This is CS5045

GCR : ioc7cdl

References

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

https://medium.com/nerd-for-tech/gpt3-and-chat-gpt-detailed-architecture-study-deep-nlp-horse-db3af9de8a5d

https://web.stanford.edu/class/cs224n/slides/cs224n-2023-lecture11-prompting-rlhf.pdf

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**[*,†]

[*]equal technical contribution, [†]equal advising

Google Research, Brain Team

{adosovitskiy, neilhoulsby}@google.com

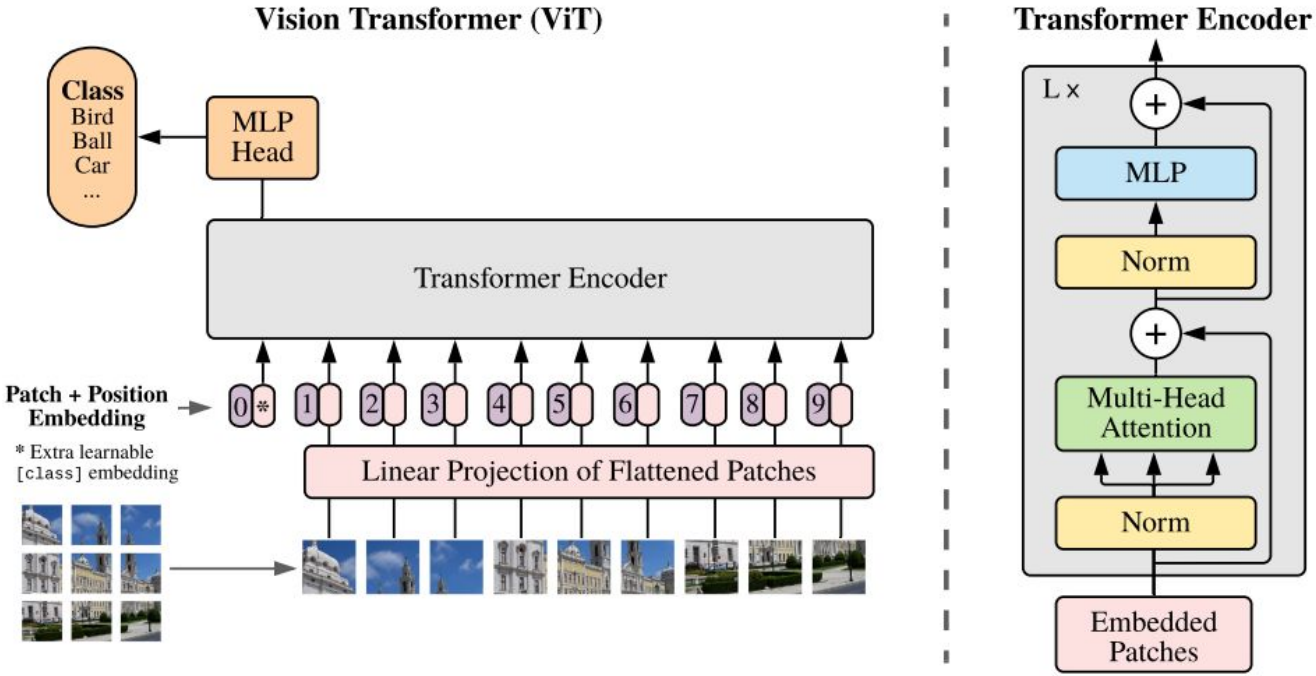


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification token" to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

If My Brother Was Half My Age When I Was 8, How Old Will He Be When I Am 50?

John
(Programmer)

If your brother was half your age when you were 8, that means he was 4 years old at the time. Since you are asking how old he will be when you are 50, you can use the information that he is currently 4 years old to calculate his age at that time. If he is currently 4 years old and you will be 50 years old, he will be 46 years old when you are 50.

GPT - 2

The GPT-2 was trained on a massive 40GB dataset called WebText that the OpenAI researchers crawled from the internet as part of the research effort.

The smallest variant of the trained GPT-2, takes up 500MBs of storage to store all of its parameters.

The largest GPT-2 variant is 13 times the size so it could take up more than 6.5 GBs of storage space.

Improving Language Understanding by Generative Pre-Training

Improving Language Understanding by Generative Pre-Training

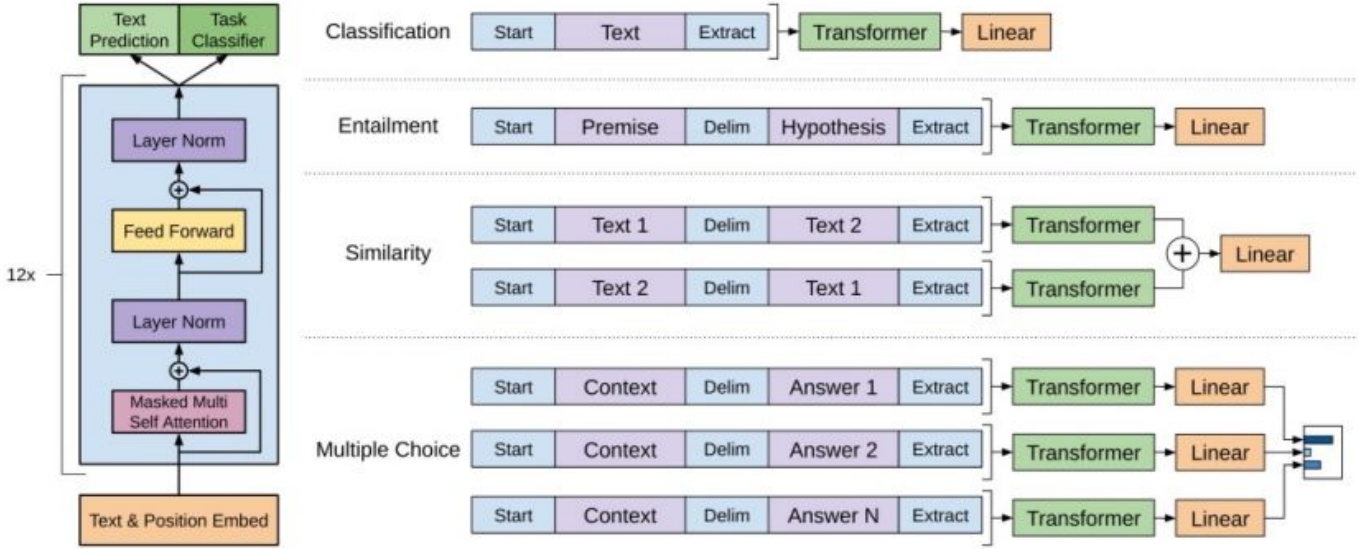
Alec Radford OpenAI alec@openai.com
Karthik Narasimhan OpenAI karthikn@openai.com
Tim Salimans OpenAI tim@openai.com
Ilya Sutskever OpenAI ilyasu@openai.com

Abstract

Natural language understanding comprises a wide range of diverse tasks such as textual entailment, question answering, semantic similarity assessment, and document classification. Although large unlabeled text corpora are abundant, labeled data for learning these specific tasks is scarce, making it challenging for discriminatively trained models to perform adequately. We demonstrate that large gains on these tasks can be realized by *generative pre-training* of a language model on a diverse corpus of unlabeled text, followed by *discriminative fine-tuning* on each specific task. In contrast to previous approaches, we make use of task-aware input transformations during fine-tuning to achieve effective transfer while requiring minimal changes to the model architecture. We demonstrate the effectiveness of

Generative Pretrained Transformer (GPT) [Radford et al., 2018]

How do we format inputs to our decoder for **finetuning tasks?**



The linear classifier is applied to the representation of the [EXTRACT] token.

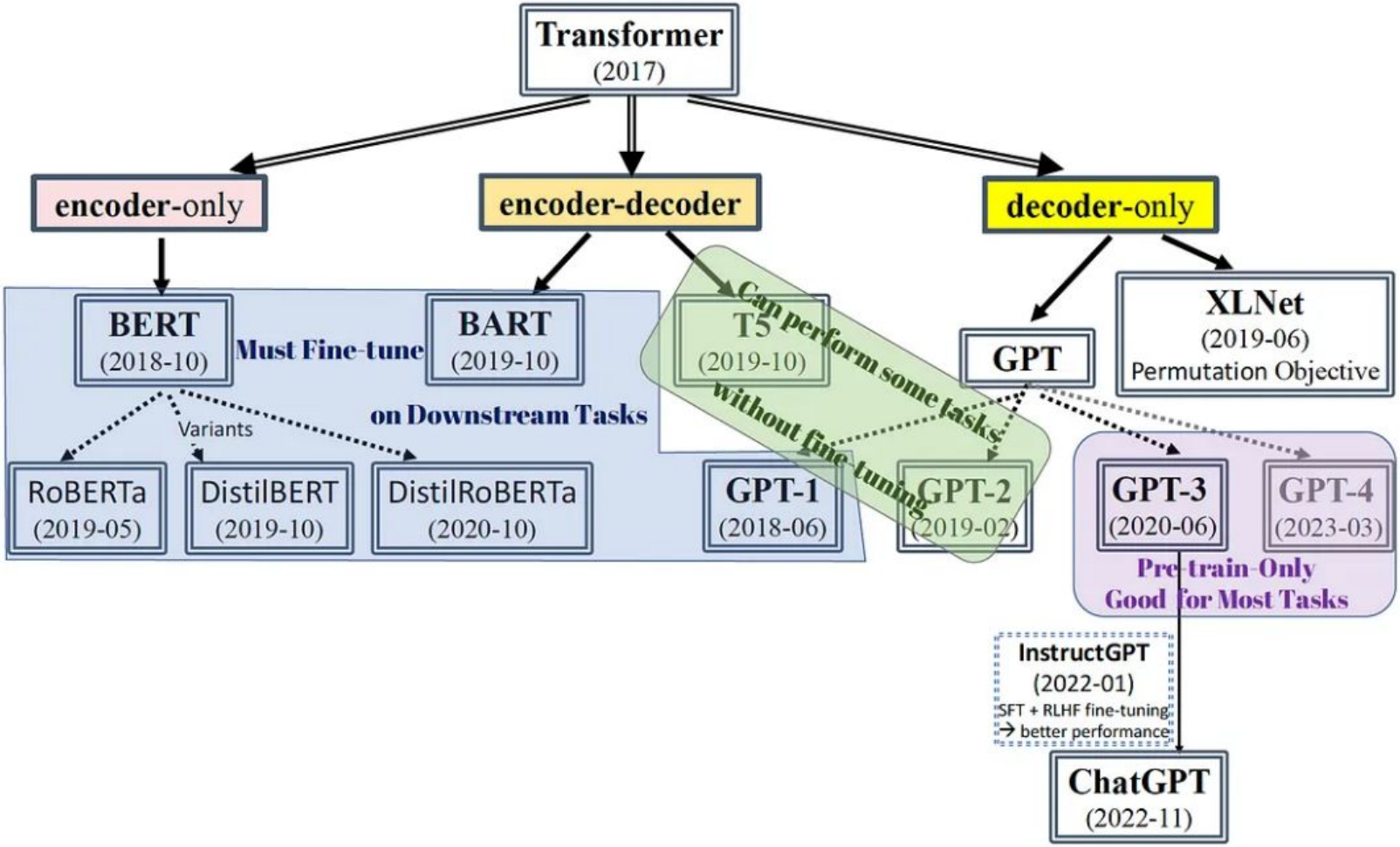
PRE-TRAINING OBJECTIVE

(a). Pre-training: 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 modifying the fine-structure of the model, which demands significant labor force.

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 and outperformed other fine-tuned models in certain downstream tasks.

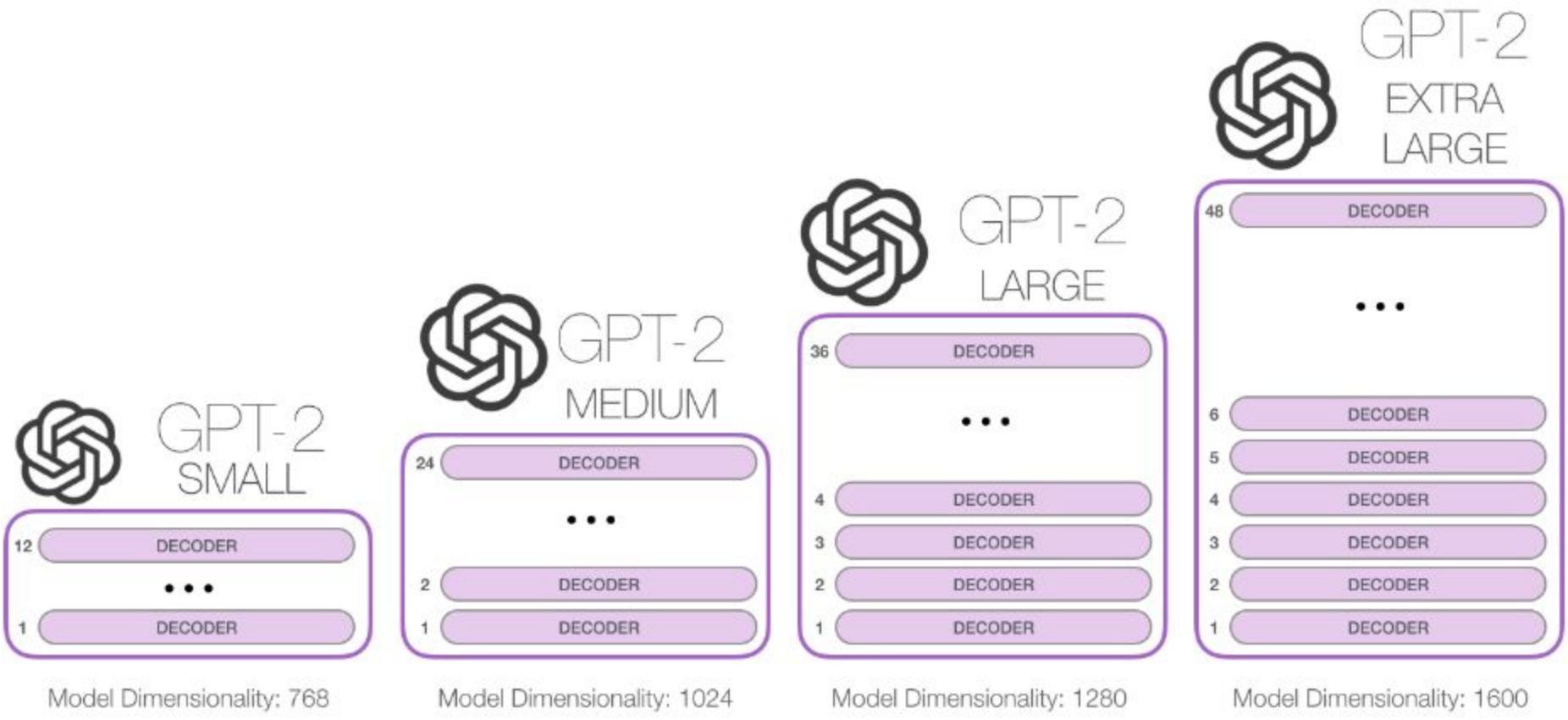

Transformer
(2017)
encoder-only
encoder-decoder
decoder-only
BERT
(2018-10)
Must Fine-tune
on Downstream Tasks
BART
(2019-10)
T5
(2019-10)
Can perform some tasks without fine-tuning
GPT
XLNet
(2019-06)
Permutation Objective
Variants
RoBERTa
(2019-05)
DistilBERT
(2019-10)
DistilRoBERTa
(2020-10)
GPT-1
(2018-06)
GPT-2
(2019-02)
GPT-3
(2020-06)
GPT-4
(2023-03)
Pre-train-Only
Good for Most Tasks
InstructGPT
(2022-01)
SFT + RLHF fine-tuning
→ better performance
ChatGPT
(2022-11)

117M Parameters

345M Parameters

762M Parameters

1,542M Parameters

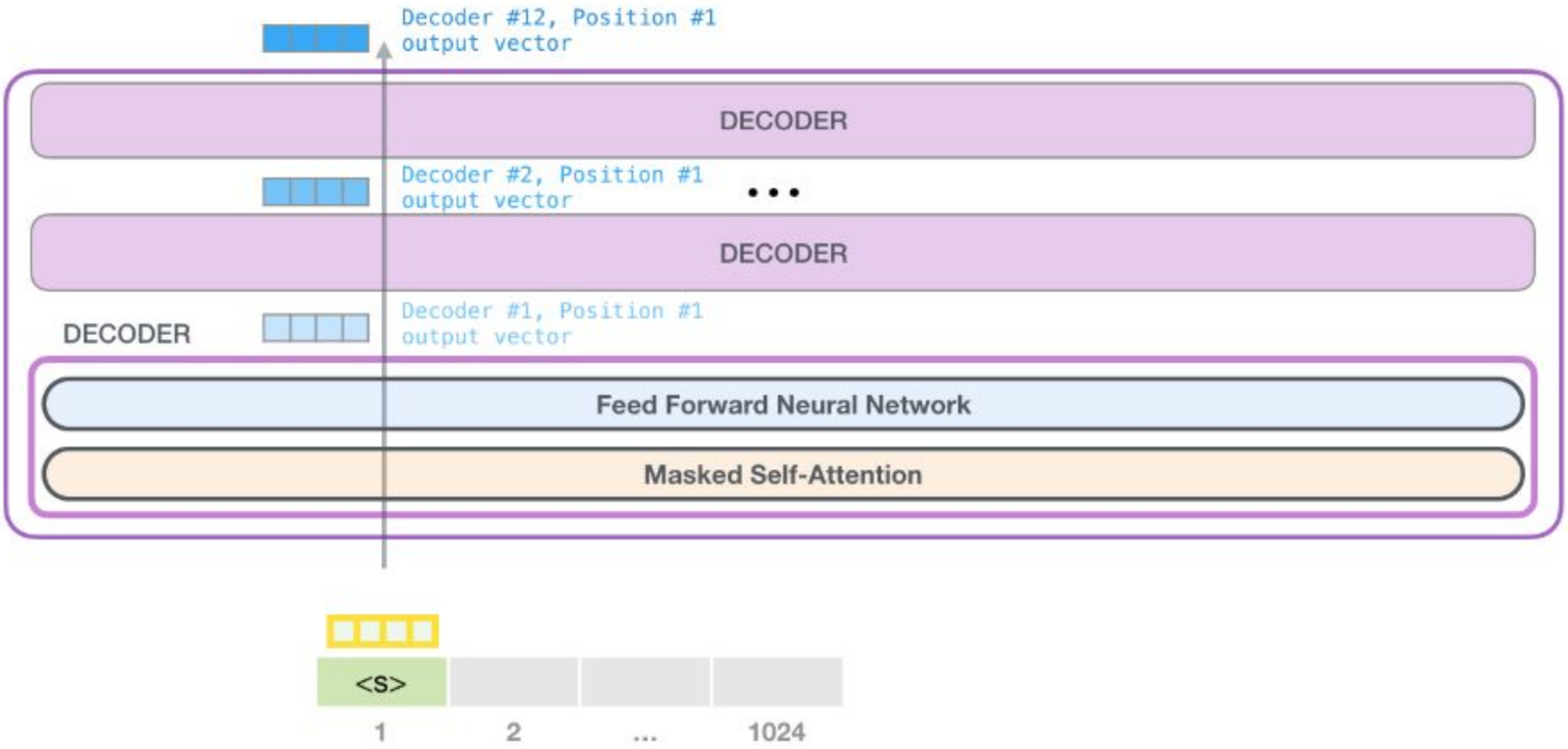

GPT-2
SMALL
12
DECODER
1
DECODER
Model Dimensionality: 768
GPT-2
MEDIUM
24
DECODER
2
DECODER
1
DECODER
Model Dimensionality: 1024
GPT-2
LARGE
36
DECODER
4
DECODER
3
DECODER
2
DECODER
1
DECODER
Model Dimensionality: 1280
GPT-2
EXTRA
LARGE
48
DECODER
6
DECODER
5
DECODER
4
DECODER
3
DECODER
2
DECODER
1
DECODER
Model Dimensionality: 1600

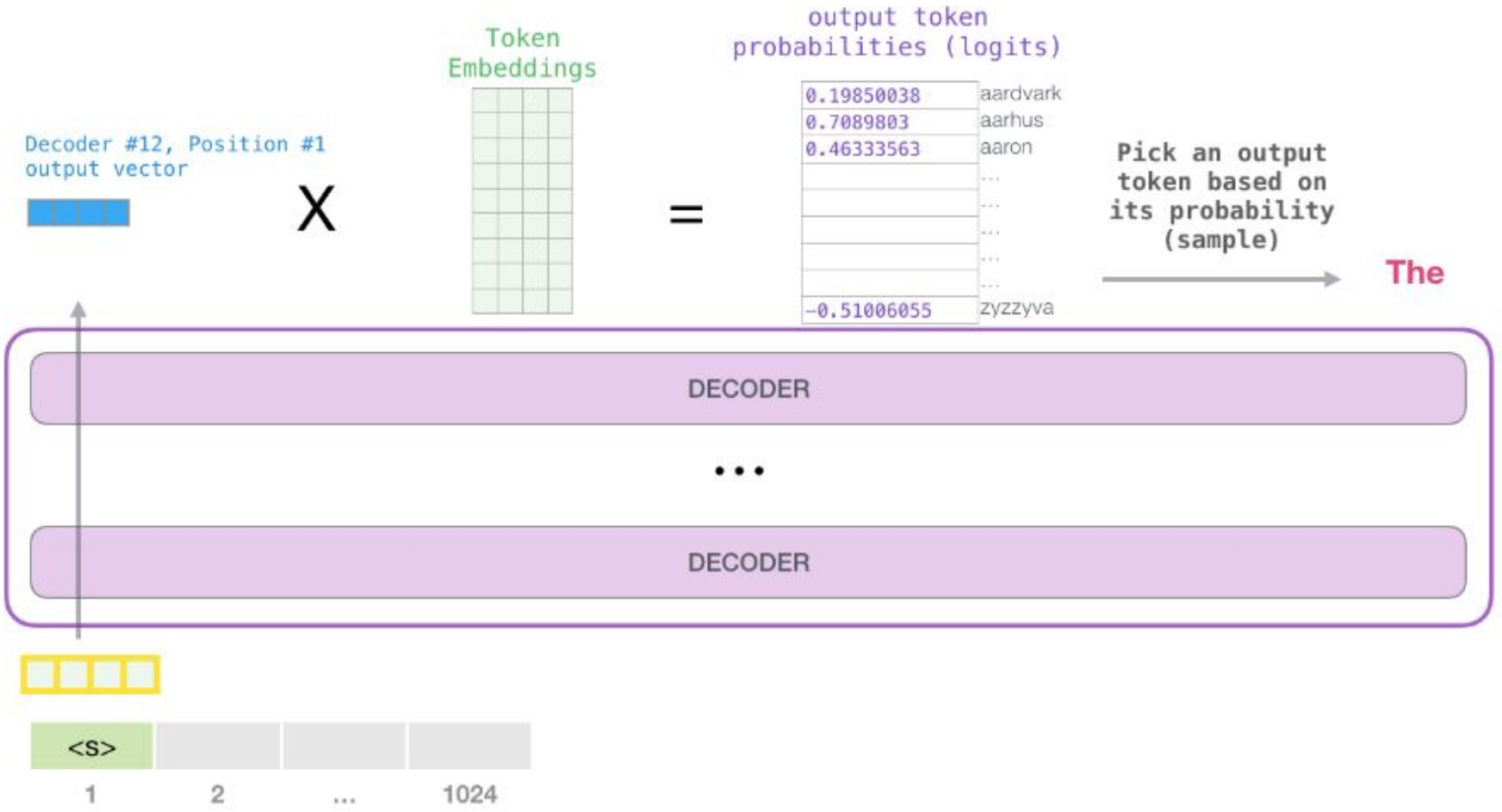
Generative Pretrained Transformer (GPT) [Radford et al., 2018]

2018's GPT was a big success in pretraining a decoder!

- Transformer decoder with 12 layers.
- 768-dimensional hidden states, 3072-dimensional feed-forward hidden layers.
- Byte-pair encoding with 40,000 merges
- Trained on BooksCorpus: over 7000 unique books.
  - Contains long spans of contiguous text, for learning long-distance dependencies.


DECODER
DECODER
Token Embeddings
Positional Encodings
=
Positional encoding for token #1
+
Token embedding of <s>
<s>
1
2
...
1024


Decoder #12, Position #1
output vector
DECODER
Decoder #2, Position #1
output vector
DECODER
DECODER
Decoder #1, Position #1
output vector
Feed Forward Neural Network
Masked Self-Attention
<s>
1
2
...
1024


Token Embeddings
output token probabilities (logits)
Decoder #12, Position #1 output vector
X
=
0.19850038 aardvark
0.7089803 aarhus
0.46333563 aaron
-0.51006055 zyzzyva
Pick an output token based on its probability (sample)
The
DECODER
DECODER
<s>
1
2
...
1024

GPT

GPT-1 (2018, 117 million parameters) did not exhibit emergent capabilities and heavily relied on fine-tuning for individual downstream tasks.

GPT-2 (2019, 1.5 billion parameters) introduced the phenomenon of in-context learning for a few tasks.

GPT-3 (2020, 175 billion parameters) has surprisingly demonstrated strong in-context learning capabilities, including zero-shot and few-shot learning abilities.

ChatGPT

It is a variant of the popular GPT-3 (Generative Pertained Transformer 3) model.

Chat GPT was modified and improved using both supervised and reinforcement learning methods, with the assistance of human trainer (RLHF).Chat GPT also has 176 billion parameters same as GPT -3 model. The learning includes 3 Steps.

- Supervised fine tuning of GPT 3.5 Model
- Reward Model
- Proximal Policy Optimization (PPO)

ChatGPT

ChatGPT is a pre-trained GPT-3 model that has been fine-tuned using the InstructGPT method.

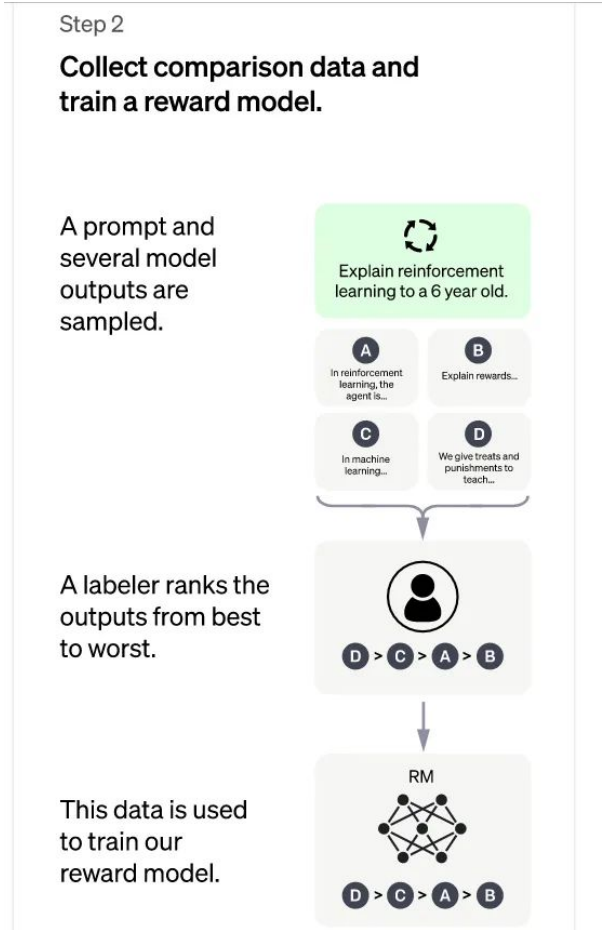
InstructGPT method was used for fine-tuning, which combines supervised learning of demonstration texts from labelers, then with reinforcement learning of generation text scoring and ranking, which are referred to as Reinforcement Learning from Human Feedback (RLHF).

Supervised fine tuning (Step1)

In first Step a pretrained GPT-3 model is used and it will be fine tuned with the help of labelers by creating a supervised dataset. Input Queries were collected from the actual user entries and model generated different responses with respect to that input prompts. The labelers then wrote an appropriate response to the input prompt's (how they want to see that prompt to be answered).The GPT-3 model was then fine-tuned using this new supervised dataset, to create GPT-3.5 model.

Reward Model (Step 2)

In this step SFT model is used and different input/prompts queries fed to the finetuned model and different responses were generated (4 to 7)for every input/prompt. Then labeler determines a reward for each of these outcomes and this reward is proportional to the quality of response with respect to initial prompt.



Proximal Policy Optimization (PPO) RL algo- Step3

In this step we pass unseen input sequences to the clone SFT model we got in step1. The model will generate response with respect to the input prompt. We pass the response to our reward model which we got in step 2 to understand, how high quality was this response for that input prompt and the output reward will be used to finetune the parameters of our SFT model .This is how our SFT model will incorporate more human like characteristics and behavior’s via Reinforcement Learning.