An In-Depth Look at the Transformer Based Models

Yule Wang, PhD · Follow
14 min read · Mar 17

— — BERT, GPT, T5, BART, and XLNet: ***Training Objectives and Architectures*** *Comprehensively Compared*

ChatGPT is primarily based on GPT-3, a Transformer decoder-only model. GPT-3 exhibits exceptional contextual learning abilities through its autoregressive pre-training process, which does not necessarily require any further fine-tuning on downstream tasks. GPT-4, released on Tuesday, also follows an autoregressive decoder-only approach. Meanwhile, Google's recent announced PaLM-E (March 10, 2023), an embodied multimodal model, abandons their valued encoder component, in favor of a decoder-only architecture to address multi-tasks

in a unifying neural network setting. This article aims to thoroughly compare different Transformer-based architectures and pretraining objectives for different large language models (LLMs), exploring their advantages and disadvantages, rather than discussing the newly revealed GPT-4 or PaML-E.

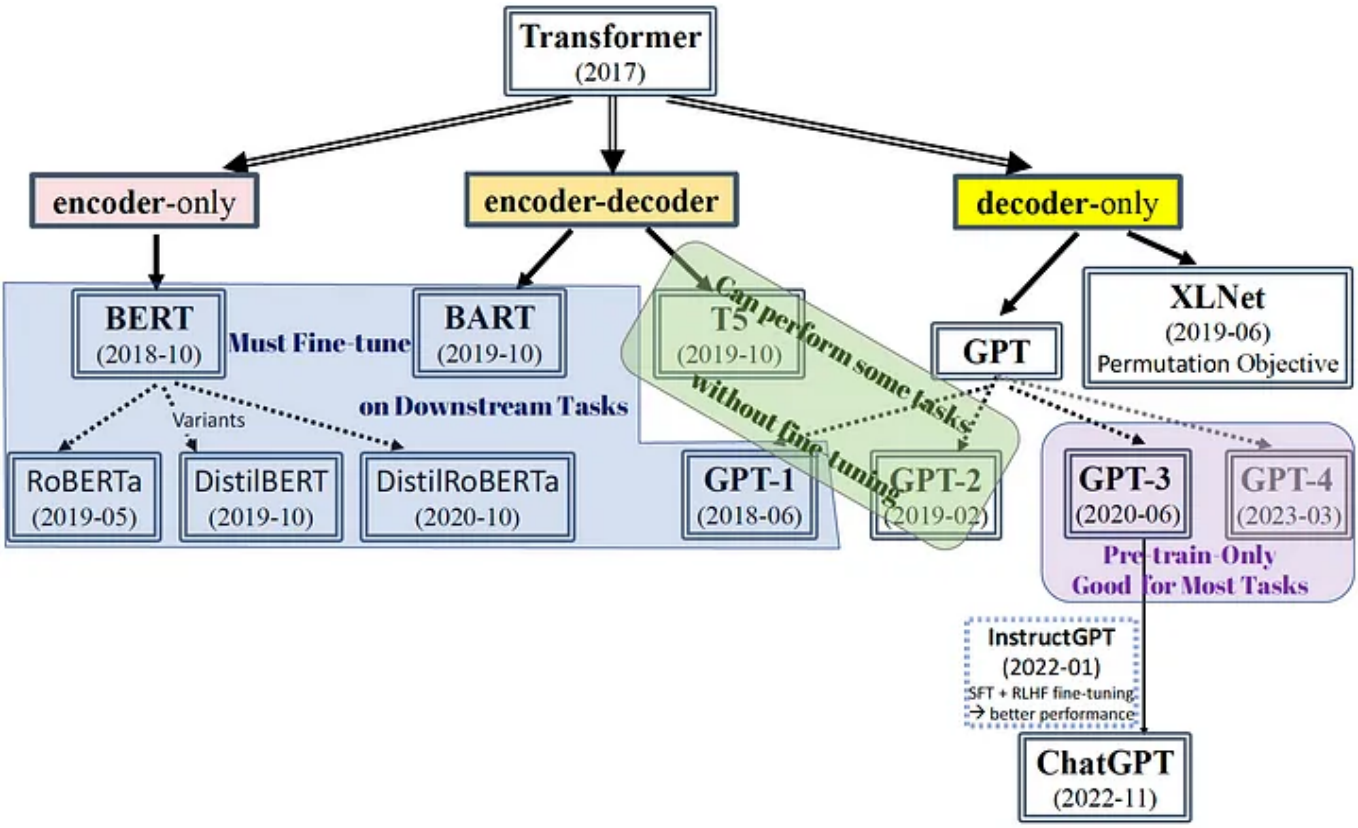


Fig. 1: Transformer-based models graph. The graph illustrates models of different architectures — encoder-only (autoencoding AE), decoder-only (autoregressive AR), and encoder-decoder models, and whether they require fine-tuning on downstream task-specific datasets. (**Image Source: created by author**)

BERT, GPT, T5, BART, and XLNet are members of the Transformer (Vaswani, et al., 2017) family. These models leverage either the Transformer's encoder, decoder, or both for language understanding or text generation. Since 2018, Transformer models have successfully superseded traditional LSTM and

CNN networks in natural language processing tasks. In my blog "Step-by-Step Illustrated Explanations of Transformer", I have provided a clear explanation of how the Transformer operates.

> *ChatGPT is a language model that builds upon the pre-trained GPT-3 and incorporates additional fine-tuning through the InstructGPT method.*

In contrast to **BERT, which employs the encoder**, the **GPT** models (GPT-1 to GPT-4) have mostly remained the same architecture utilizing Transformer **decoder** stacks. The variances in architecture and pre-training objective among language models determine whether a model excels in text generation task or language understanding task. We will examine each Transformer-based model individually. However, first, I will introduce some critical fundamentals such as **architectures, pre-training objectives, tokenizers, and positional encodings**, which vary across models and are responsible for their distinct strengths and weaknesses in solving different natural language tasks.

Recently I gave a seminar presentation that's directly related to the topic discussed in this blog article and the response from the audience (which numbered 70) was very positive. I am delighted to share the YouTube recording video of the presentation, which is currently available in a Chinese-speaking version. *If you're interested in an English version, please let me know*.

The presentation slides can be found here.

OUTLINE

1. Critical Fundamentals

··· 1.1 **Architecture**

····· Encoder-Based

····· Decoder-Based

····· Transformer-XL (*Improved Transformer*)

··· 1.2 **Pre-Training Objective**

····· Pre-training Process (*must*)

····· Fine-tuning on downstream tasks (*must or not? — depends on objective*)

····· Autoencoding (AE) — Denoising Objective

····· Autoregressive (AR) — Seq-to-Seq Objective
····· Permutation Autoregressive Objective (*AR variant, XLNet uses*)

··· 1.3 **Positional Encoding**
····· Absolute Positional Encoding
····· Relative Positional Encoding (*short introduction*)

···1.4 **Tokenizer** (*sub-word level*)
····· WordPiece (*short introduction*)
····· Byte-Pair-Encoding (BPE) (*short introduction*)

2. Models

··· 2.1 **BERT**
····· RoBERTa
····· DistilBERT
····· DistilRoBERTa

··· 2.2 **GPT**
····· GPT-1 (*must fine-tuning*)
····· GPT-2 (*some in-context learning ability)*
····· GPT-3 (*phenomenal in-context learning; ChatGPT = pre-trained GPT-3 + fine-tuned by InstructGPT method*)

··· 2.3 **T5**

··· 2.4 **BART**

··· 2.5 **XLNet**

3. Conclusions

1. Critical Fundamentals

1. 1 ARCHITECTURE

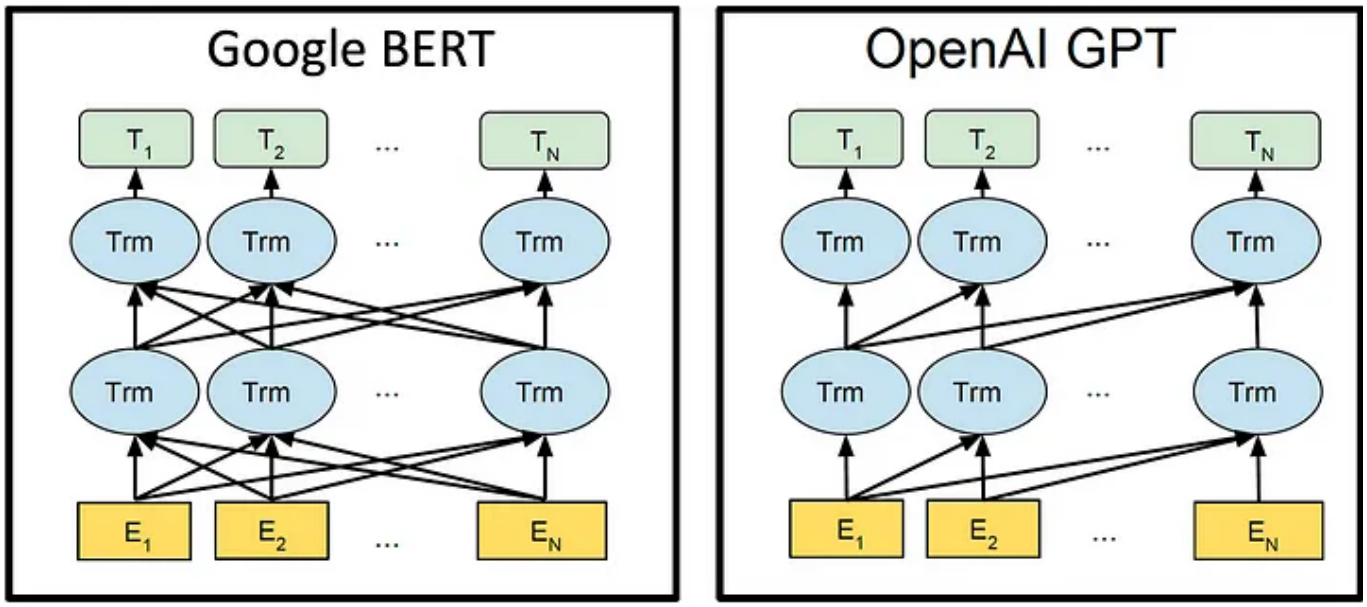


Fig. 2: BERT vs GPT — typical autoencoding (AE), encoder-only, bidirectional model vs typical autoregressive (AR), decoder-only, left-to-right model. Note that this figure is only qualitatively correct. To accurately understand transformer encoder and decoder structures, please read my blog "Step-by-Step Illustrated Explanations of Transformer". (Image Source: Devlin, et. al., 2018)

Transformer was an encoder-decoder neural network with a core component — self-attention mechanism. (*Please refer to my blog "Step-by-Step Illustrated Explanations of Transformer" for an in-depth understanding of the self-attention mechanism.*) The **encoder** performs **parallel processing** on all tokens, allowing for **bidirectional** understanding, while the **autoregressive** output text generation nature of the **decoder** limits it to **left-to-right** processing. The choice to prioritize building upon the **encoder block or the decoder block** in the architecture is closely **linked to the selection of pre-training objectives**, which will be discussed in the Section 1.2.

An improved version of the Transformer model, called **Transformer-XL**, was proposed in 2019 to address the issue of the **extra-long dependency** in context. The **original** Transformer-based models can only learn a fixed-length segment of 512–2048 tokens. When dealing with lengthy texts, learning **multiple segments independently** can lead to the **inability to capture interdependencies** between these segments. Transformer-XL incorporates a **segment-level recurrence mechanism**, which **caches the previous segments** in a hidden state to be reused as an extended context when processing the next segment. XLNet is just built upon the Transformer-XL architecture. Below are examples of models that use encoder-only, decoder-only, or both:

• **Encoder**-only Models: BERT;
• **Decoder**-only Models: GPT, XLNet (Transformer-XL, permutation);
• **Encoder-Decoder** Models: T5, BART.

1.2 PRE-TRAINING OBJECTIVE

A language model typically undergoes a two-step process:

(a). **Pre-training**: This establishes a decisive foundation for the model by training on huge and diverse text datasets (such as Wikipedia, question-answering websites, literature books, etc.) to establish a broad and **upper-level** understanding of natural language patterns in an **unsupervised** manner.

(b). **Fine-tuning**: The pre-trained model is further trained on **lower-level**, more specific **downstream tasks** separately, such as sentiment analysis, text classification, named entity recognition, machine translation, and question-answering, etc. However, the fine-tuning on downstream tasks requires the creation of carefully **prepared datasets** with corresponding **labels** and often

involves **modify**ing the **fine-structure** of the model, which demands significant labor force.

*Please refer to Appendix B1.1 of the BERT paper for information on **downstream tasks**, which can be categorized into four types: Sentence Pair Classification Tasks, Single Sentence Classification Tasks, Question Answering Tasks, and Single Sentence Tagging Tasks.*

However, our **ultimate goal** is to **unify all downstream tasks into one solitary pre-training task**, thus eliminating the need for any subsequent fine-tuning on separate tasks. Google introduced the T5 model in 2019, which **treated all NLP tasks as text-generation** tasks, even for text classification problems. Furthermore, GPT-3 demonstrated **phenomenal in-context learning capability during the pre-training process only** (Fig. 7 & Fig.8) and outperformed other fine-tuned models in certain downstream tasks.

The choice of pre-training objectives can significantly impact the performance of a Transformer-based model and **the degree of adaptation required for fine-tuning on specific tasks**. **Autoregressive (AR)** and **autoencoding (AE)** are currently two successful types of objectives. **AR** models focus on **regenerating text sequences** and are particularly skilled in **text generation** tasks such as machine translation, abstractive summarization and question-answering. **AE** models aim to **reconstruct** the original text from **corrupted text data** and excel in **language understanding**. Below are the details of AR and AE:

Autoregressive (AR): AR models requires the involvement of **decoder** stacks in the generation process. The objective is the maximize the log-likelihood under the forward autoregressive factorization:

$$\max_{\theta} \log p_\theta(x_1, \ldots, x_T) \approx \sum_{t=1}^{T} \log p_\theta(x_t \mid x_1, \ldots, x_{t-1}) = \sum_{t=1}^{T} \log \frac{\exp(y_{x_1,\ldots,x_{t-1}}(x_t))}{\sum_{x'} \exp(y_{x_1,\ldots,x_{t-1}}(x'))}, \quad (1)$$

where θ is the network parameters, and $y_{x1,\ldots,xt-1}$ represent the function of a token x given the previous text sequence x_1, ..., x_{t-1}.
This objective is well-suited for text generation tasks, but it is limited to **left-to-right** generation.

Permutation objective — AR variant**:** However, **XLNet** overcomes this limitation by using **permutations** of the factorization order of the previous tokens sequence in its training objective. The details of **the permutation language modeling objective** will be discussed in Section 2.5 — XLNet.

Autoencoding (AE): AE models utilize a **denoising objective**, in which tokens are randomly masked and the model aims to reconstruct them. The encoder stacks are involved in this process. Let x_m represent the masked tokens at specific positions in a text sequence of length T and x_{-m} represent the corrupted text sequence resulting from these masks. The objective is to predict the original masked tokens by maximizing the log-likelihood

$$\max_{\theta} \log p_\theta(\mathbf{x}_\mathbf{m} \mid \mathbf{x}_{\overline{\mathbf{m}}}) \approx \sum_{t=1}^{T} m_t \log p_\theta(x_t \mid \mathbf{x}_{\overline{\mathbf{m}}}) = \sum_{t=1}^{T} m_t \log \frac{\exp\left(y_{\mathbf{x}_{\overline{\mathbf{m}}}}(x_t)\right)}{\sum_{x'} \exp\left(y_{\mathbf{x}_{\overline{\mathbf{m}}}}(x')\right)}, \quad (2)$$

where m_t is 1 if the t-th token is masked, and 0 otherwise.
The AE objective allows bidirectional processing during pre-training, enabling better context understanding. However, this objective **assumes independence,** meaning that all tokens have an equal probability of being masked and reconstructed, regardless of their interdependence.
Additionally, there is a **discrepancy between the pre-training objective and**

the transfer learning for text generation tasks, which is more similar to the natural process of human communication.

• **Autoencoding (AE)** models: BERT, BART, and T5;
• **Autoregressive (AR)** models: GPT;
• **Permutation AR**: XLNet (*Section 2.5*).

1.3 POSITIONAL ENCODING

Transformer-based models encode positional information by adding **positional embeddings** to token embeddings, ensuring **parallel processing** while preserving text sequence order. **The original Transformer model** used sinusoidal functions of **absolute positions,** with the insight that the relation between two tokens in different positions could be expressed as the cosine similarity of their positional embeddings.

However, **Transformer-XL introduced relative positional encodings** to handle position encodings in different segments and recognized that the distance between each pair of tokens is a more critical factor in self-attention calculations than their absolute positions. R_s represents the relative distance s between two positions and the vector R_s is trainable during self-attention calculations. More information on relative encoding can be found on page 5 of the Transformer-XL paper.

• **Absolute Positional Encoding**: BERT, GPT, BART,
• **Relative Positional Encoding**: T5, XLNet.

1.4 TOKENIZER

A token can be simply referred to as a single word or a sequence of consecutive words (*n*-grams). However, since English words have various inflectional forms (verb tenses, transition between verbs and nouns, plural,

compound words, etc.) , **sub-word based tokenization** is a viable option as it breaks down **a word into sub-word units** to capture its as root, prefix, suffix and other linguistic elements. For instance, “tiresome” can be decomposed into “tire” and “some” , while “tired” can be broken down into “tire” and “d”. In this way, “tiresome” and “tired” can be recognized to have the same derivations.

Fig. 3: An example of the sub-word tokenization process using WordPiece. (Image Source: Google AI blog, song et. al., 2021)

Large language models (LLMs) commonly use sub-word tokenization, with two primary methods being **WordPiece** and **Byte-Pair-Encoding (BPE).** The WordPiece method collects sub-word units by maximizing the likelihood of the vocabulary dataset over all possible n-gram characters. On the other hand, BPE is similar to WordPiece but uses all 256 Unicode characters and thus can include special chracters and eliminate the need for adding a special symbol to represent punctuation. Here is a reference for detailed information of WordPiece and BPE.

- **WordPiece**: BERT, T5;
- **Byte-Pair-Encoding (BPE)**: GPT-2, GPT-3, BART, RoBERTa.

2. MODELS

2.1 BERT (Bidirectional Encoder Representations from Transformers)

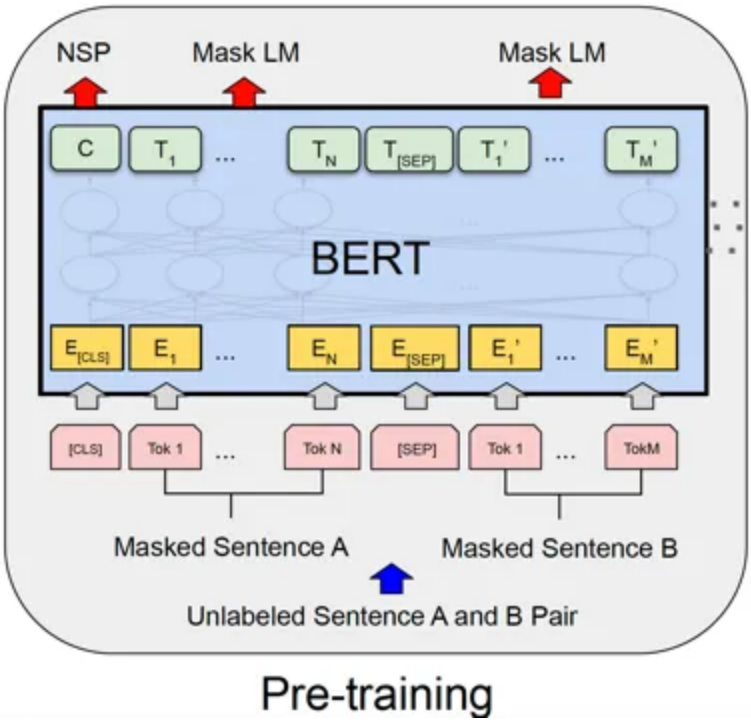


Fig. 4: Denoising pre-training precess of BERT. Only masked tokens (Masked LM) needs to be predicted at the output. Next sentence prediction (NSP) , which involves predicting whether sentence B is the actual next sentence following sentence A, is a binary classification task that is typically performed in conjunction with a denoising objective during pre-training. [CLS] is a special token that represents the start of every input example, and [SEP] is a special separator token. (Image Source: Devlin, et. al., 2018)

Google introduced BERT in 2018 as a **bidirectional** model that exclusively utilizes **encoder stacks.** BERT outperformed GPT-1 (earlier 2018), which processes text data from left-to-right, in various tasks with the same number of parameters, largely due to its bidirectional processing capability.

Pre-training Process:

One of the pre-training objectives of the BERT model is the **denoising** objective, which involves predicting **15% of randomly masked tokens** using Eq. 1.
Notably, among these 15% randomly masked tokens, **80%** of the time they are **replaced with [MASK] tokens**, **10%** of the time they will be **replaced with a random word** for the error-correction learning. The remaining **10%** the time they are **kept unchanged** to maintain the overall context understanding when some crucial words are masked out.

BERT's pre-training includes the **Next Sentence Prediction (NSP)** objective to assess the model's ability to **understand the overall meaning of a sentence** instead of just specific tokens. The task involves a binary classification task to predict whether **the next sentence is the actual consecutive sentence**. 50% of the time the actual consecutive sentence is presented, and 50% of the time a random sentence from the same literature is provided.

Fine-tuning Process:

Real-life language tasks are not denoising tasks, leading to a discrepancy between pre-training and fine-tuning for BERT models. Therefore, fine-tuning is necessary for individual downstream tasks.

BERT categorizes downstream tasks into four types: (a) Sentence Pair Classification Tasks (e.g., semantic similarity between two sentences), (b) Single Sentence Classification Tasks (e.g., sentiment analysis), (c) SQuAD (Question-Answering), and (d) Named Entity Tagging. For a better understanding of the downstream evaluation datasets, please refer to Appendix B1.1 of the BERT paper.

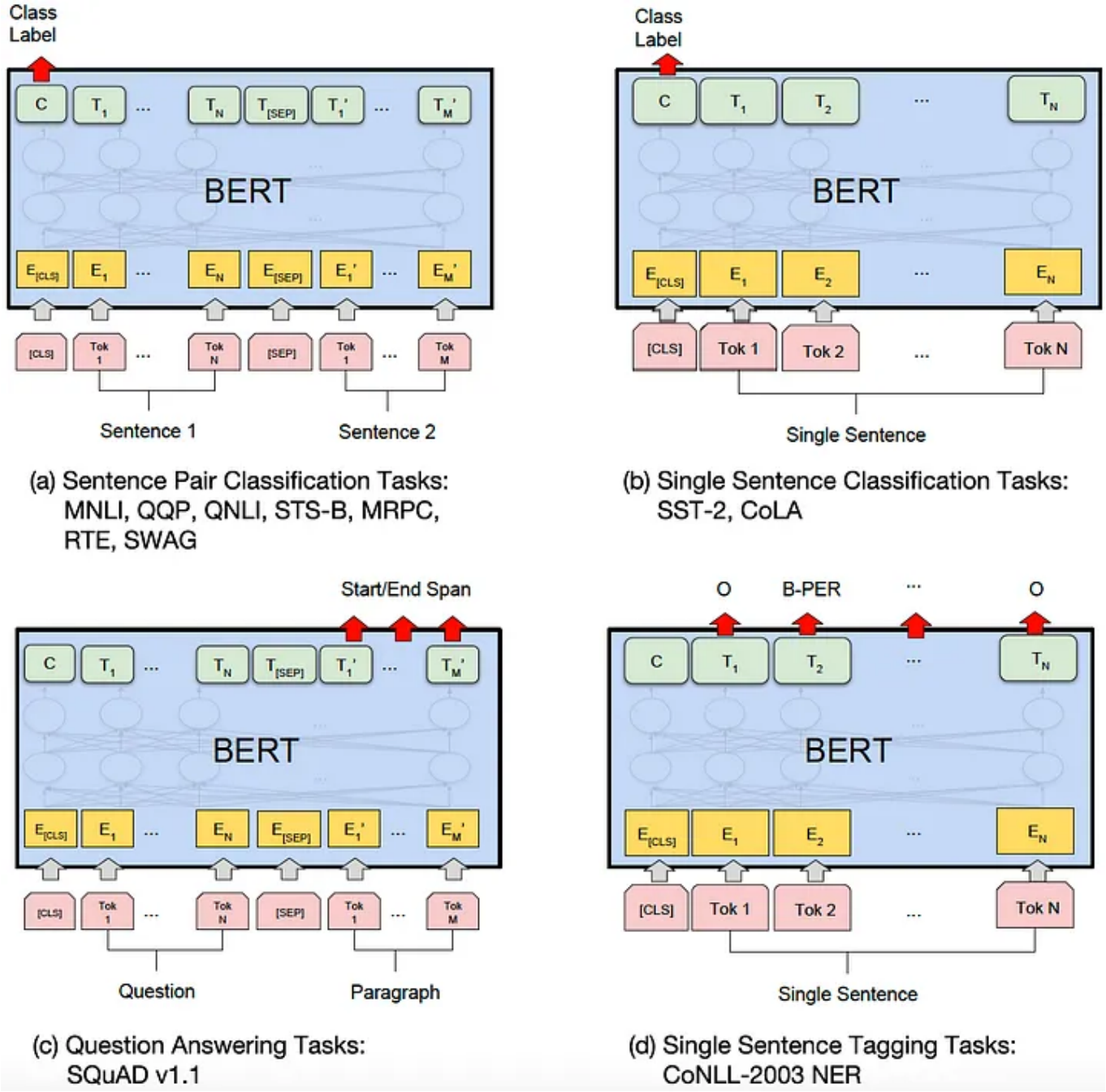


Fig. 5: Fine-tuning procedures on each individual downstream tasks after pre-training. (a) Sentence Pair Classification Tasks (e.g., semantic similarity between two sentences), (b) Single Sentence Classification Tasks (e.g., sentiment analysis), (c) SQuAD (Question-Answering), and (d) Named Entity Tagging. (Image Source: Devlin, et al., 2018)

.....

BERT has several variants — RoBERTa, DistilBERT, and DistillRoberta. Here are brief overviews of each one.

2.11 RoBERTa (Robustly Optimized **BERT** Pretraining Approach)

RoBERTa is similar to BERT with some modifications:

1). BERT's **Nest sentence prediction (NSP) is removed** from RoBERTa's pre-training objective;
2). In RoBERTa, the randomization masking of **15% of tokens are changing for each pre-training epoch**, as opposed to remaining static throughout all training epochs as in BERT;
3). RoBERTa uses the **BPE tokenization** method instead of BERT's WordPiece;
4). **More extensive training data** with **lengthier sequence segments** are trained.

2.12 DistilBERT (**Distil**led version of **BERT**):

DistilBERT utilizes a smaller, lightweight Transformer network as a **student model** to mimic the soft labels or probability distributions of **BERT, a larger teacher model**, rather than predicting the actual labels. This approach results in a 40% reduction in the size of the BERT model, with half of the encoder stacks being reduced in DistilBERT. Despite the smaller size, DistilBERT **retains 97% of the language understanding capabilities** of BERT, and **inference time is improved by 60%.**

2.13 DistilRoBERTa (**Distil**led version of **RoBERTa**)

Like DistilBERT, the fundamental concept behind this approach is to train a smaller and more lightweight student model to emulate the behavior of a larger teacher model. However, in this case, the teacher model is RoBERTa.

2.2 GPT (Generative Pre-Training language model)

OpenAI proposed the GPT-1–4 models that only used decoder stacks, making them left-to-right autoregressive models. Although they do not have the same bidirectional understanding as BERT, their generative approach aligns well with downstream tasks, as all natural language tasks can be treated as generative tasks.

GPT-1 (2018, 117 million parameters) did not exhibit emergent capabilities and heavily relied on fine-tuning for individual downstream tasks. The figure below illustrates how downstream tasks are categorized into four categories and fine-tuned accordingly.

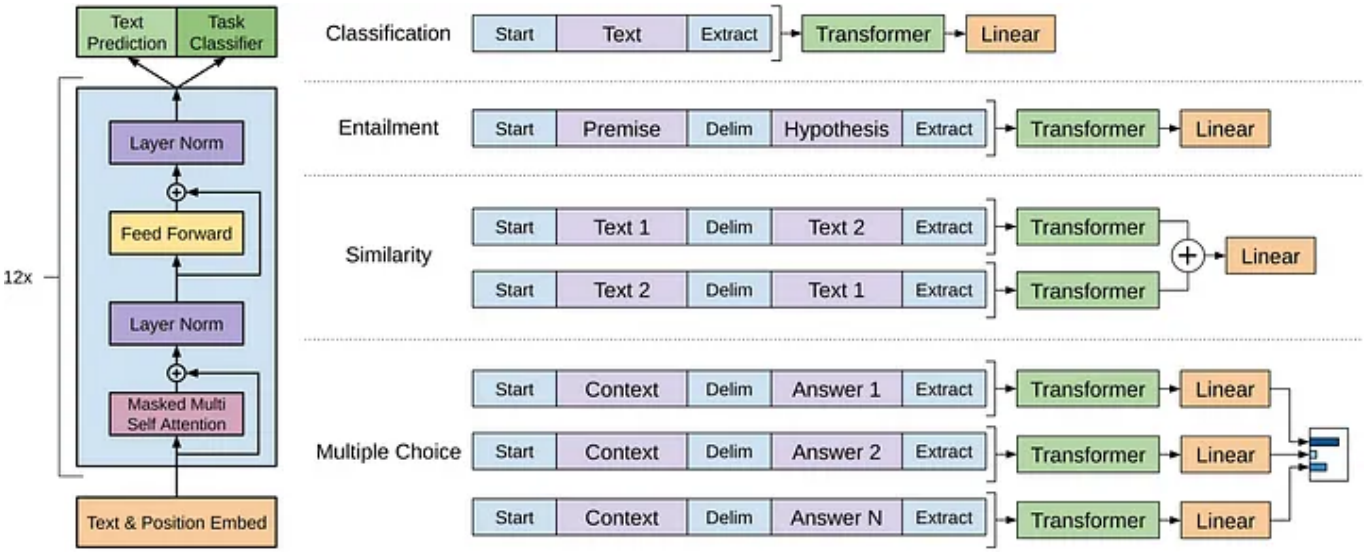


Fig. 6: Decoder-only architecture for GPT models (left) and finetuning procedure for four different categories of downstream tasks for GPT-1 and GPT-2 (right). (Image Source: Radford et. al., 2018)

GPT-2 (2019, 6 billion parameters) introduced the phenomenon of in-context learning for a few tasks, and improved its tokenizer by using Byte-level Encoding (BLE) on top of the original spacy tokenizer used in GPT-1.

GPT-3 (2020, 175 billion parameters) has surprisingly demonstrated strong **in-context learning capabilities**, including zero-shot and few-shot learning abilities as depicted in Figs.7 and 8. In few-shot or zero-shot settings without

further fine-tuning, GPT-3 can achieve comparable performance with other fine-tuned state-of-the-art (SOTA) models. To improve its performance on multi-tasks, the **InstructGPT** method was used for fine-tuning, which combines supervised learning of demonstration texts from labelers, then with reinforcement learning of generation text scoring and ranking, which are referred to as **Reinforcement Learning from Human Feedback (RLHF).** This approach allows for the sharing of network parameters across all downstream tasks and prompts, rather than having to fine-tune for each individual downstream task. ChatGPT is a pre-trained GPT-3 model that has been fine-tuned using the InstructGPT method.

This week, Long-awaited GPT-4 (2023) was finally revealed! As a multi-modal model, it can decompose a photo's basic elements and provide a complex understanding of its context. A more comprehensive understanding of the GPT-4 blog will be shared in the future.

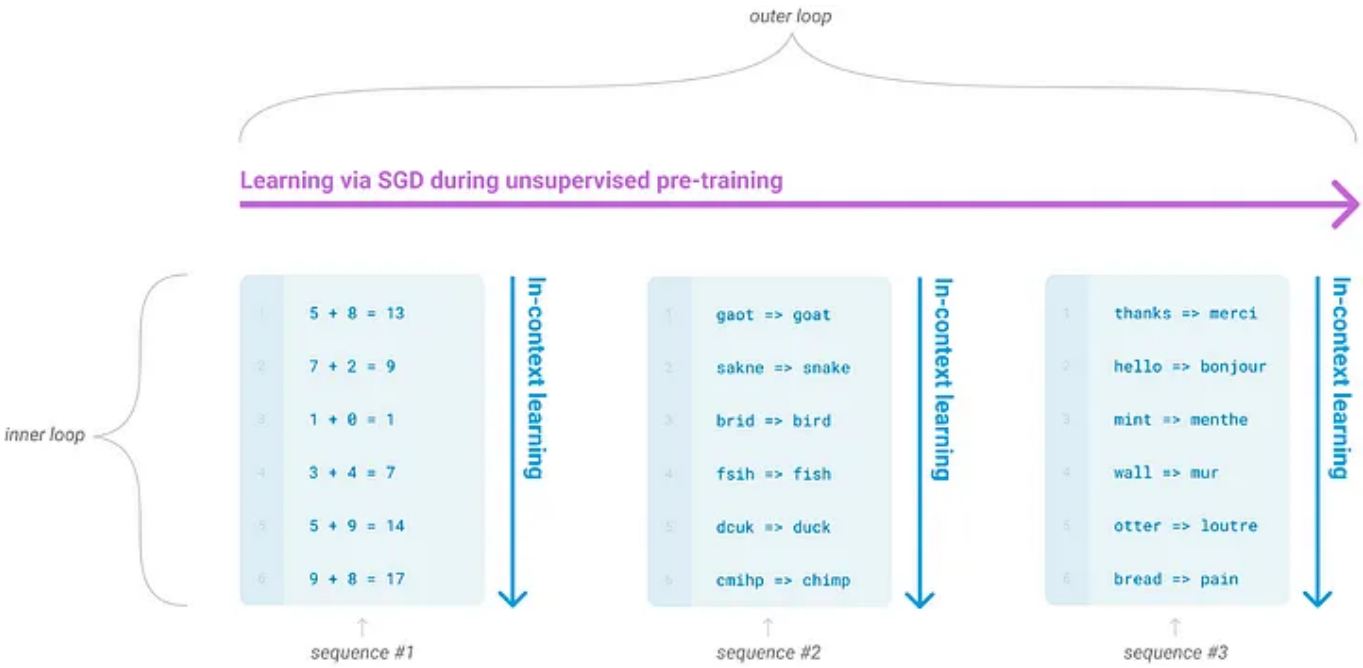


Fig. 7: GPT-3's In-context learning with unsupervised pre-training only. Without further fine-tuning on specific tasks, GPT-3 learns (left) simple arithemetic operations; (middle) word typo auto-correction; (right) multi-language word translation. The patterns are learnt from the pre-training context dataset only. (Image Source: Brown et. al., 2020)

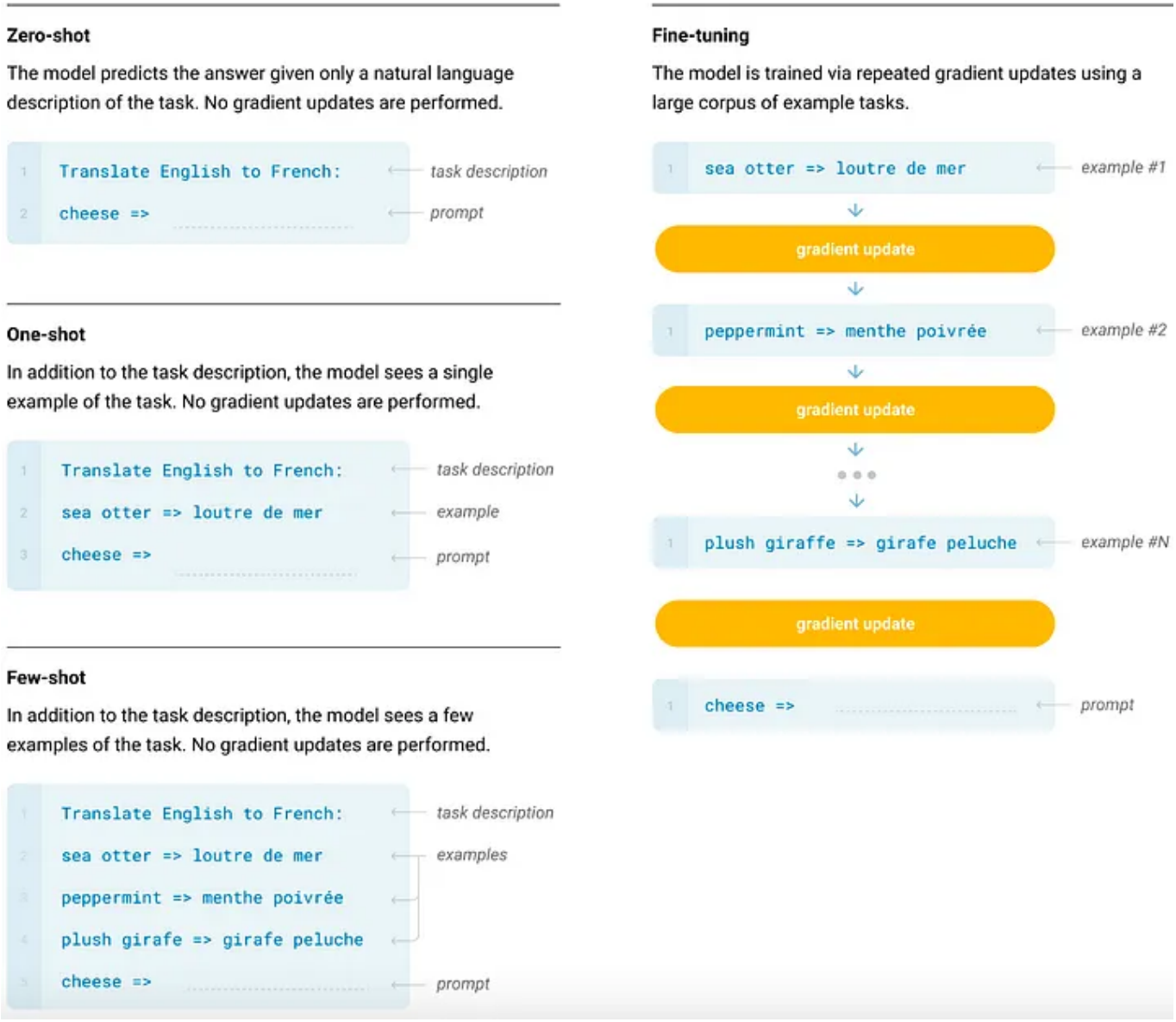


Fig. 8: In-context learning abilities from pre-training process (zero-shot learning and few-shot learning) vs Fine-tuning. Left: In-context learning abilities do not involve network weights updates. Zero-shot learning: the model can infer a pattern in a task without any examples as task descriptions. One-shot or few-shot learning: with one or few task examples to explain the task, the model can deduce the correct form of answer. Right: Solving a task by fine-tuning, in the other hand, requires preparing a dataset specific to the task and training it by updating network weights. (Image Source: Brown et. al., 2020)

2.3 BART (Bidirectional and Auto-Regressive Transformers):

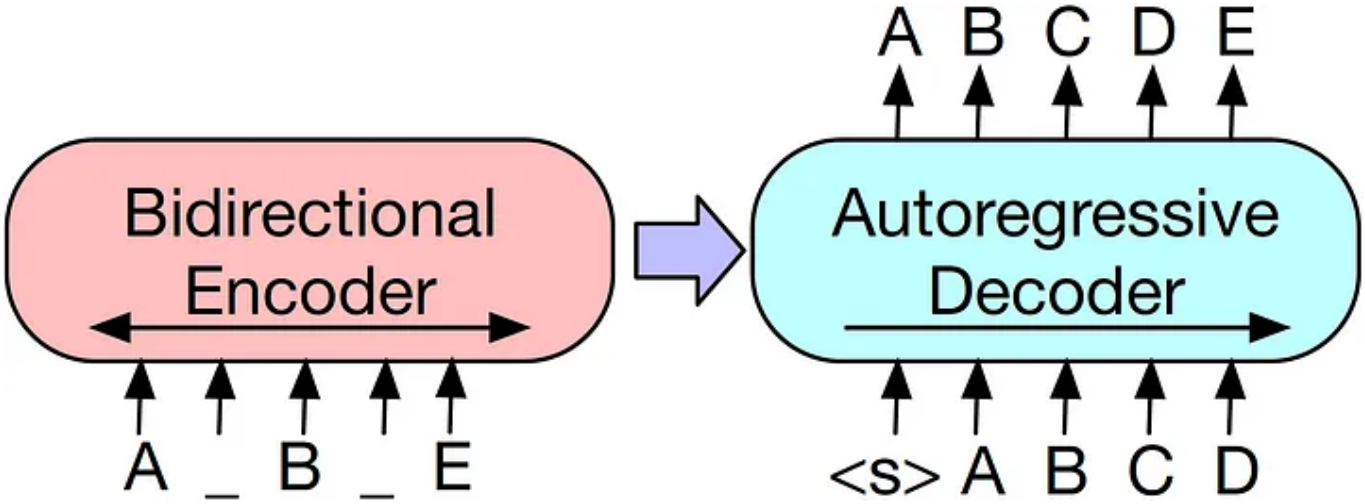


Fig. 9: Encoder-Decoder architecture of BART. In the pre-training process, the encoder component is responsible for learning to fill in missing text in the corrupted training data (denoising), later the decoder component works to autoregressive reconstruct of the text. The pre-training process for this model follows a similar approach to T5. (Image source: Lewis et. al., 2019)

BART (2019) proposed by Facebook greatly resembles T5 in terms of the **denoising pre-training objective** and the **encoder-decoder** architecture, with the only difference being that **30%** of the tokens are masked and **sentence permutation** is used in the pre-training text. To gain an understanding of BART, please refer to how T5 works in Section 2.4.

2.4 T5 (Text-To-Text Transfer Transformer)

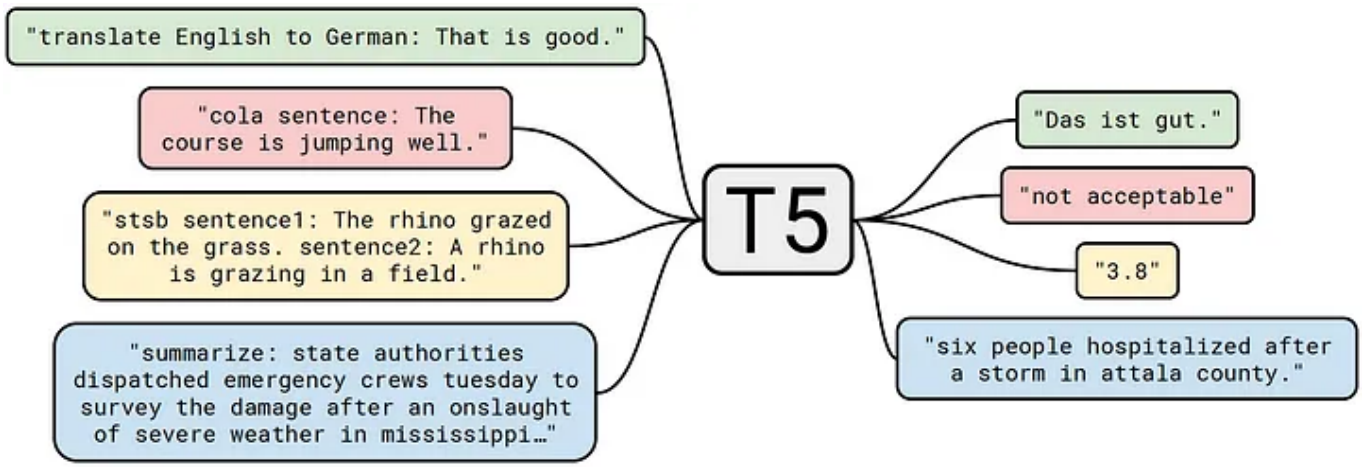


Fig. 10: T5 integrates all downstream tasks by considering them as a text generation problem and utilizing prefixes as guidelines. The "cola" task involves **binary classification** for sentence acceptability judgment, but T5 model outputs **text generation, instead of 0 or 1**. "stsb" is to calculate semantic text similarity between two sentences. || P.S. During the T5 pre-training process, the C4 dataset (Colossal Clean Crawled Corpus) is utilized, which has been thoroughly prepared with modifications made to the original text, including the

incorporation of prefixes like “Translate the sentence:” to aid in identifying the prompt’s intent. (Image Source: Raffel et. al., 2019.)

In 2020, Google proposed T5 as a unified model capable of transforming all downstream tasks into text generative tasks, even classification problems.

T5 uses an **encoder-decoder architecture and a denoising objective**, after experimenting with several unsupervised pre-training objectives and architectures. During **pre-training, 15% of the tokens** fed into the encoder are **randomly masked. But it is** a modified version of BERT’s masking and **denoising** algorithm: **consecutive masked tokens are replaced by**, a sentinel and are treated as **a new single token** added to the original vocabulary. This helps the model learn to predict how many words are missing in the blank. Later, **generative capability** is learned by randomly splitting the input text in the dataset, with the first part fed into the encoder and the second part treated as the output to be **auto-regressively regenerated.**

2.5 XLNet (Transformer-XL generative autoregressive model)

XLNet (2019) was introduced by Carnegie Mellon University and Google, incorporating an pre-training objective developed upon the classical **autoregressive (AR) objective.** The new objective maximizes the likelihood of expectation of **all possible permutations of a sequence.** By this, XLNet **overcomes several limitations**, including the **pretrain-finetune discrepancy** caused by BERT’s denoising objective and the **left-to-right context learning limitation** of the classical AR objective. XLNet is based on Transformer-XL, which includes a **segment recurrence mechanism** to process extra-long context and uses **relative positional encoding**. The objective is

$$\max_{\theta} \mathbb{E}_{\mathbf{z}\sim Z_T} \left[\sum_{t=1}^{T} \log p_\theta(x_{z_t} \mid x_{\mathbf{z}_{<t}}) \right], \tag{3}$$

where z denotes a **permutation** in the set Z_T, which contains all possible permutations of the text sequence x of length T. The t-th token at permutation sequence z is denoted by x_z_t, and the tokens sequence preceding the t-th position are denoted by $x_z<t$. It is worth noting that the **permutation** operation **does not disrupt the original order** of the text sequence since the positional encoding retains the sequence order information.

3. CONCLUSIONS

This article extensively covers Transformer-based models such as BERT, GPT, T5, BART, and XLNet. It focuses primarily on encoder or decoder-based architectures and pre-training objectives. The article reveals that autoencoder (AE) models excel in bi-directional context understanding. However, autoregressive (AR) models have an advantage in that there is less discrepancy between pre-training and fine-tuning downstream tasks. Additionally, when the model is scaled to a certain point, pre-trained models already exhibit comparable performance with state-of-the-art models for some tasks, without the need for further fine-tuning. Surprisingly, GPT-3 has demonstrated in-context learning abilities. It appears that AR models are the future, but XLNet attempts to combine the bidirectional context learning of AE models by proposing the permutation language modeling objective.

Thank you for taking the time to read this article. If you found it helpful, **please upvote**, as it took me 10 days to complete. :D

Also check out my last blog "Step-by-Step Illustrated Explanations of Transformer" !

REFERENCES

- Devlin, et al., 2018. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”
- Radford et. al., 2018. “Improving Language Understanding by Generative Pre-Training”
- Radford et. al., 2019. “Language Models are Unsupervised Multitask Learners”
- Brown et. al., 2020. “Language Models are Few-Shot Learners”
- Lewis et. al., 2019. “BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension”
- Raffel et. al., 2019. “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer”
- Dai et. al., 2019. “Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context”
- Sanh et. al., 2019. “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter”
- Liu et. al., 2019. “RoBERTa: A Robustly Optimized BERT Pretraining Approach”
- Google AI blog, Song et. al., 2021. “A Fast WordPiece Tokenization System”
- https://huggingface.co/docs/transformers/tokenizer_summary

Biography

Yule Wang, Physics PhD, NLP Machine Learning Engineer

My LinkedIn: https://www.linkedin.com/in/yule-wang-ml/

My YouTube Channel

Yule Wang

My name is Yule Wang. I achieved a PhD in physics and now I am a machine learning engineer. This is my personal blog...

www.youtube.com

Other YT Videos:

ChatGPT's reinforcement model — InstructGPT

Word-Embeddings: GloVe, CBOW, skip-gram

Transformer Based T5 Model

NLP Bert Gpt Xlnet ChatGPT

More from the list: "NLP"

Curated by Himanshu Birla

Jon Gi... in Towards Data ...

Characteristics of Word Embeddings

· 11 min read · Sep 4, 2021

Jon Gi... in Towards Data ...

The Word2vec Hyperparameters

· 6 min read · Sep 3, 2021

Jon Gi... in

The Word2ve

· 15 min rea

View list

Written by Yule Wang, PhD

Follow

355 Followers

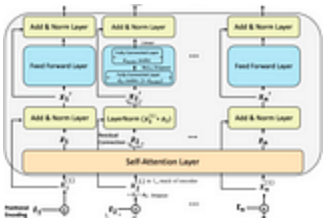
Don't just add the story to your list. Read it :p MLE, Physics PhD. YT channel: youtube.com/@yulewang_machinelearning LinkedIn: linkedin.com/in/yule-wang-ml/

More from Yule Wang, PhD

Yule Wang, PhD

Step-by-Step Illustrated Explanations of Transformer

with detailed explained Attention Mechanism

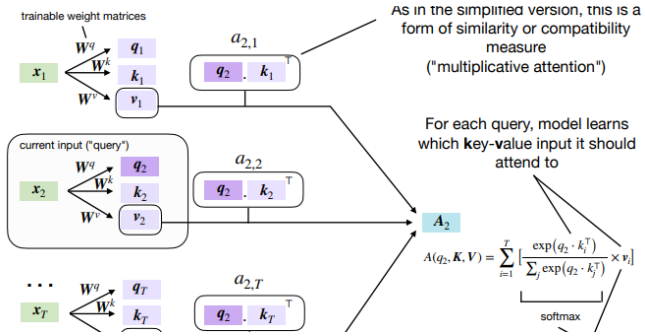


8 min read · Feb 27

260 3

See all from Yule Wang, PhD

Recommended from Medium



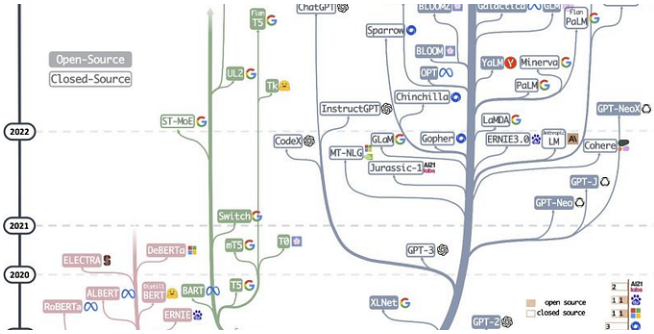
Zain ul Abideen

Attention Is All You Need: The Core Idea of the Transformer

An overview of the Transformer model and its key components.

6 min read · Jun 26

144



Haifeng Li

A Tutorial on LLM

Generative artificial intelligence (GenAI), especially ChatGPT, captures everyone's...

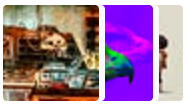
15 min read · Sep 14

372

Lists

The New Chatbots: ChatGPT, Bard, and Beyond

13 stories · 133 saves

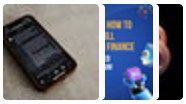


What is ChatGPT?

9 stories · 182 saves

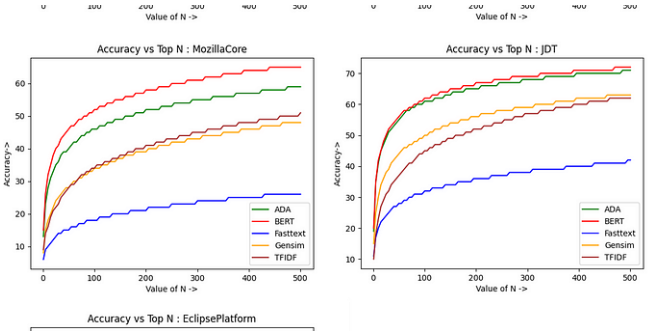
ChatGPT

21 stories · 179 saves



ChatGPT prompts

24 stories · 459 saves



Avinash Patil

Embeddings: BERT better than ChatGPT4?

In this study, we compared the effectiveness of semantic textual similarity methods for...

4 min read · Sep 19

3 1

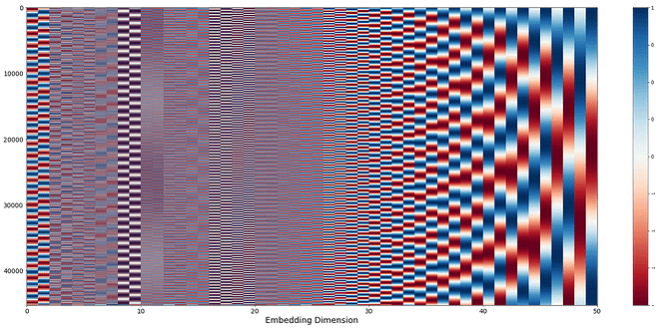
David Shapiro

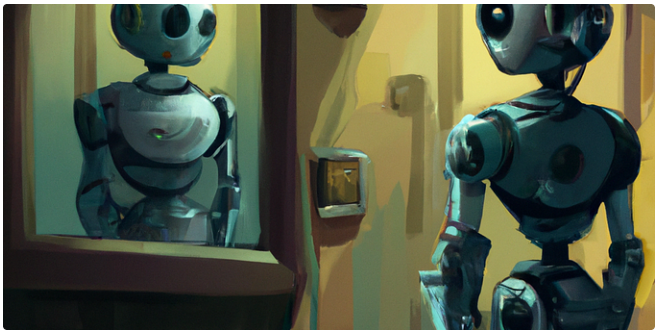
A Pro's Guide to Finetuning LLMs

Large language models (LLMs) like GPT-3 and Llama have shown immense promise for...

12 min read · Sep 23

283 6





Eugene Ku

Transformer Architecture (Part 1—Positional Encoding)

Thomas van Dongen in Towards Data Science

Demystifying efficient self-attention

Nowadays, arguably the most popular and influential model behind the hypes of deep...

4 min read · Aug 22

15

A practical overview

20 min read · Nov 7, 2022

477 2

See more recommendations