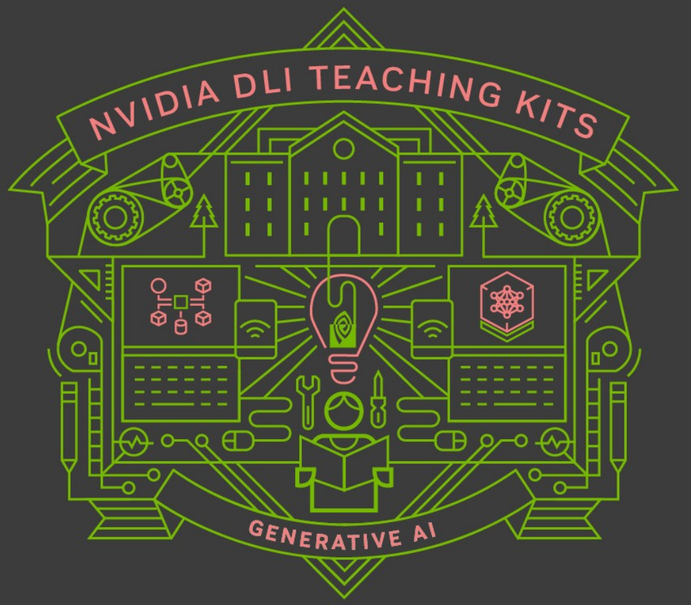
Lecture 4.1 - LLM Architecture Variants

Generative AI Teaching Kit



This lecture

- Review of the original Transformer
- Encoder only Models
- Decoder only Models
- Encoder-decoder models
- New Variants

Review of the original Transformer

Recap: The Original Transformer

Last time we saw the biggest innovation in machine learning in the last decade, the **Transformer**.

The paper, “Attention Is All You Need”, introduced a new paradigm for processing sequences in parallel with self-attention.

This model was used as a translation tool and was shown to outperform the state-of-the-art, even with a fraction of the compute requirements.

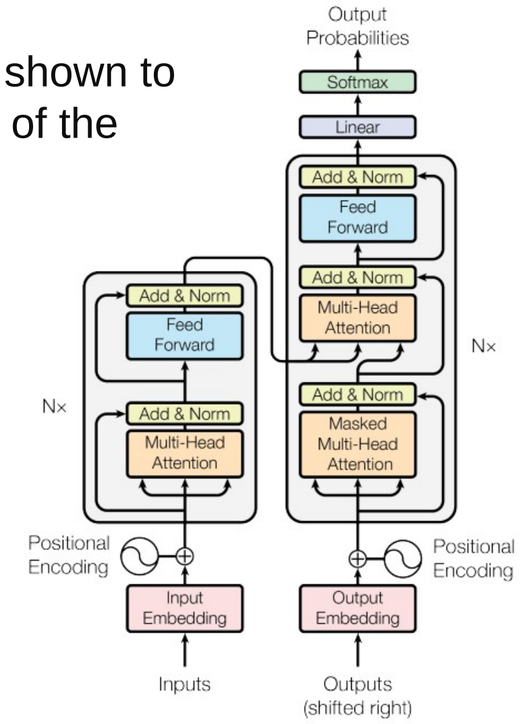


Figure 1: The Transformer - model architecture.

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

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

*Equal contribution. Listing order is random. Jakob proposed replacing RNNs with self-attention and started the effort to evaluate this idea. Ashish, with Illia, designed and implemented the first Transformer models and has been crucially involved in every aspect of this work. Noam proposed scaled dot-product attention, multi-head attention and the parameter-free position representation and became the other person involved in nearly every detail. Niki designed, implemented, tuned and evaluated countless model variants in our original codebase and tensor2tensor. Llion also experimented with novel model variants, was responsible for our initial codebase, and efficient inference and visualizations. Lukasz and Aidan spent countless long days designing various parts of and implementing tensor2tensor, replacing our earlier codebase, greatly improving results and massively accelerating our research.

†Work performed while at Google Brain.

‡Work performed while at Google Research.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

Breakthrough innovation

Architecture Details:

- Transformer Blocks/Layers: **6** encoder and **6** decoder layers.
- Hidden Dimension for the word embeddings: **1024**
- Feedforward Hidden Dimension: **4096** (ie a 4x expansion internally)
- Attention Heads for multi-headed attention: **16**
- Sequence Length: Up to 512 tokens.

Total Parameters: 213 million (213M)

Training Details

- Task: English to German Translation
- Batch Size: 131,072 tokens per batch
- Training Steps: 100 k
- Dataset: WMT 2014 English-to-German (4.5M sentence pairs).

Training Time: ~3.5 days on 8xP100 GPUs for English-to-German

Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	**$3.3 \cdot 10^{18}$**	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

Table 4: The Transformer generalizes well to English constituency parsing (Results are on Section 23 of WSJ)

Parser	Training	WSJ 23 F1
Vinyals & Kaiser el al. (2014) [37]	WSJ only, discriminative	88.3
Petrov et al. (2006) [29]	WSJ only, discriminative	90.4
Zhu et al. (2013) [40]	WSJ only, discriminative	90.4
Dyer et al. (2016) [8]	WSJ only, discriminative	91.7
Transformer (4 layers)	WSJ only, discriminative	91.3
Zhu et al. (2013) [40]	semi-supervised	91.3
Huang & Harper (2009) [14]	semi-supervised	91.3
McClosky et al. (2006) [26]	semi-supervised	92.1
Vinyals & Kaiser el al. (2014) [37]	semi-supervised	92.1
Transformer (4 layers)	semi-supervised	92.7
Luong et al. (2015) [23]	multi-task	93.0
Dyer et al. (2016) [8]	generative	93.3

Composition and Task focus

This model architecture was composed of two components:

An Encoder:
- Processes the input sequence all at once.
- Maps the input tokens into continuous vector representations using self-attention and feedforward layers.
- Captures contextual relationships across the entire sequence.
- Used in tasks like text classification and feature extraction.

A Decoder:
- Generates output tokens one by one in an autoregressive manner.
- Attends to both the encoder's output and previously generated tokens using masked self-attention.
- Produces meaningful sequences based on learned patterns (e.g., translation, text generation).

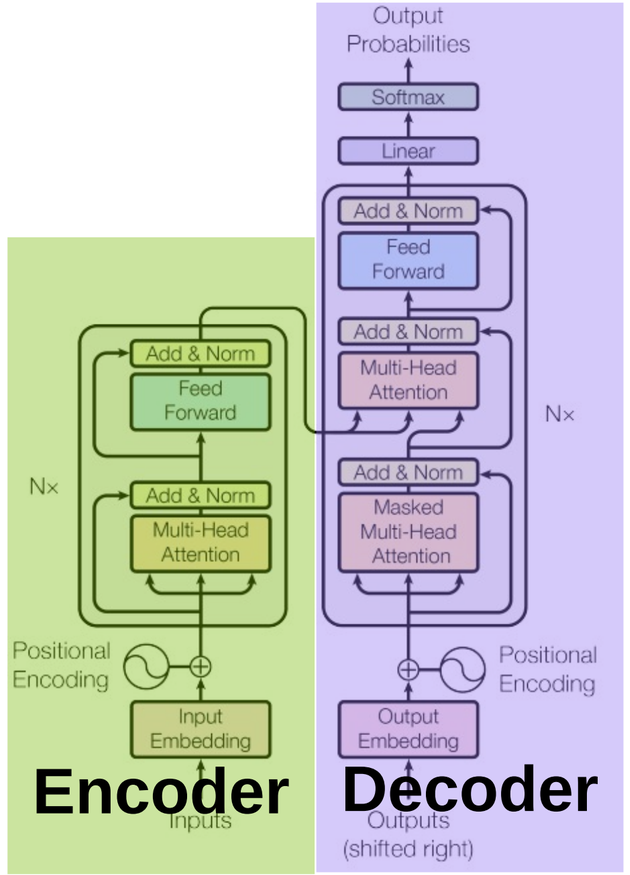


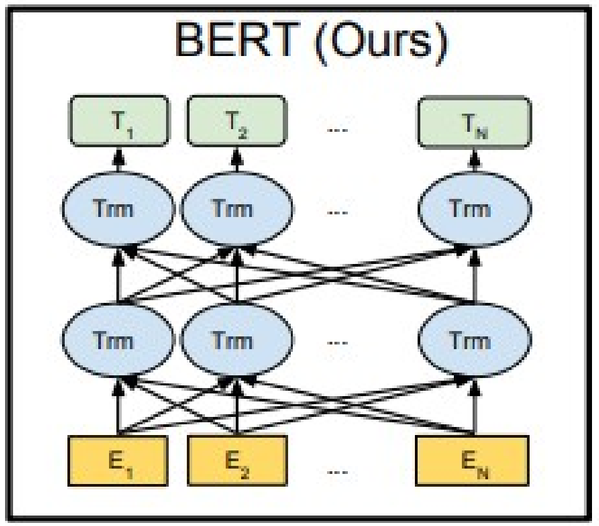
Figure 1: The Transformer - model architecture.

But do we always need both components?

Encoder only Models

Splitting the Transformer – BERT/Embeddings

If we focus on just the encoder side of the original transformer, we end up with the **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. Unlike the original Transformer (which was designed for seq-to-seq tasks like translation), BERT **focuses on deep bidirectional context learning**.

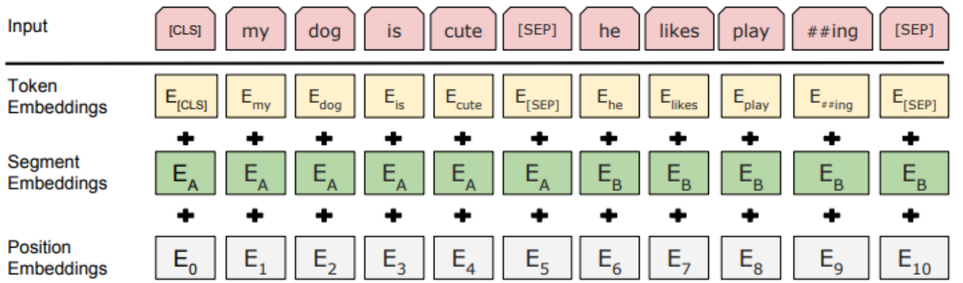


What Happens When We Keep Only the Encoder?

- The model **processes all tokens simultaneously**, rather than autoregressively generating output token-by-token.
- Instead of predicting the next token, BERT learns **contextual embeddings** using **masked language modeling (MLM)**.
- **No autoregressive decoding**, making it ideal for feature extraction rather than generation.

How Encoder Embeddings Work

- **Token Embeddings**: Each token is mapped to a high-dimensional contextualized vector.
- **[CLS] Token**: Represents a fixed-length sentence embedding, useful for tasks like classification or retrieval.
- **Layer-wise Representations**: Embeddings can be extracted from different encoder layers depending on the task.



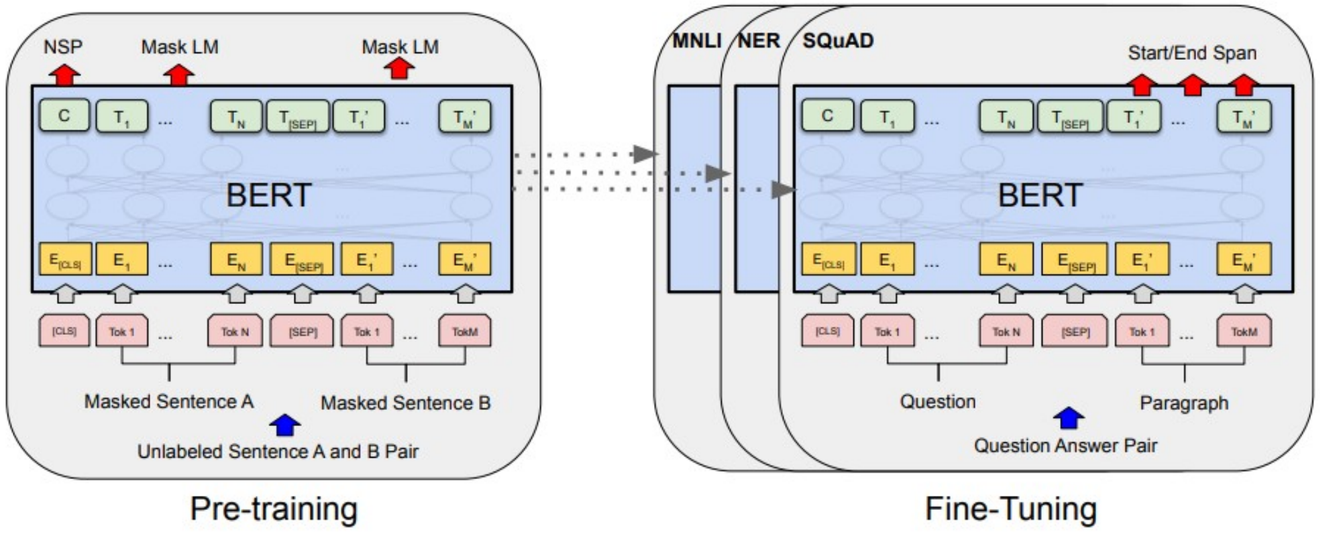
DARTMOUTH ENGINEERING | NVIDIA

Training Encoder Models - BERT

Pretraining: Learning Universal Representations
BERT undergoes self-supervised learning on massive unlabeled text corpora using two key objectives:

- Masked Language Modeling (MLM)
  - Randomly masks 15% of tokens and trains the model to predict them.
  - Encourages deep bidirectional context learning by making the model rely on both left and right words.
- Next Sentence Prediction (NSP)
  - Given two sentences, the model predicts if the second follows the first.
  - Helps capture sentence-level relationships (though later models like RoBERTa remove this step).

Pretraining Output: A general-purpose transformer capable of understanding contextual meaning across various domains.



Fine-Tuning: Adapting BERT for Specific Tasks

- After pretraining, BERT is fine-tuned on labeled datasets for specific NLP applications. Fine-tuning involves:
- Adding a Task-Specific Output Layer
- Training on a Labeled Dataset

Fine-Tuning Output: A specialized BERT model optimized for task-specific performance (e.g., sentiment analysis, QA, information retrieval).

Evolution of BERT models

The Problem with Original BERT

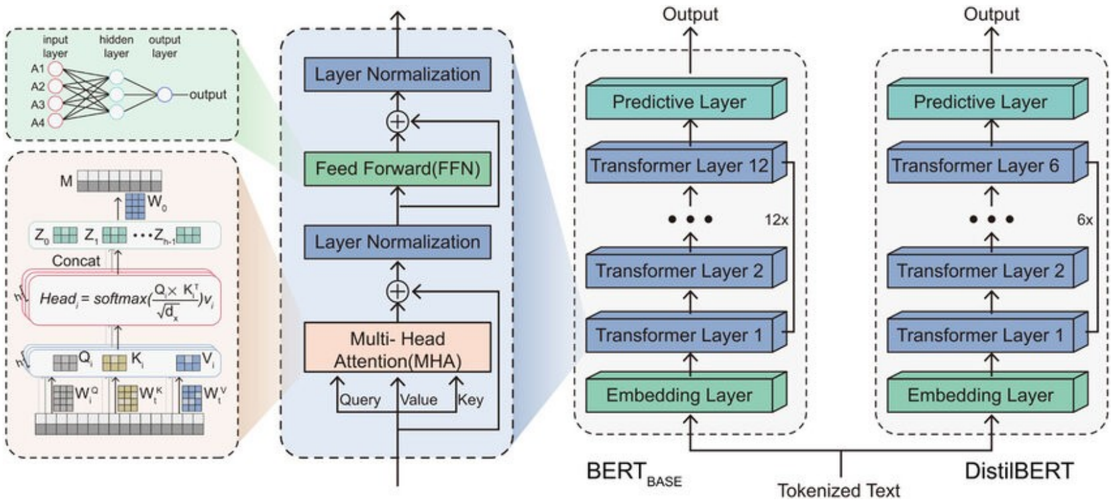
Introduced bidirectional attention, making it strong at understanding context.
But: Computationally expensive, requires large-scale training, and Next Sentence Prediction (NSP) is ineffective.

RoBERTa – More Data, No NSP

- Removed NSP, trained on 10x more data with dynamic masking.
- Better accuracy but requires even more compute than BERT.

DistilBERT – Making BERT Faster

- Used knowledge distillation to create a lighter version of BERT.
- 60% fewer parameters, 60% faster inference, retaining 95% of original BERT's accuracy.



Why This Evolution Matters

- BERT started as powerful but inefficient → newer models optimized for speed, size, and compute efficiency.
- Trade-offs: More efficient models reduce size but may lose flexibility or require different training techniques.
- Modern NLP uses these advancements to deploy transformer models at scale, even on limited hardware.

SBERT and Cross-Encoder Models

SBERT Extending BERT for Sentence Similarity

Why Standard BERT is Inefficient for Sentence Comparison:

- BERT processes each sentence independently, meaning similarity nee be computed after embedding extraction.
- This results in slow, inefficient comparisons, especially for large-scale retrieval tasks.

SBERT (Sentence-BERT)
How It Works

1. Encodes sentences separately using a shared BERT model.
2. Outputs fixed-length embeddings, which can be compared using cosin similarity.
3. Fine-tuned with tasks like semantic textual similarity (STS).

Advantages

- Fast similarity search – Ideal for retrieval and clustering.
- Fixed embeddings – Precomputed vectors allow efficient comparison.

Limitations

- Less precise for nuanced comparisons – Doesn't leverage full pairwise attention like cross-encoders.

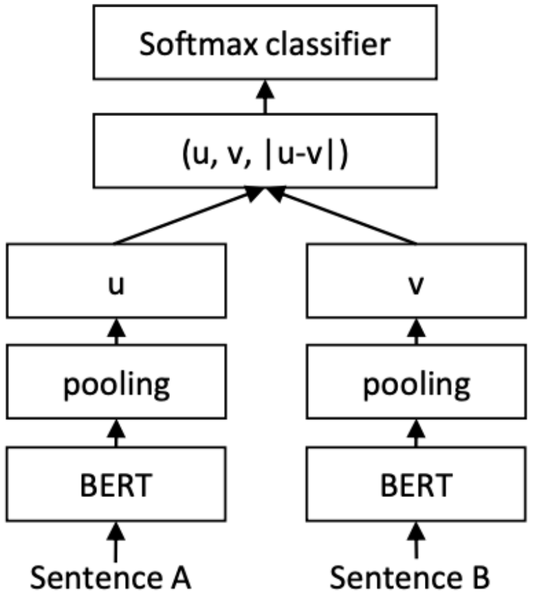


Figure 1: SBERT architecture with classification objective function, e.g., for fine-tuning on SNLI dataset. The two BERT networks have tied weights (siamese network structure).

SBERT and Cross-Encoder Models

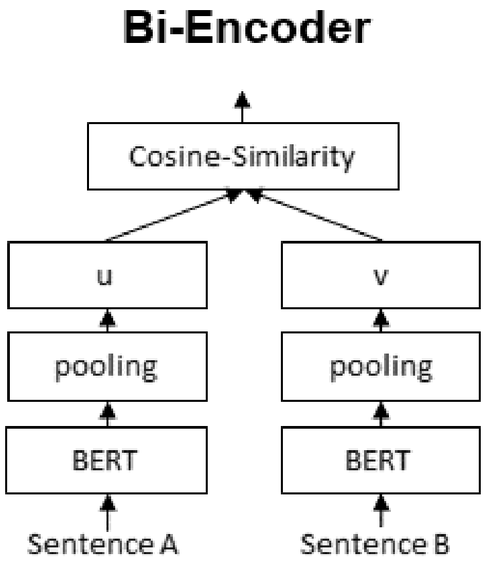
Cross-Encoders: Instead of processing sentences separately, a cross-encoder **jointly encodes both inputs**, allowing **full self-attention across sentences**.

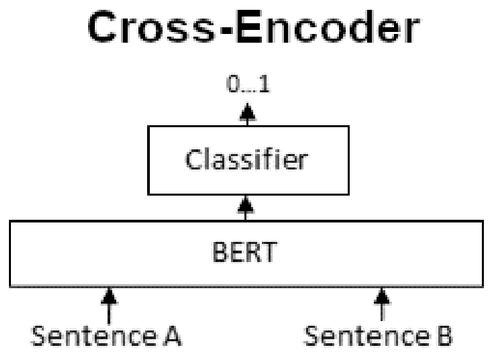
How It Works
1. The model takes both sentences **concatenated together** as input.
2. Outputs a **single scalar score** (e.g., similarity score for STS).
3. Typically fine-tuned with classification loss for ranking.

Advantages
- **Higher accuracy** – Fully utilizes attention across both sentences.
- **Better for ranking** – Used in **re-ranking pipelines** for search engines and QA.

Limitations
- **Much slower** – Requires **recomputing embeddings every time**, making it impractical for large-scale retrieval.

Choosing SBERT vs. Cross-Encoders
- **Need efficiency:** Use **SBERT** for retrieval tasks.
- **Need accuracy:** Use **Cross-Encoders** for ranking and fine-grained comparisons.
- Many NLP systems use **both together** – SBERT for initial retrieval, followed by a Cross-Encoder for refinement.





Decoder only Models

Splitting the Transformer – Autoregressive

Focusing on Just the Decoder: The Autoregressive Transformer

If we focus only on the decoder side of the original Transformer, we get an autoregressive model, such as **GPT** (Generative Pretrained Transformer). Unlike the original Transformer (which was designed for seq-to-seq tasks like translation), decoder-only models specialize in generating text token-by-token, making them ideal for open-ended text generation.

What Happens When We Keep Only the Decoder?

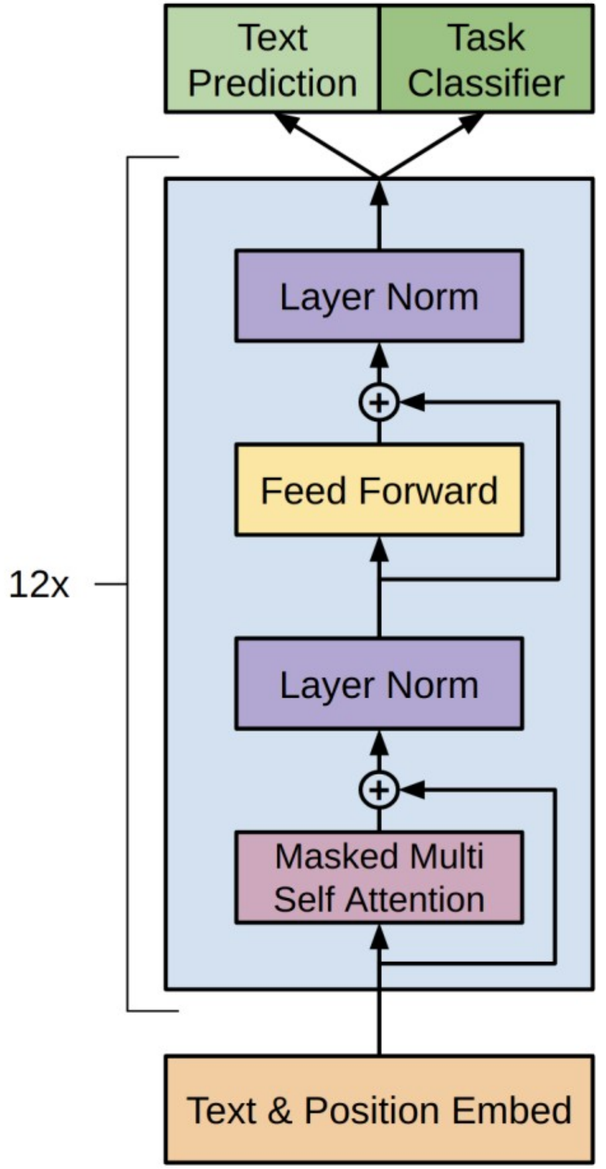
- The model generates tokens sequentially, predicting the next token based on previous tokens.
- Uses causal (unidirectional) self-attention, meaning tokens cannot attend to future words.
- Unlike encoder-based models (e.g., BERT), decoder-only models do not generate embeddings for entire sequences at once—they generate text step-by-step.

How Decoder-Based Models Generate Text

- Token-by-Token Generation: Each new token is conditioned on all previously generated tokens.
- Autoregressive Sampling: Uses methods like greedy decoding, beam search, or nucleus sampling to determine the next token.
- No Fixed-Length Representations: Unlike encoder models that output a static embedding, decoder models produce dynamic text sequences, making them useful for chatbots, story generation, and machine translation.

Key Differences from Encoder-Only Models

- Optimized for generation, rather than feature extraction.
- Uses causal self-attention, preventing tokens from seeing the future.
- Can generate arbitrarily long text, making them flexible for applications like chatbots, dialogue models, and creative writing.



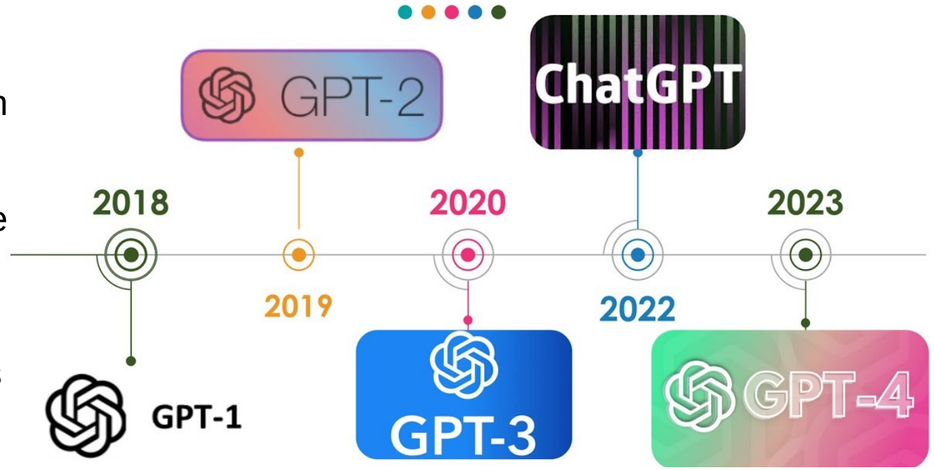
GPT and the Decoder Models

GPT-1 (2018) – Introducing Autoregressive Pretraining
The first Generative Pretrained Transformer (GPT-1) introduced the idea of pretraining on large text corpora using causal (unidirectional) self-attention. It was trained on BookCorpus (7000 books) using a left-to-right language model objective, meaning it predicted the next word based on previous context. While effective for text generation, it lacked fine-tuning capabilities for downstream tasks.

GPT-2 (2019) – Scaling Up and Zero-Shot Learning
GPT-2 significantly increased model size (up to 1.5B parameters) and was trained on a much larger dataset (WebText, 40GB of internet text). Unlike GPT-1, it demonstrated strong zero-shot learning, meaning it could perform tasks like translation and summarization without explicit fine-tuning. OpenAI initially withheld its release due to concerns over potential misuse in text generation.

GPT-3 (2020) – Massive Scale and Few-Shot Learning
GPT-3 pushed scale even further, with 175B parameters, making it one of the largest models of its time. It introduced few-shot learning, where the model could generalize to new tasks using only a few examples provided in-context. This made it far more flexible, reducing the need for task-specific fine-tuning. However, it was computationally expensive and sometimes generated incoherent or biased responses due to training data limitations.

GPT-4 (2023) – Improved Reasoning and Multimodality
GPT-4 refined previous architectures with better alignment, stronger reasoning capabilities, and improved factual accuracy. It introduced multimodal capabilities, allowing it to process both text and images. Compared to GPT-3, it was more reliable, creative, and better at nuanced instructions, reducing biases and hallucinations through improved training techniques and human feedback alignment.



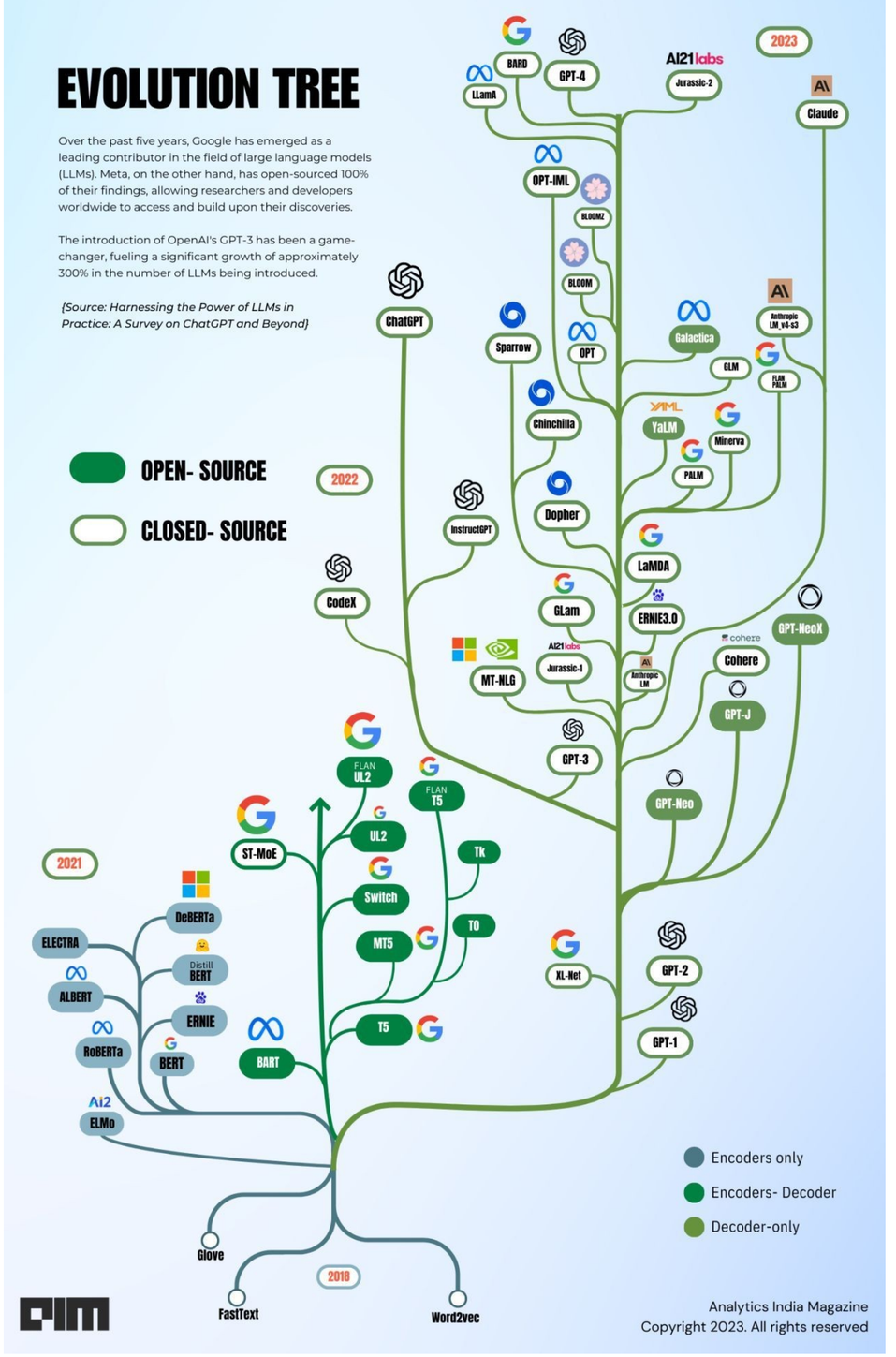
Evolution of Decoder Models

2019–2020: Scaling Transformer Decoders
Following GPT-2's success, researchers realized that scaling decoder-only transformers significantly improved language generation. Models like GPT-3 (175B parameters, 2020) demonstrated that sheer size enabled few-shot learning, allowing models to perform new tasks with minimal examples. However, these models were computationally expensive, prone to hallucinations, and had limited reasoning abilities.

2021–2022: Efficiency and Alignment
As models grew, focus shifted toward optimizing inference and improving factual accuracy. Approaches like Mixture of Experts (MoE) (e.g., GLaM, 2021) enabled selective activation of model parameters, reducing compute costs while maintaining performance. Alignment techniques such as Reinforcement Learning from Human Feedback (RLHF), popularized by InstructGPT (2022), made models safer and more useful for practical applications.

2023–2024: Multimodality and Reasoning
With GPT-4 (2023), models became multimodal, processing text and images together. Advances in long-context understanding (e.g., Anthropic's Claude 2) and retrieval-augmented generation (RAG) enabled more reliable knowledge-based responses. Research also explored smaller, specialized models (e.g., Meta's LLaMA series) to make deployment more accessible.

2025 and Beyond: Specialization and Efficiency
By 2025, decoder models are expected to become more efficient, customizable, and domain-specific. Innovations in speculative decoding, dynamic token generation, and hardware-optimized architectures will likely improve speed and affordability. The rise of agent-like models, capable of long-term memory and planning, will push the boundaries of autonomous decision-making beyond simple text generation.



Dominance of Decoder Models

While many models are still being released that are Encoder based, the vast majority of effort is centered around decoder-only models.

Early Use Cases: Encoders and Encoder-Decoder Models

- Encoders (BERT, RoBERTa, SBERT, etc.) excel at understanding language but are limited to classification and retrieval tasks.
- Encoder-Decoder (T5, BART, etc.) were designed for sequence-to-sequence tasks like translation and summarization but required more complex architectures.
- Limitation: Neither approach was optimized for open-ended text generation, leading to decoder-only models emerging as the dominant paradigm.

Why Decoder-Only Models Took Over

- Optimized for Autoregressive Generation: Predicting one token at a time allows for flexible, dynamic text generation.
- Scaling Laws Favor Decoders: Research (Kaplan et al., 2020) showed that bigger models trained autoregressively outperform alternative architectures given enough compute.
- General-Purpose Capability: Models like GPT-3 and GPT-4 adapt to a wide range of tasks without task-specific architectures, making them ideal for chatbots, coding assistants, and creative writing.

Dominance in Industry & Research

- NLP Applications: ChatGPT, Claude, Gemini, and LLaMA all use decoder-only models because they are superior at long-form generation and instruction following.
- Multimodality Expansion: GPT-4 and Gemini introduced text + image processing, further solidifying decoder models as the backbone of AI.
- Fine-Tuning and Customization: Open-source decoders like LLaMA-2, Mistral, and Falcon have enabled research and deployment at various scales.

Encoder-decoder models

Evolving the Original Transformer Model

While the original transformer was introduced as an encode-decoder format. Those individual families have shown significantly more development than the dual format.

The Original Encoder-Decoder Transformer (2017)

- Introduced in “Attention Is All You Need”, designed for sequence-to-sequence tasks like machine translation.
- The encoder processes input into contextual representations, while the decoder generates output token by token.
- Used self-attention in both encoder and decoder, with cross-attention in the decoder to connect input and output.

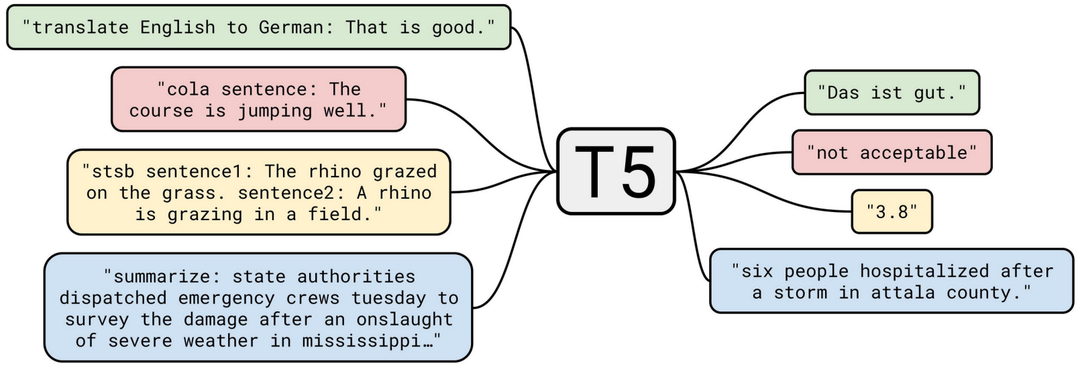
Early Innovations (2018–2020)

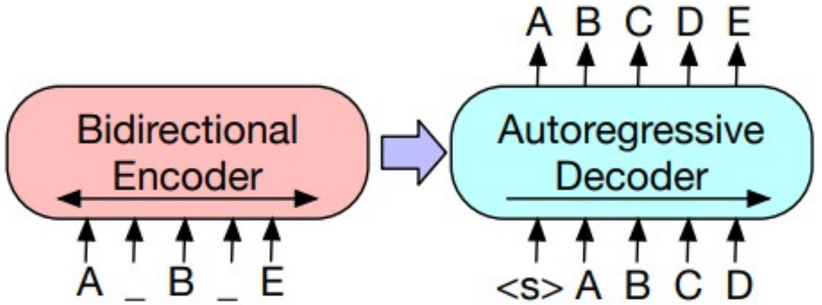
1. BERT’s Influence on Pretraining
- T5 (2019): Treated all NLP tasks as text-to-text problems, unifying classification, translation, and summarization.
- BART (2020): Used denoising pretraining to improve robustness in text generation.

2. Hybrid Approaches for Better Representations
- MASS (2019): Combined masked token prediction with sequence-to-sequence training.
- mBART (2020): Extended BART to multilingual settings, improving cross-lingual translation.

Challenges & Decline in Adoption (2021–2023)

- Computational inefficiency: Training both an encoder and a decoder made models slower and more memory-intensive.
- Decoder-only models (GPT, LLaMA) became dominant for generation, outperforming encoder-decoder models in open-ended tasks.
- Encoder-only models (BERT, RoBERTa) remained superior for retrieval and classification, leaving encoder-decoder models in a niche space.

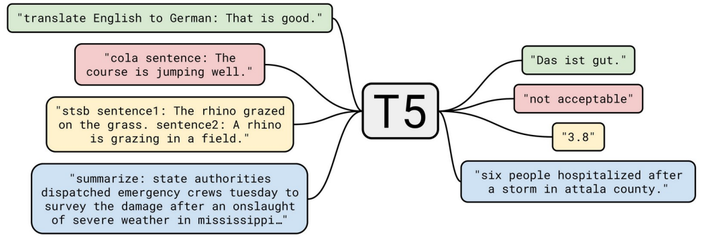




T5 (Text-to-Text Transfer Transformer) and Seq2Seq Models

1. T5 (Text-to-Text Transfer Transformer) – 2019

Reformulated all NLP tasks as text-to-text problems, where both input and output are in natural language.
Architecture: Encoder-Decoder transformer, similar to the original transformer but designed for general-purpose NLP.
Trained on C4 dataset (Colossal Clean Crawled Corpus), a massive web-scraped dataset.



Advantages:

- Unified approach: Works for classification, translation, summarization, and QA with the same architecture.
- Strong fine-tuning performance across diverse NLP benchmarks.

Limitations:

- Expensive to train due to its encoder-decoder structure.
- Not instruction-tuned, meaning it required fine-tuning per task.

2. Flan-T5 – Instruction-Tuning for Better Generalization (2022–2023)

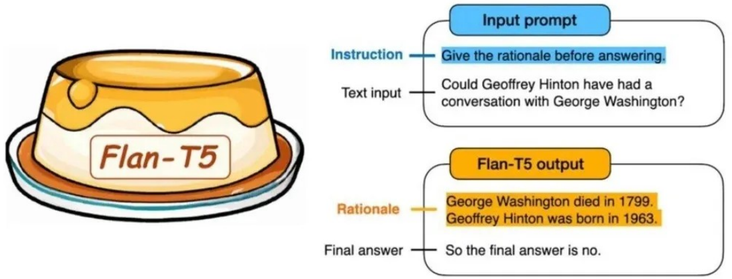
Built on T5, but trained with instruction tuning, meaning it learned from a wide variety of task prompts instead of requiring separate fine-tuning.

Training Enhancements:

- Multi-task learning: Exposed to thousands of NLP tasks to improve generalization.
- Better zero-shot and few-shot learning than original T5.

Advantages:

- Stronger out-of-the-box performance on unseen tasks.
- More efficient than GPT-3 while achieving comparable results.
- Smaller, accessible open-source versions make it practical for real-world applications.



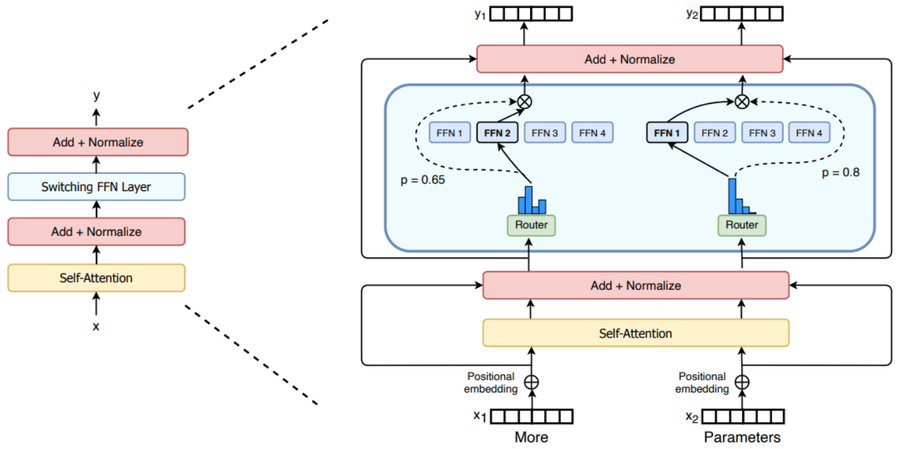
Limitations:

- Still limited to text-based tasks, unlike multimodal models like GPT-4 or Gemini.
- Larger versions (Flan-T5-XL/XXL) require significant compute for fine-tuning.

New Variants

Beyond the Transformer – Mixture-of-Experts

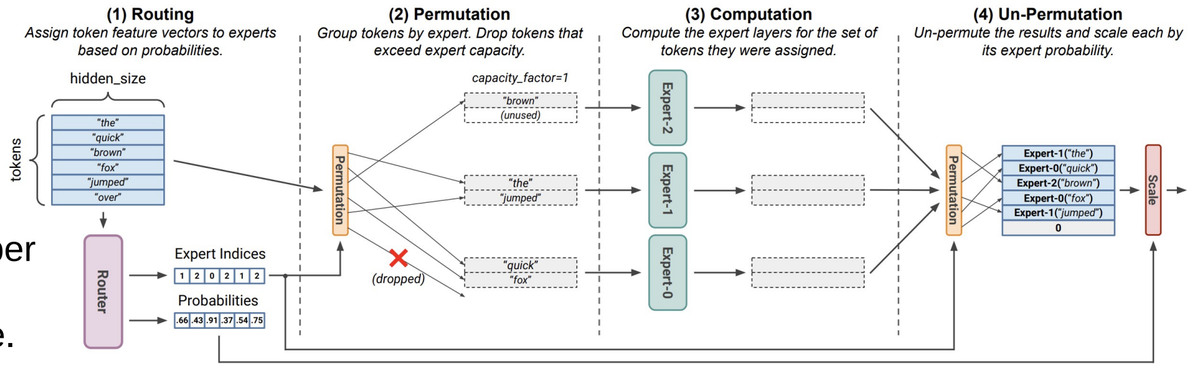
What is Mixture of Experts (MoE)?
A neural network design where only a subset of model parameters are activated per input, reducing compute costs while maintaining high capacity.
Uses a router mechanism to select which “expert” subnetworks process each token.



Why MoE?
- Increases model size without proportional compute cost (e.g., GLaM, Switch Transformer).
- Improves efficiency by activating only relevant experts per input, rather than the entire model.
- Scales effectively, making large models more computationally feasible.

Challenges of MoE
- Complex training dynamics (balancing expert utilization and load balancing).
- Increased memory requirements for storing multiple expert networks.
- Harder to deploy compared to dense models like GPT-3.

Key Applications
- Google’s Switch Transformer (2021): 1.6T parameters, but only 1/64 active per token.
- GLaM (Google, 2021): Achieved GPT-3 level performance with less compute.
- Recent MoE Models (2023–2025): Used in open-source LLMs and hybrid architectures for efficiency.



Beyond the Transformer – Multimodal

Multimodal Models are AI models that process and generate multiple types of data (e.g., text, images, audio, video).
Unlike text-only transformers (GPT, BERT), multimodal models integrate different input formats within a unified architecture.

1. Vision-Language Models (VLMs)

- CLIP (2021): Learned joint text-image embeddings, enabling zero-shot classification.
- Flamingo (2022): Fine-tuned for image captioning and reasoning using few-shot learning.

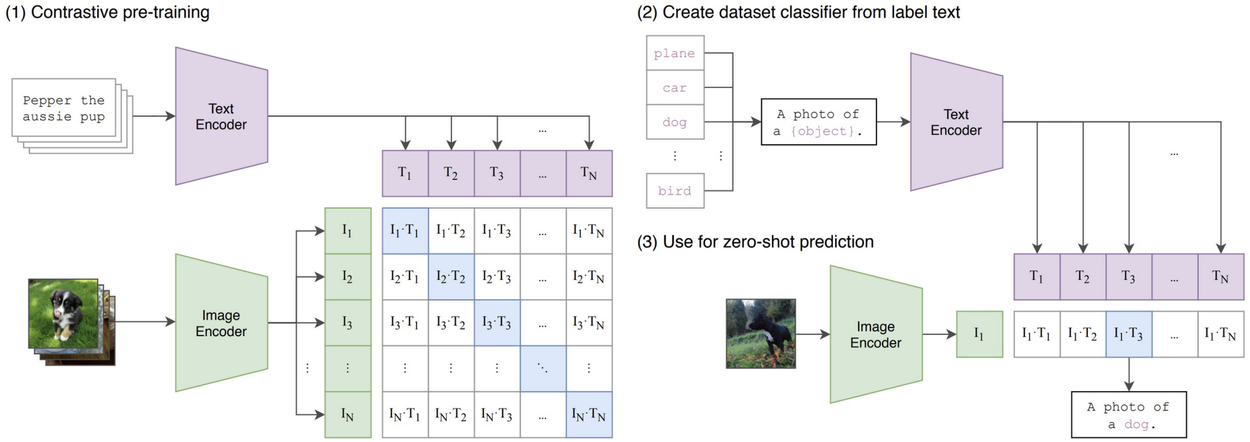
2. Multimodal Transformers

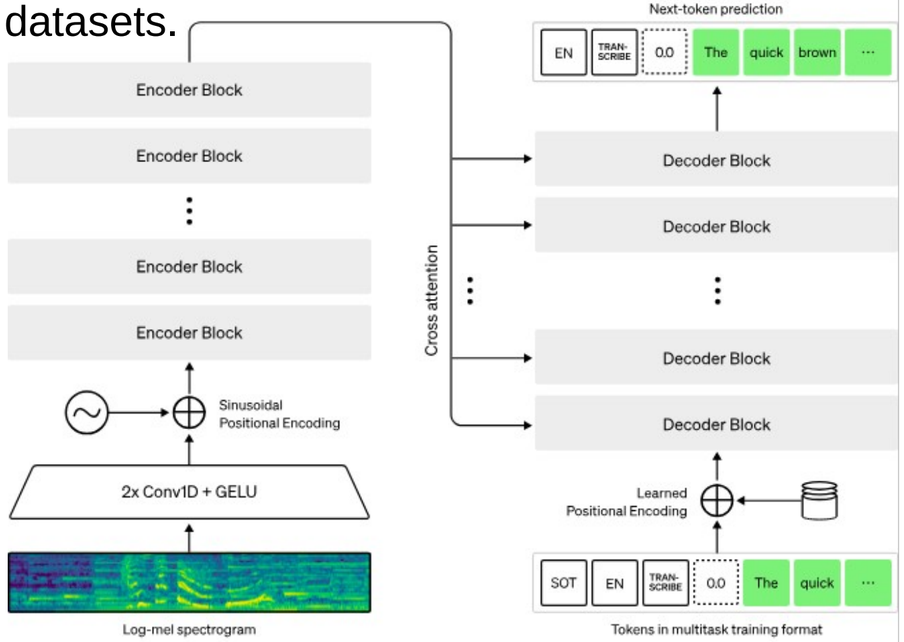
- PaLI (2022) & Gemini (2024): Use encoder-decoder architectures to jointly process text and images.
- GPT-4V (2023): Extended GPT-4 with visual processing, enabling image-based reasoning.

3. Speech & Audio Integration

- Whisper (2022): High-quality speech recognition trained on diverse multilingual datasets.
- MM1 (2024, Meta): Explored text, image, and speech in a single transformer.

We'll learn more about these later in the course!



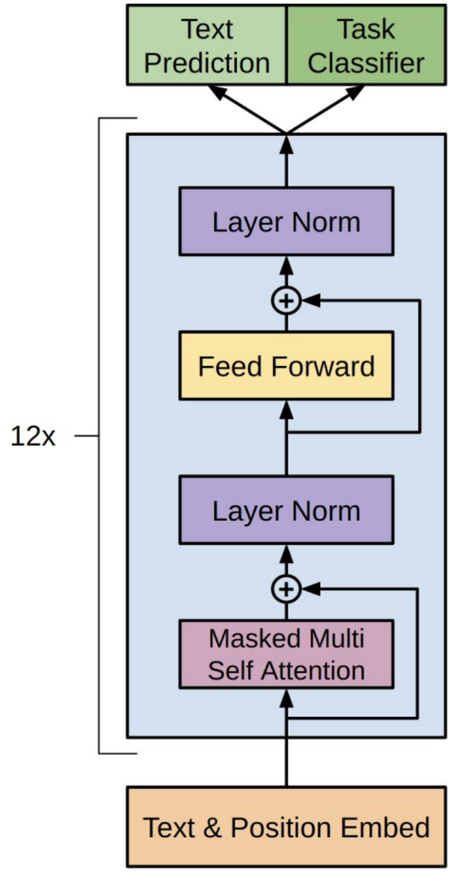


Wrap Up

LLM Architecture Variants

- Today we introduced a number of variants of the original transformer architecture.
- We saw the encoder-only models which can be used to generate highly enriched embedding vectors that can be used for several applications.
- The decoder-only LLMs were introduced, the basis of models like GPT and Llama which have dominated the GenAI landscape.
- We also saw the continuation of the encode—decoder architecture and,
- Some of the new variants of LLMs including the Mixture of Experts and Multimodal models

---



Thank you!