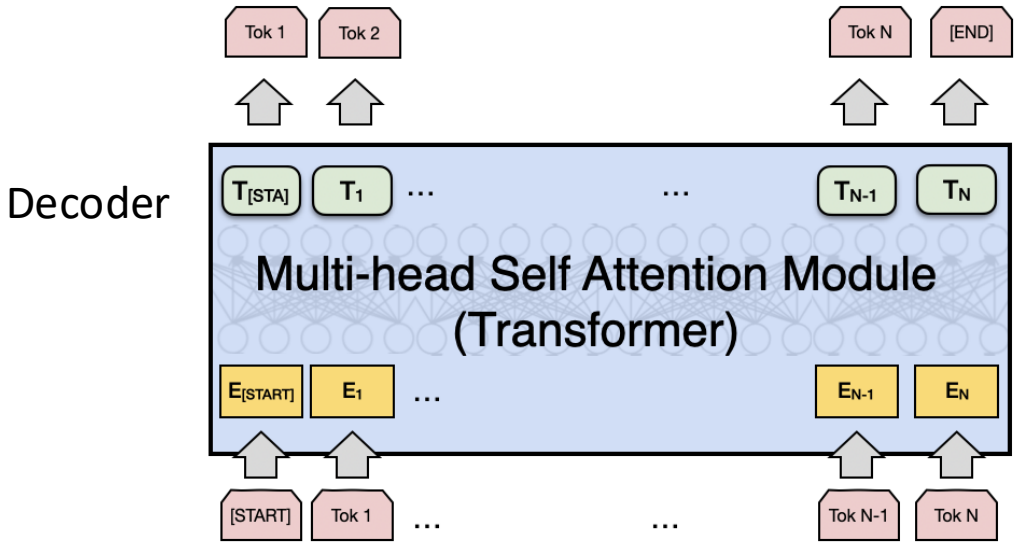
Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding . https://arxiv.org/abs/1810.04805



The T5 Encoder-Decoder Model



The GPT-2, GPT-3 Decoder-only Model



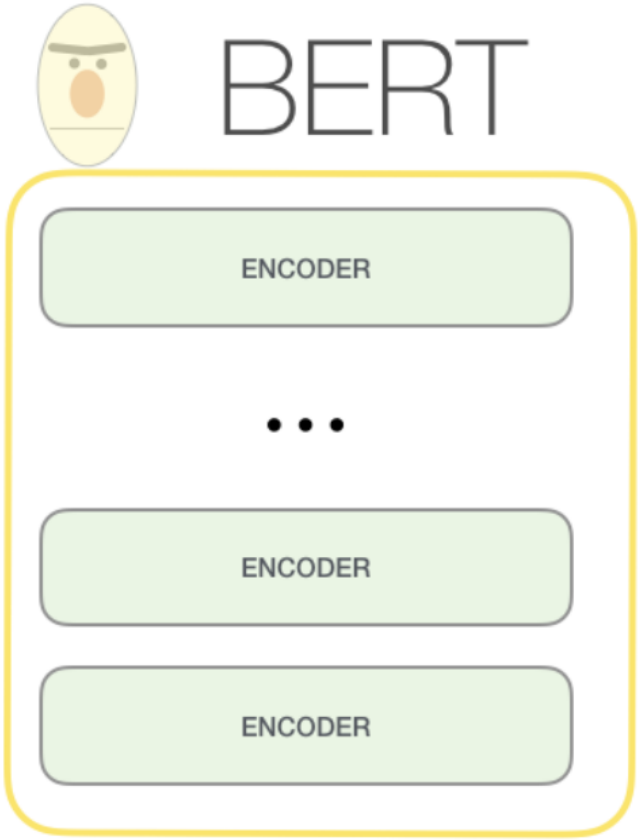
The GPT-2 Model (Feb, 2019)

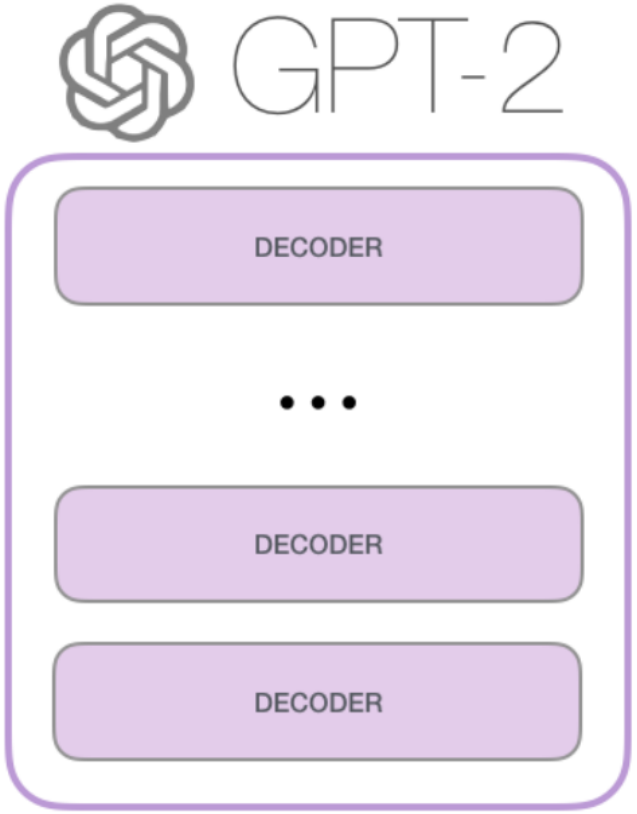
Language Models are Unsupervised Multitask Learners

Alec Radford [*1] **Jeffrey Wu** [*1] **Rewon Child** [1] **David Luan** [1] **Dario Amodei** [**1] **Ilya Sutskever** [**1]

https://openai.com/blog/better-language-models/

The GPT-2 Model

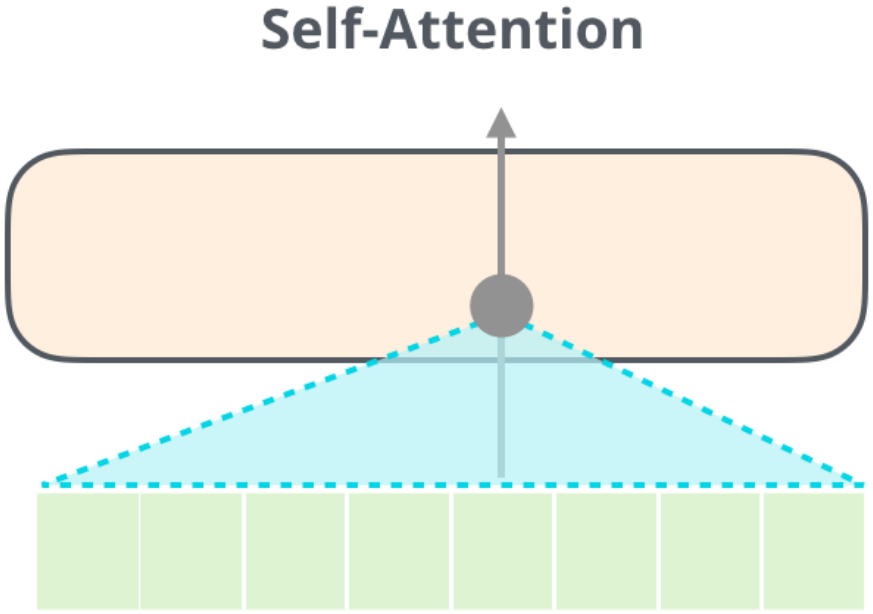




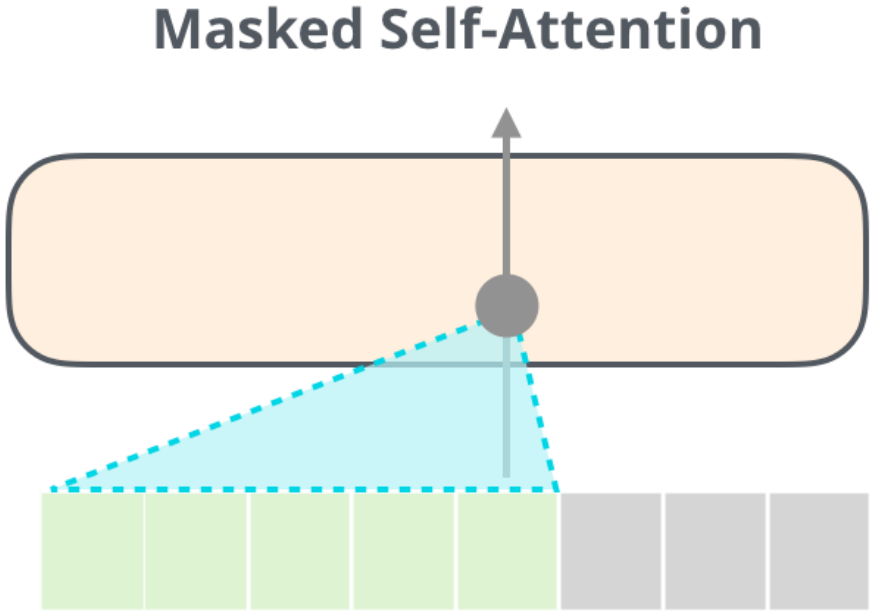
https://jalammar.github.io/illustrated-gpt2/

The GPT-2 Model

BERT

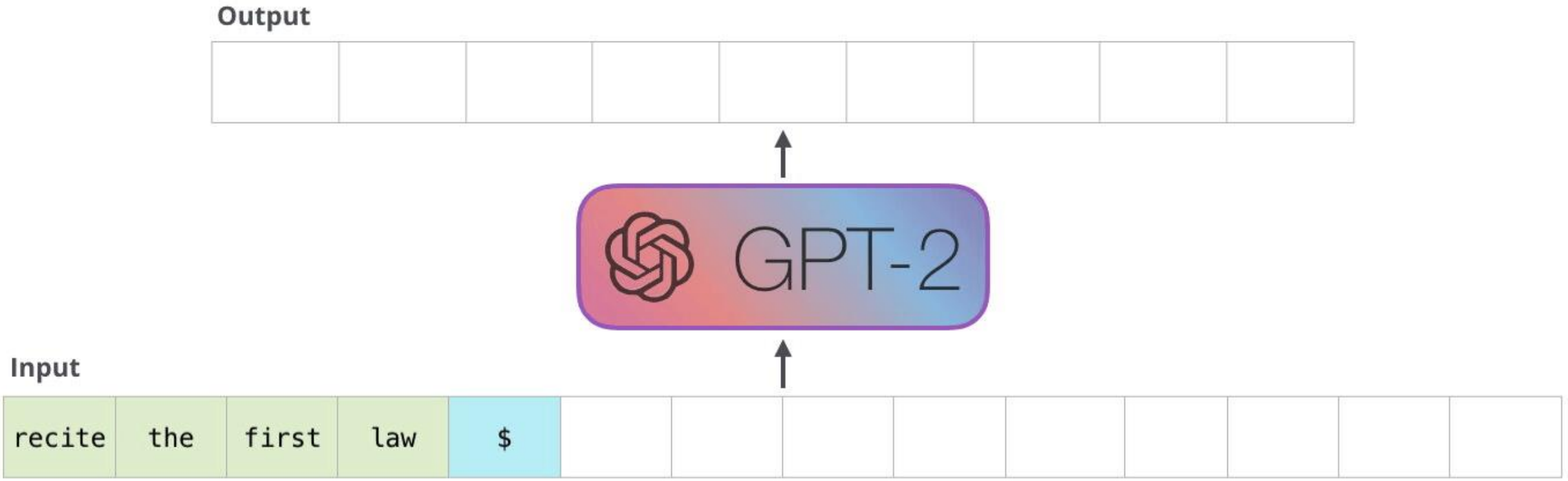


GPT



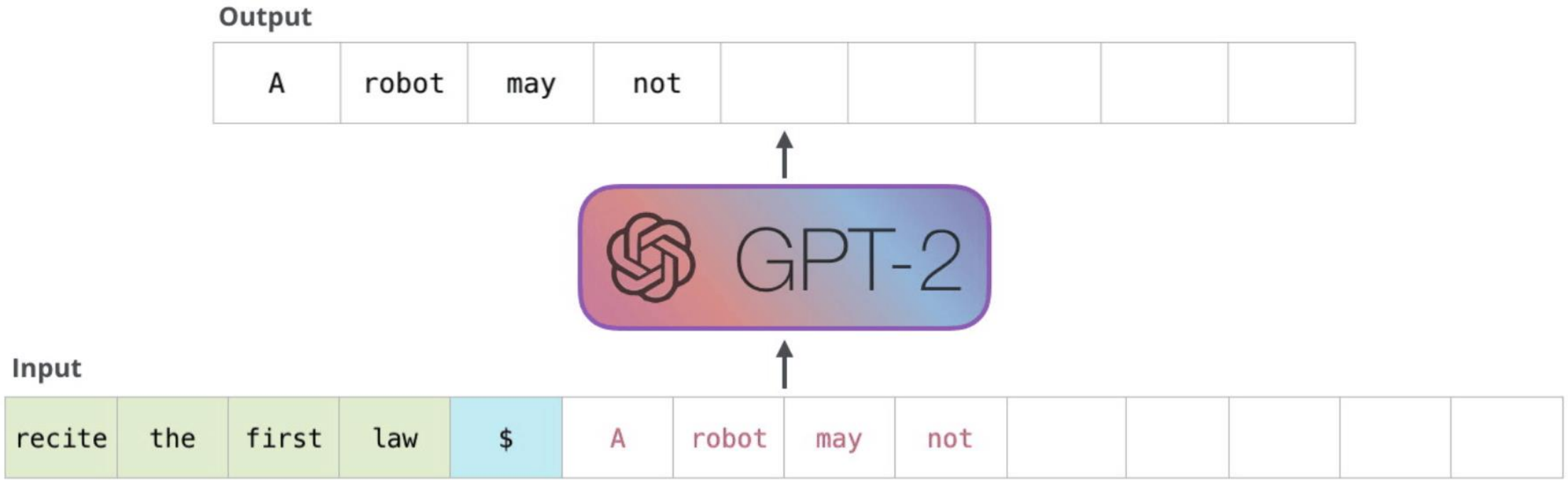
https://jalammar.github.io/illustrated-gpt2/

The GPT-2 Model



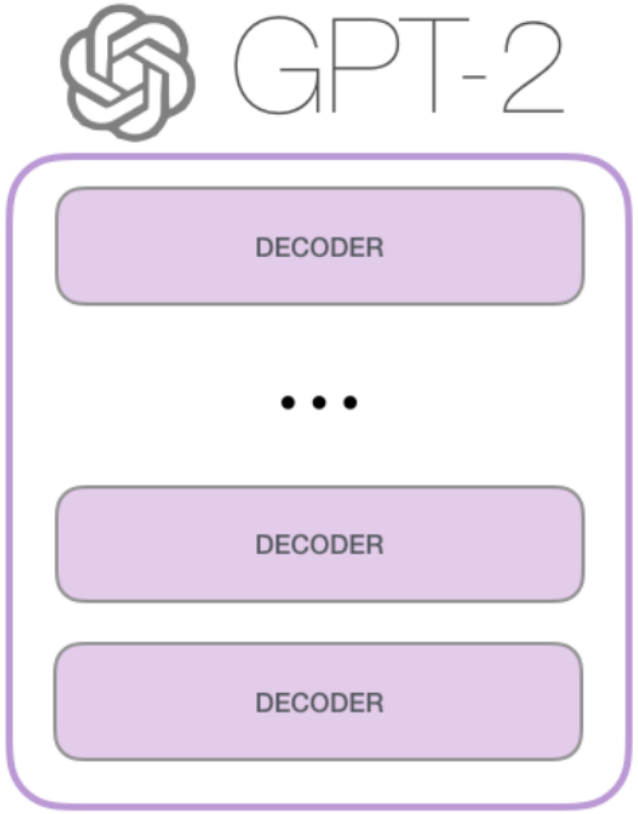
https://jalammar.github.io/illustrated-gpt2/

The GPT-2 Model



https://jalammar.github.io/illustrated-gpt2/

The GPT-2 Model



https://jalammar.github.io/illustrated-gpt2/

GPT-1 vs GPT-2 vs GPT-3

	GPT-1	GPT-2	GPT-3
Parameters	117 Million	1.5 Billion	175 Billion
Decoder Layers	12	48	96
Context Token Size	512	1024	2048
Hidden Layer	768	1600	12288
Batch Size	64	512	3.2M

https://360digitmg.com/blog/types-of-gpt-in-artificial-intelligence

GPT-3 (July, 2020)

Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
GPT-3 XL	1.3B	24	2048	24	128	1M	2.0×10^{-4}
GPT-3 2.7B	2.7B	32	2560	32	80	1M	1.6×10^{-4}
GPT-3 6.7B	6.7B	32	4096	32	128	2M	1.2×10^{-4}
GPT-3 13B	13.0B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3 175B or “GPT-3”	175.0B	96	12288	96	128	3.2M	0.6×10^{-4}

https://arxiv.org/abs/2005.14165

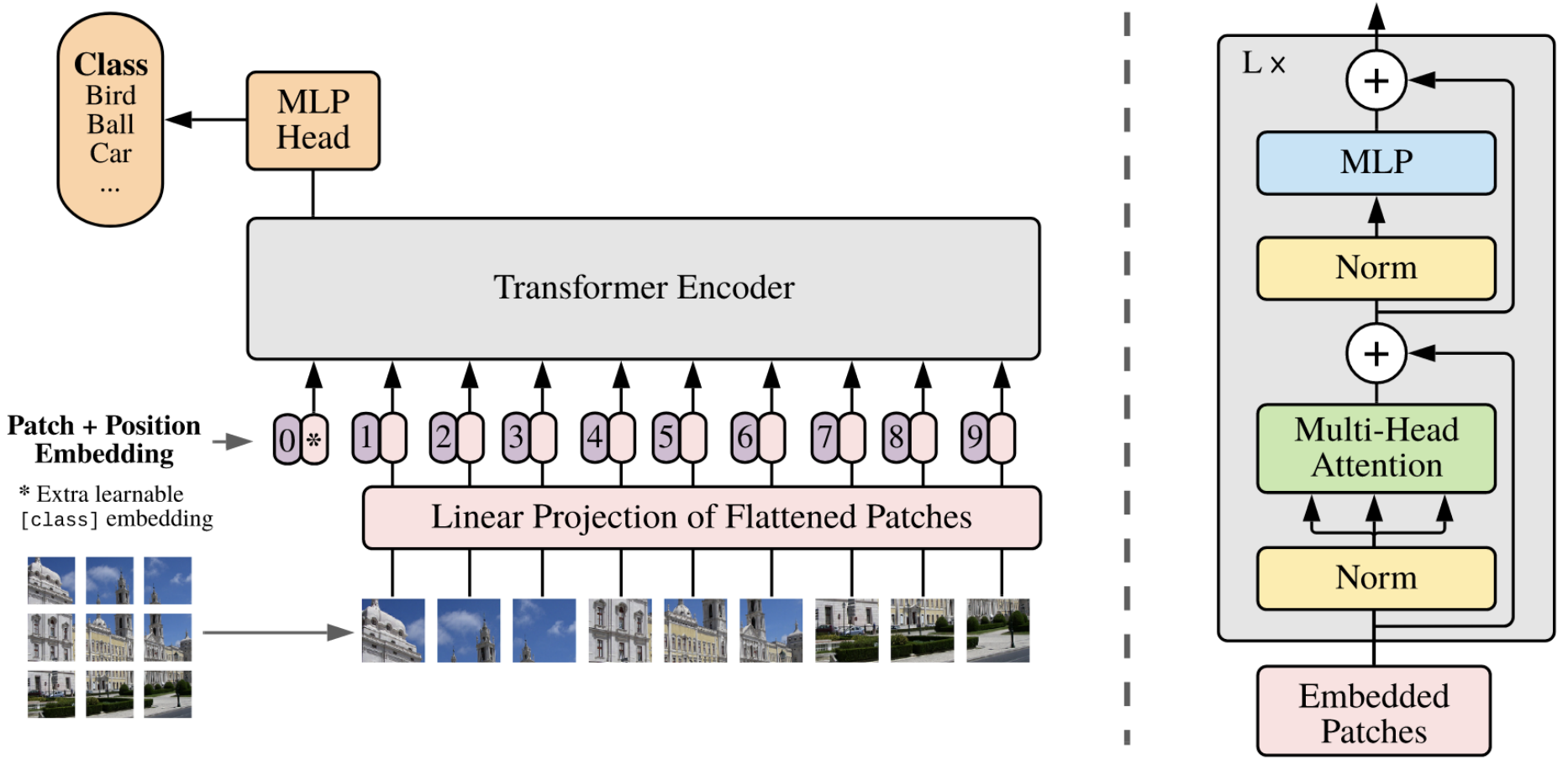
GPT family keeps growing

- GPT-3.5
- GPT-3.5-turbo
- GPT-4, GPT 4.1
- GPT-4-turbo
- GPT-4o
- o1, o3, o3-pro
- GPT-5.2 (Thinking)

Competitors

- Gemini family (Gemini Pro) (Google) (Gemini Pro 3.0, Gemini Pro 3.1, Gemini-Flash)
- Mistral 7xMoE (Open Source by Mistral.ai)
- Llama-2, Llama-3(Open Source by Meta AI), Llama-4
- Qwen3.5, Qwen3, Qwen2.5 (Alibaba)
- DeepSeekV3, DeepSeek, DeepSeek-R1 (Open Source by DeepSeek Team)
- Claude Opus 4.6 (Extended), Claude3.5 Sonnet, Haiku, etc (Anthropic)
- Grok3, Grok4 (Twitter/xAI)

Vision Transformers

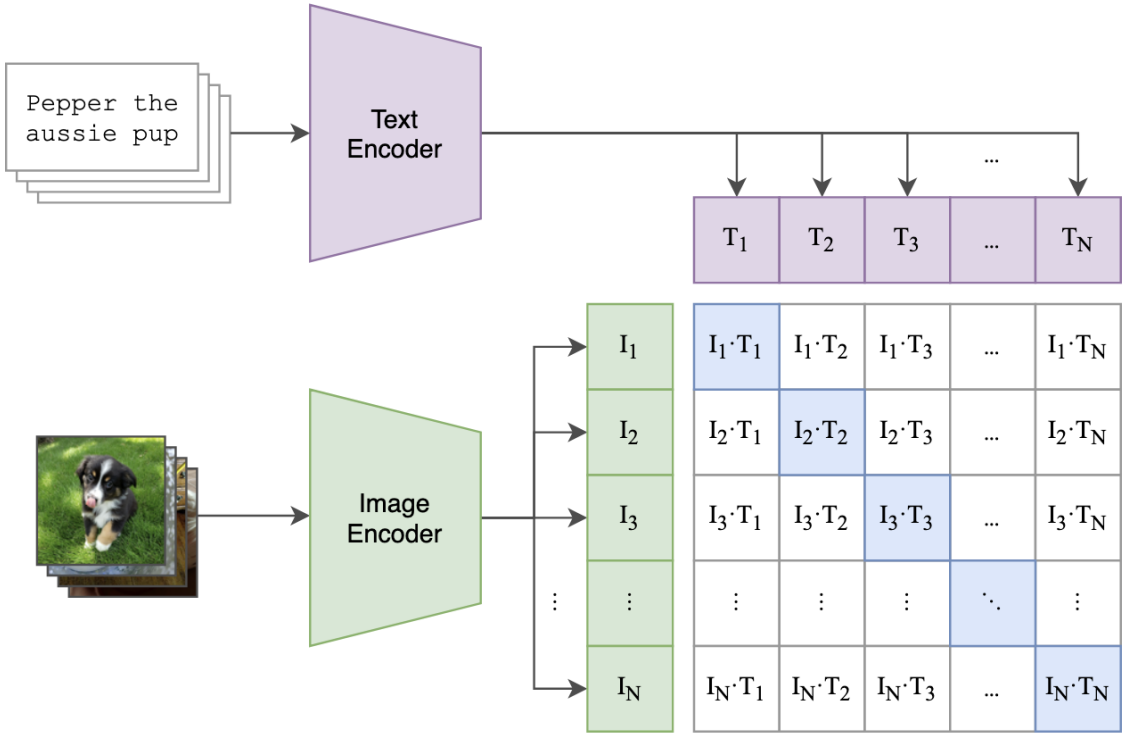


https://arxiv.org/abs/2010.11929

An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby

The CLIP Model



$$L = \sum_k \ell_1(I_k T_k) + \ell_2(I_k T_k)$$

$$\ell_1(I_k T_k) = -\log\left(\frac{\exp(sim(I_k, T_k))}{\sum_{t=1}^{2N} 1[k \neq i]\exp(sim(I_k, T_t))}\right)$$

$$\ell_2(I_k T_k) = -\log\left(\frac{\exp(sim(I_k, T_k))}{\sum_{t=1}^{2N} 1[k \neq i]\exp(sim(I_t, T_k))}\right)$$

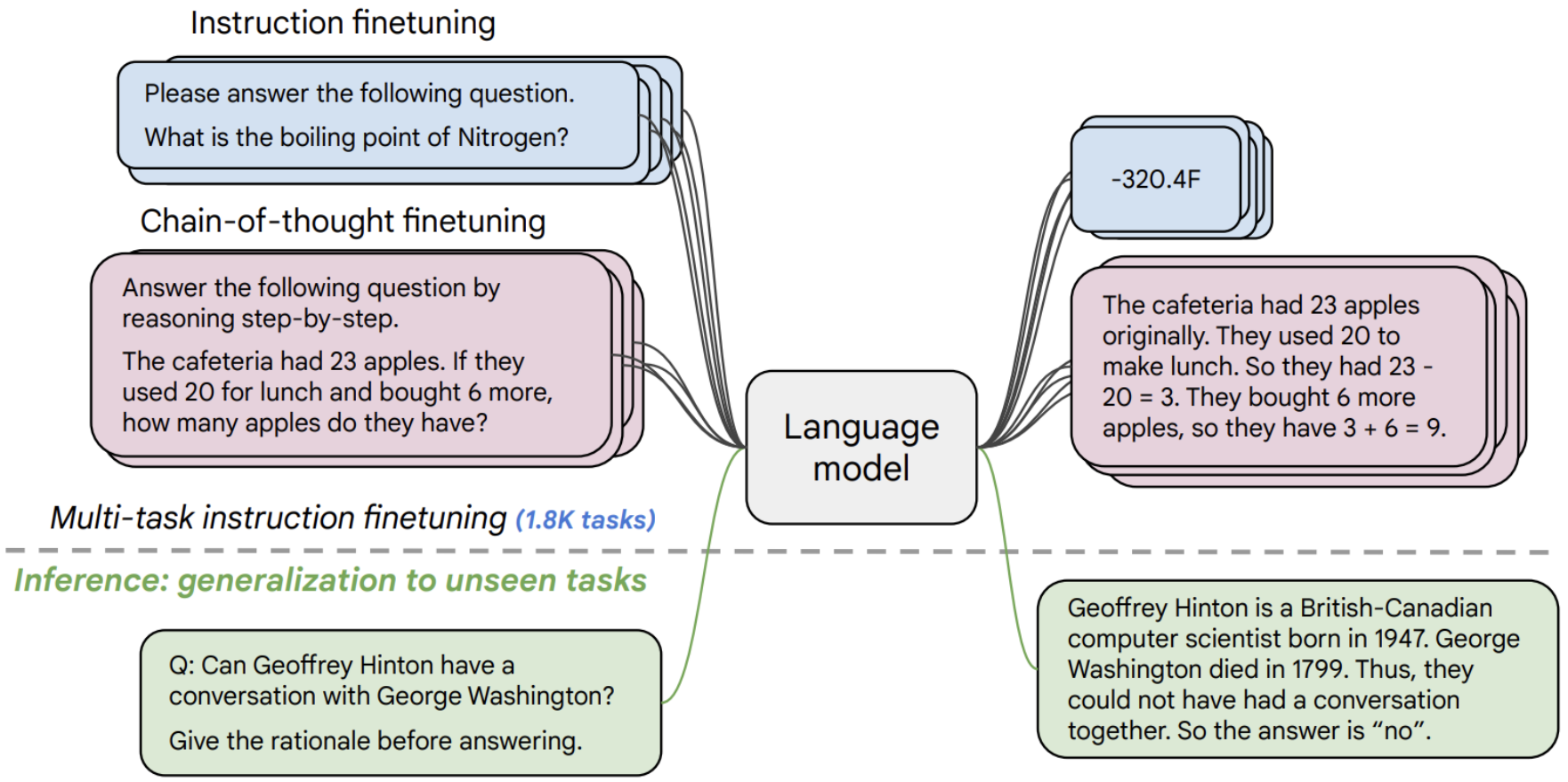
https://arxiv.org/abs/2103.00020

Learning Transferable Visual Models From Natural Language Supervision
Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, Ilya Sutskever

Next Word Prediction is limited

- Predicting the next word can lead to intelligent behavior such as the one exemplified earlier however this still limited
- What makes some of the new LLMs special? ChatGPT (GPT-3.5, 3.5 Turbo, 4, 4-turbo), FLAN-T5, OPT-IML
  - SFT: Supervised Finetuning (Curated Input/Output Instruction Sets)
  - DPO: Direct Preference Optimization
  - PPO: Proximal Policy Optimization (Reinforcement Learning with Human Feedback)
  - GRPO: Group-relative Policy Optimization (DeepSeek)

Instruction Tuning (e.g. FLAN-T5 by Google)



https://arxiv.org/pdf/2210.11416.pdf

FLAN-T5

Model input (Disambiguation QA)

Q: In the following sentences, explain the antecedent of the pronoun (which thing the pronoun refers to), or state that it is ambiguous.

Sentence: The reporter and the chef will discuss their favorite dishes.

Options:
(A) They will discuss the reporter's favorite dishes
(B) They will discuss the chef's favorite dishes
(C) Ambiguous

A: Let's think step by step.

Before instruction finetuning

The reporter and the chef will discuss their favorite dishes.
The reporter and the chef will discuss the reporter's favorite dishes.
The reporter and the chef will discuss the chef's favorite dishes.
The reporter and the chef will discuss the reporter's and the chef's favorite dishes.

✖ **(doesn't answer question)**

FLAN-T5

Model input (Disambiguation QA)

Q: In the following sentences, explain the antecedent of the pronoun (which thing the pronoun refers to), or state that it is ambiguous.

Sentence: The reporter and the chef will discuss their favorite dishes.

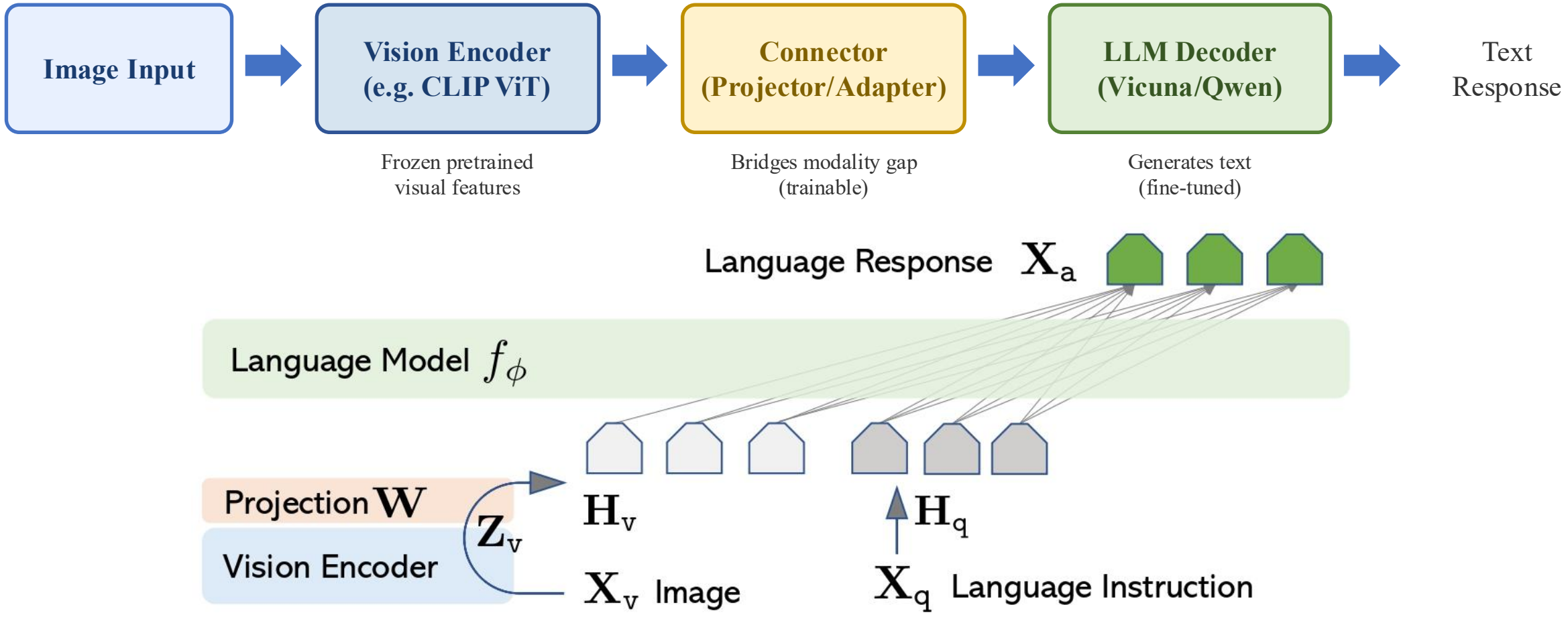
Options:
(A) They will discuss the reporter's favorite dishes
(B) They will discuss the chef's favorite dishes
(C) Ambiguous

A: Let's think step by step.

After instruction finetuning

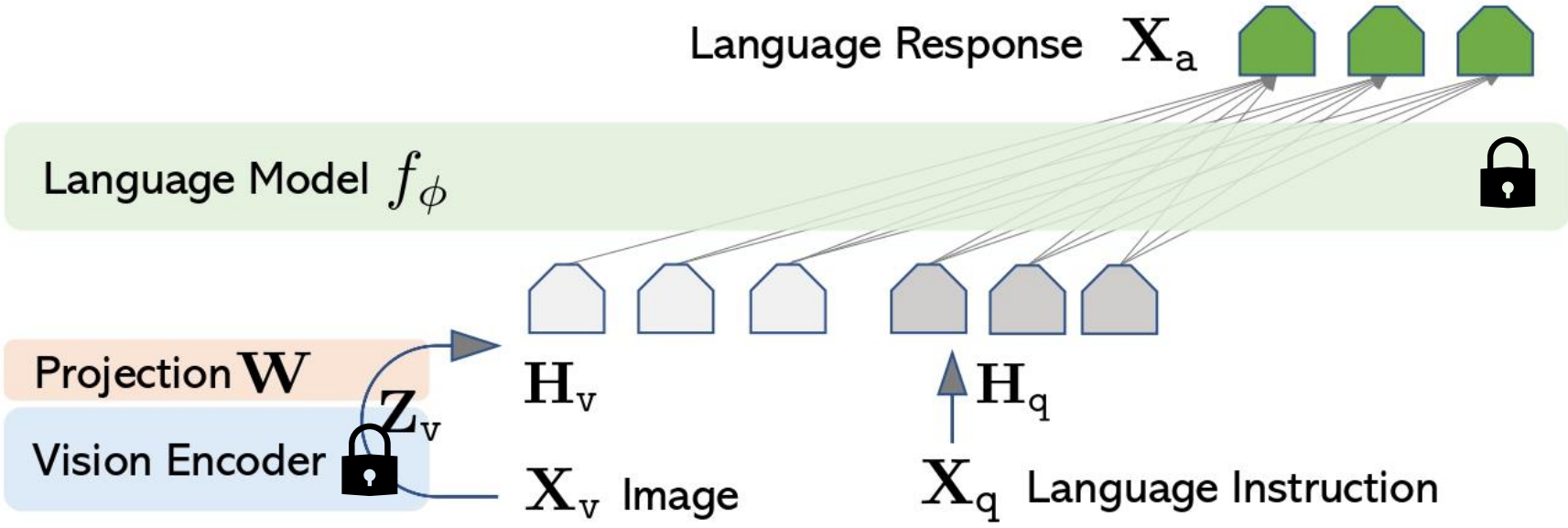
The reporter and the chef will discuss their favorite dishes does not indicate whose favorite dishes they will discuss. So, the answer is (C). ✅

Three Core Components of Llava-style Vision-Language Models



Source: [Liu et al. 2023]

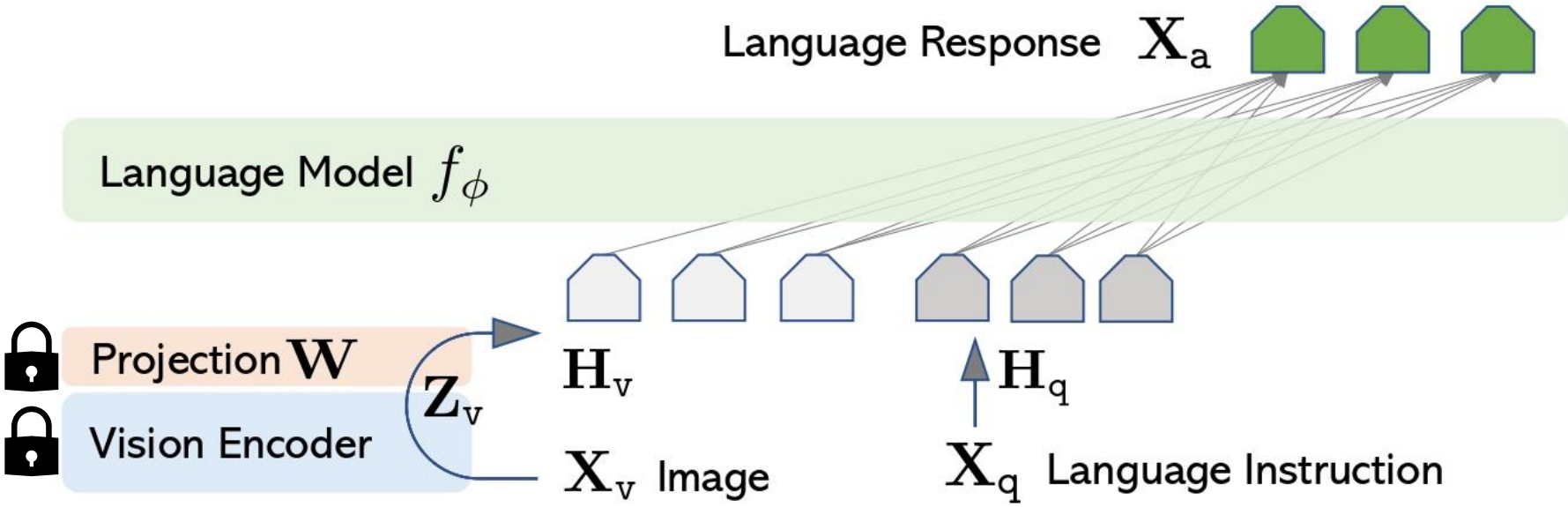
Three Core Components of Llava-style Vision-Language Models



- Training Stage 1 / 2
  - large-scale image-text pretraining with **the LLM and Visual Encoder Frozen**
  - Data: 600K filtered CC3M

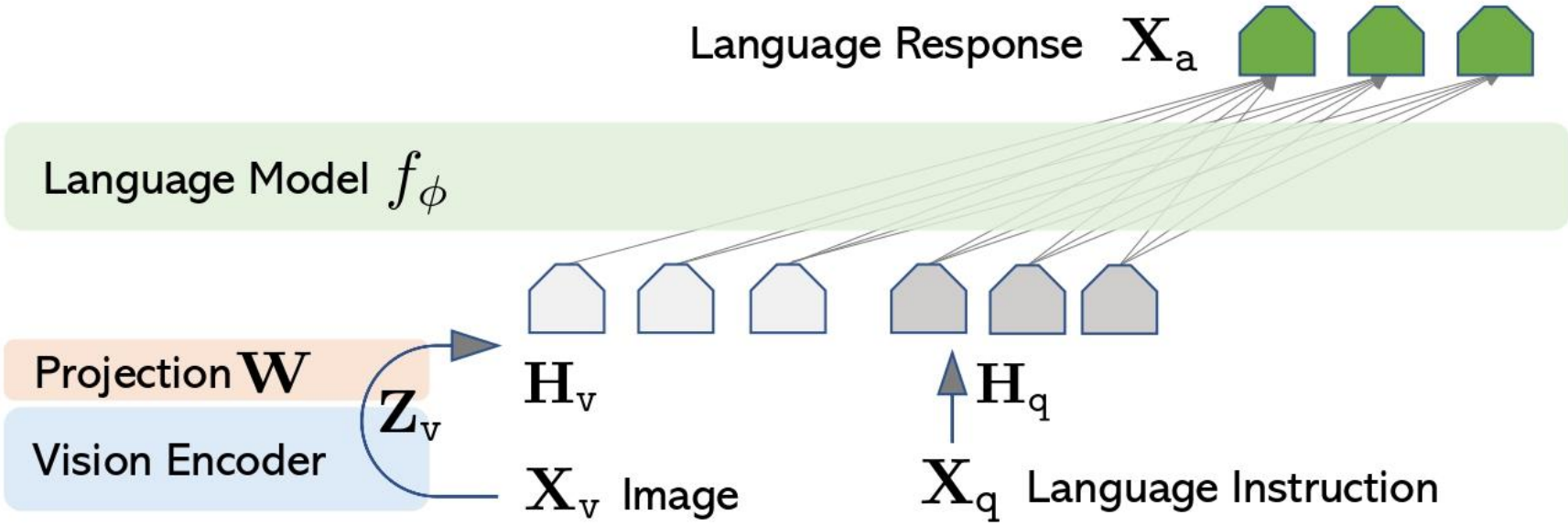
a river has burst it 's banks and has spread out onto arable farmland alongside

Three Core Components of Llava-style Vision-Language Models



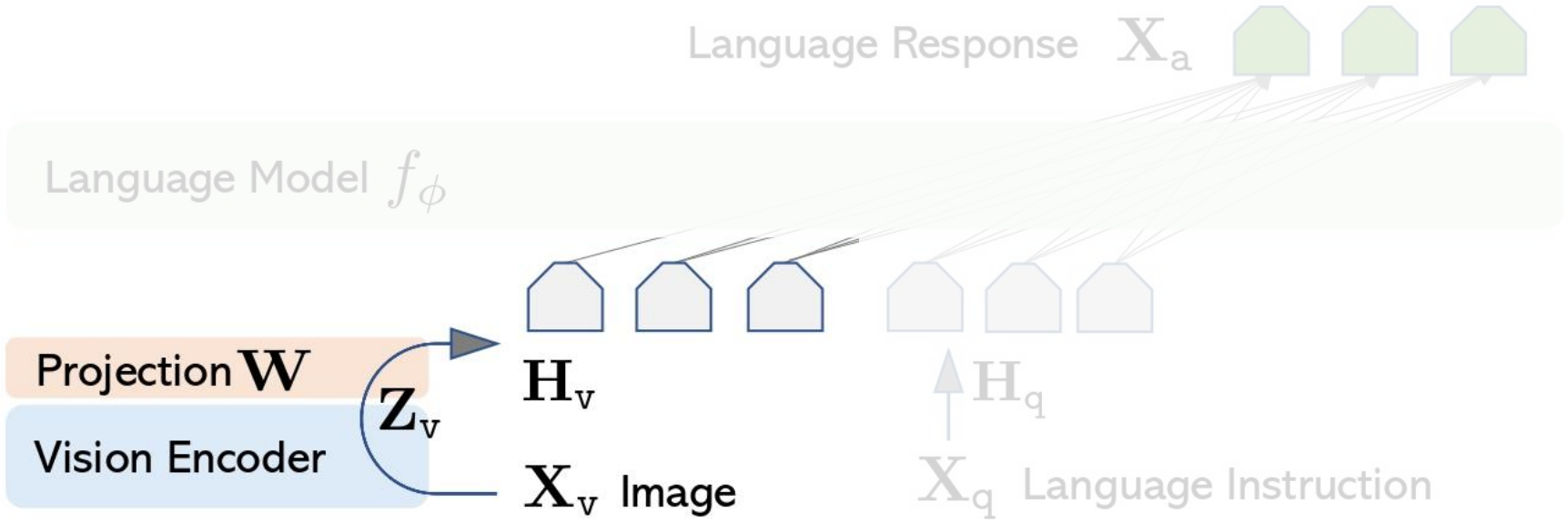
- Training Stage 2 / 2
  - visual instruction tuning stage: use 150K GPT-generated multimodal instruction-following data

Three Core Components of Llava-style Vision-Language Models



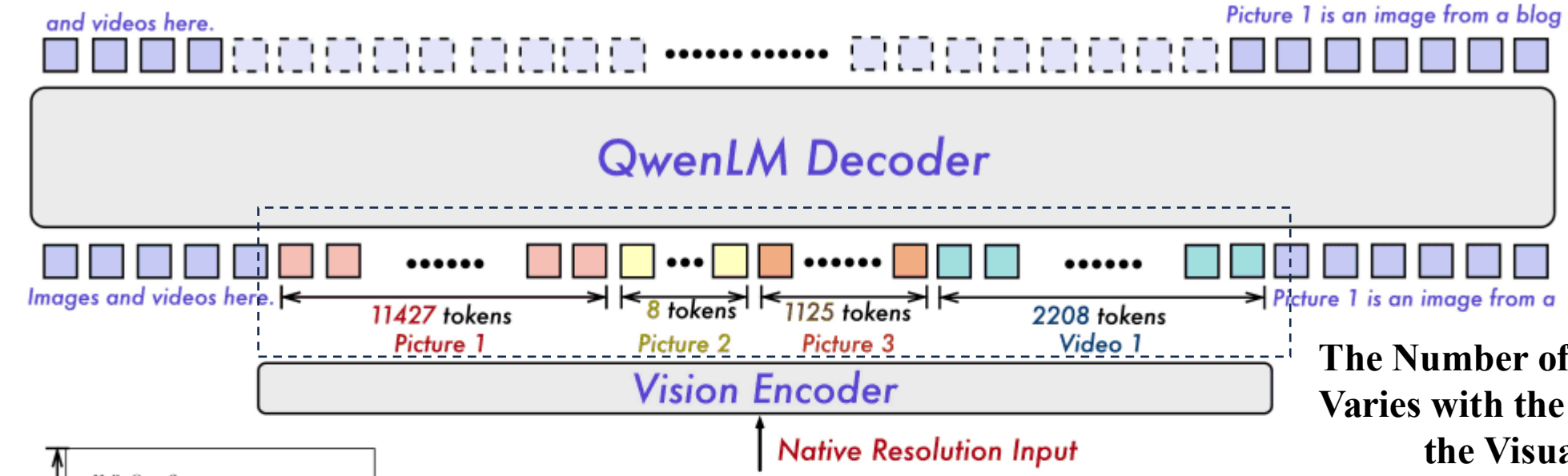
- LLaVA-1.5 Update
  - Projection Layer: Linear Projection -> MLP
  - Visual Encoder: CLIP ViT-L/14 -> **CLIP-ViT-L-336px**
  - Data Mix: Stage 1 - **558K LAION-CC-SBU**; Stage 2: 150K + **515K VQA data from academic-oriented tasks**

The Bottleneck of LLaVA: Fixed Visual Tokens for Every Image

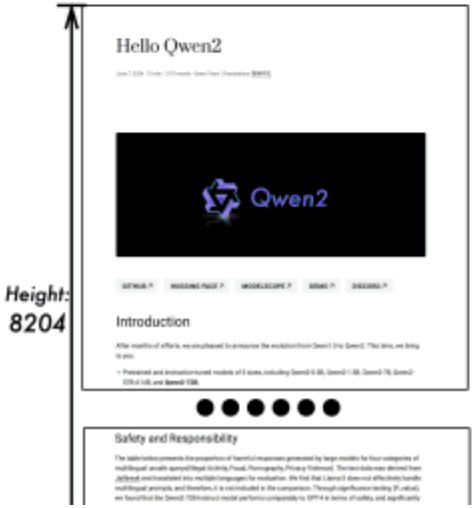


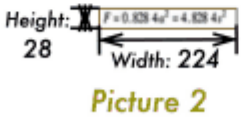
- **The limitation of early VLMs (LLaVA 1.5 and before)**
  - **Each image gets resized to 224x224 and processed by a CLIP ViT.**
  - **224x224 with patch size:14 ->256 patches per image**
  - **Details are not preserved at all.**

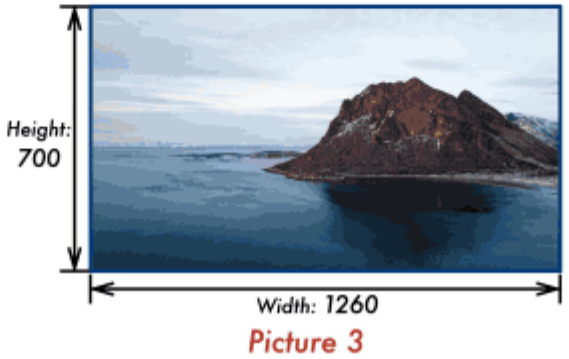
Solution: Vision-Language Models with Dynamic Resolution

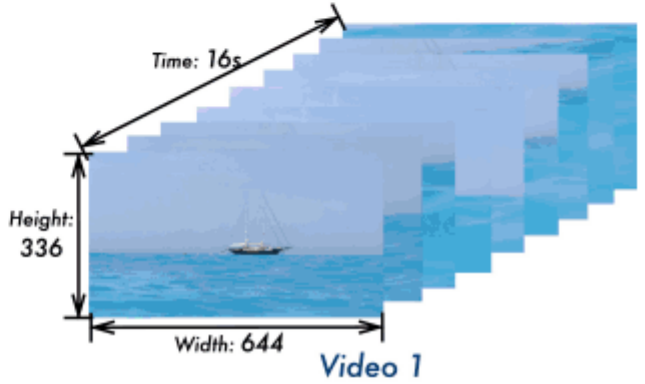


The Number of Visual Tokens Varies with the Complexity of the Visual Input!









Solution: Vision-Language Models with Dynamic Resolution

main Qwen2-VL-7B-Instruct / preprocessor_config.json

yangapku Initial commit f330ecb

raw Copy download link history blame contribute delete

```
{
  "min_pixels": 3136,
  "max_pixels": 12845056,
  "patch_size": 14,
  "temporal_patch_size": 2,
  "merge_size": 2,
  "image_mean": [
    0.48145466,
    0.4578275,
    0.40821073
  ],
  "image_std": [
    0.26862954,
    0.26130258,
    0.27577711
  ],
  "image_processor_type": "Qwen2VLImageProcessor",
  "processor_class": "Qwen2VLProcessor"
}
```

- **Step 1: Resize but keep aspect ratio**
  - If an image is too small, it can be upscaled to exceed **min_pixels.**
  - If it's too large, it can be downscaled to stay under **max_pixels.**

Resize

Resize w/ aspect ratio

Solution: Vision-Language Models with Dynamic Resolution

main Qwen2-VL-7B-Instruct / preprocessor_config.json

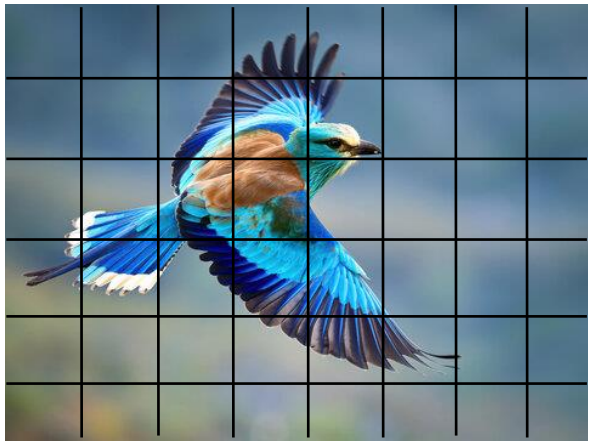
yangapku Initial commit f330ecb

raw Copy download link history blame contribute delete

```
{
  "min_pixels": 3136,
  "max_pixels": 12845056,
  "patch_size": 14,
  "temporal_patch_size": 2,
  "merge_size": 2,
  "image_mean": [
    0.48145466,
    0.4578275,
    0.40821073
  ],
  "image_std": [
    0.26862954,
    0.26130258,
    0.27577711
  ],
  "image_processor_type": "Qwen2VLImageProcessor",
  "processor_class": "Qwen2VLProcessor"
}
```

➢Step 2: Patchify

- If resized to 3584×3584: 3584/14 = 256 → 256×256 = 65,536 patch tokens
- If resized to 56×56: 56/14 = 4 → 4×4 = 16 patch tokens



The number of patches is no longer a fixed number.

Solution: Vision-Language Models with Dynamic Resolution

main / Qwen2-VL-7B-Instruct / preprocessor_config.json

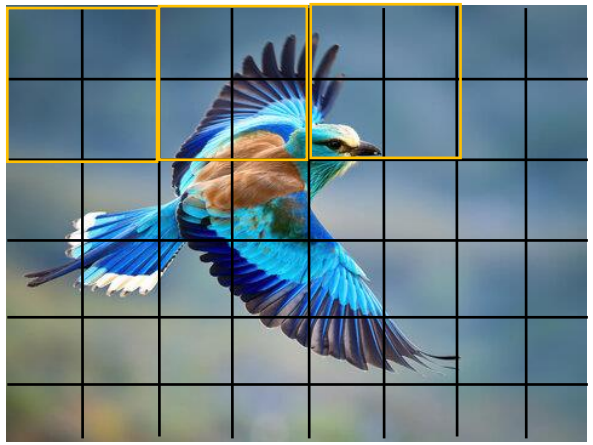
yangapku Initial commit f330ecb

raw | Copy download link | history | blame | contribute | delete

```
{
  "min_pixels": 3136,
  "max_pixels": 12845056,
  "patch_size": 14,
  "temporal_patch_size": 2,
  "merge_size": 2,
  "image_mean": [
    0.48145466,
    0.4578275,
    0.40821073
  ],
  "image_std": [
    0.26862954,
    0.26130258,
    0.27577711
  ],
  "image_processor_type": "Qwen2VLImageProcessor",
  "processor_class": "Qwen2VLProcessor"
}
```

➢Step 3: Token compression

- ➢ Qwen2-VL applies a simple MLP "compression" that merges **adjacent tokens** in token space so the LLM doesn't get overwhelmed by huge grids.
- ➢ Indicated by "merge_size" in the config.



if 48 patch tokens -> 12 visual tokens in the LLM input

Questions?