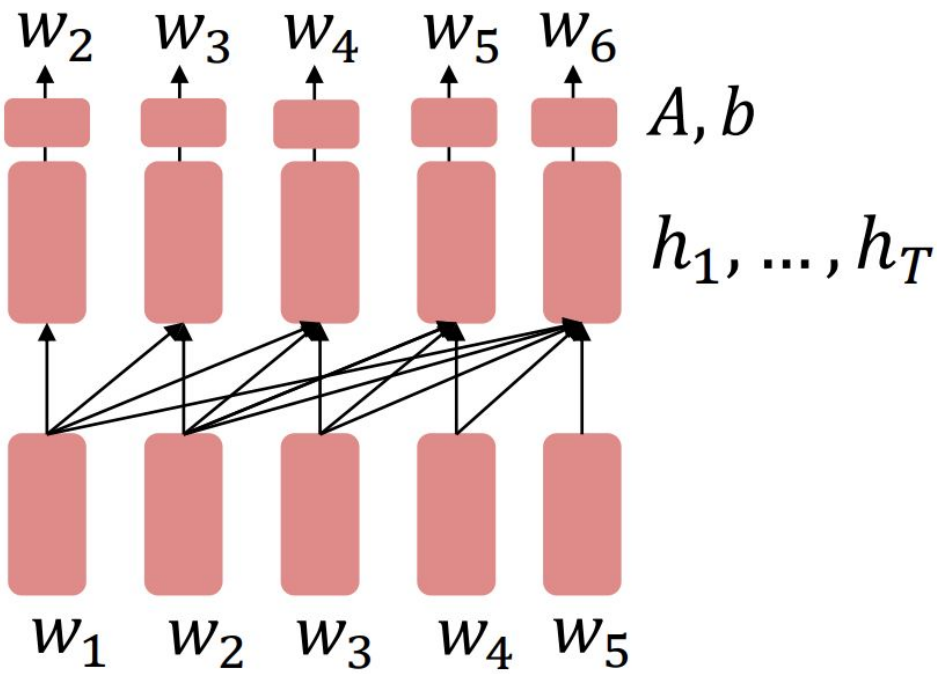
Pretraining Decoders

It's natural to pretrain decoders as language models

Pretraining objective ⇒ next word prediction



Text data → RNNs

Transformers

↳ Images (ViT)

Pretraining

Encoders (BERT)

Decoders (GPT, --)

Encoder-decoders (T5, BART) ..

Joint on

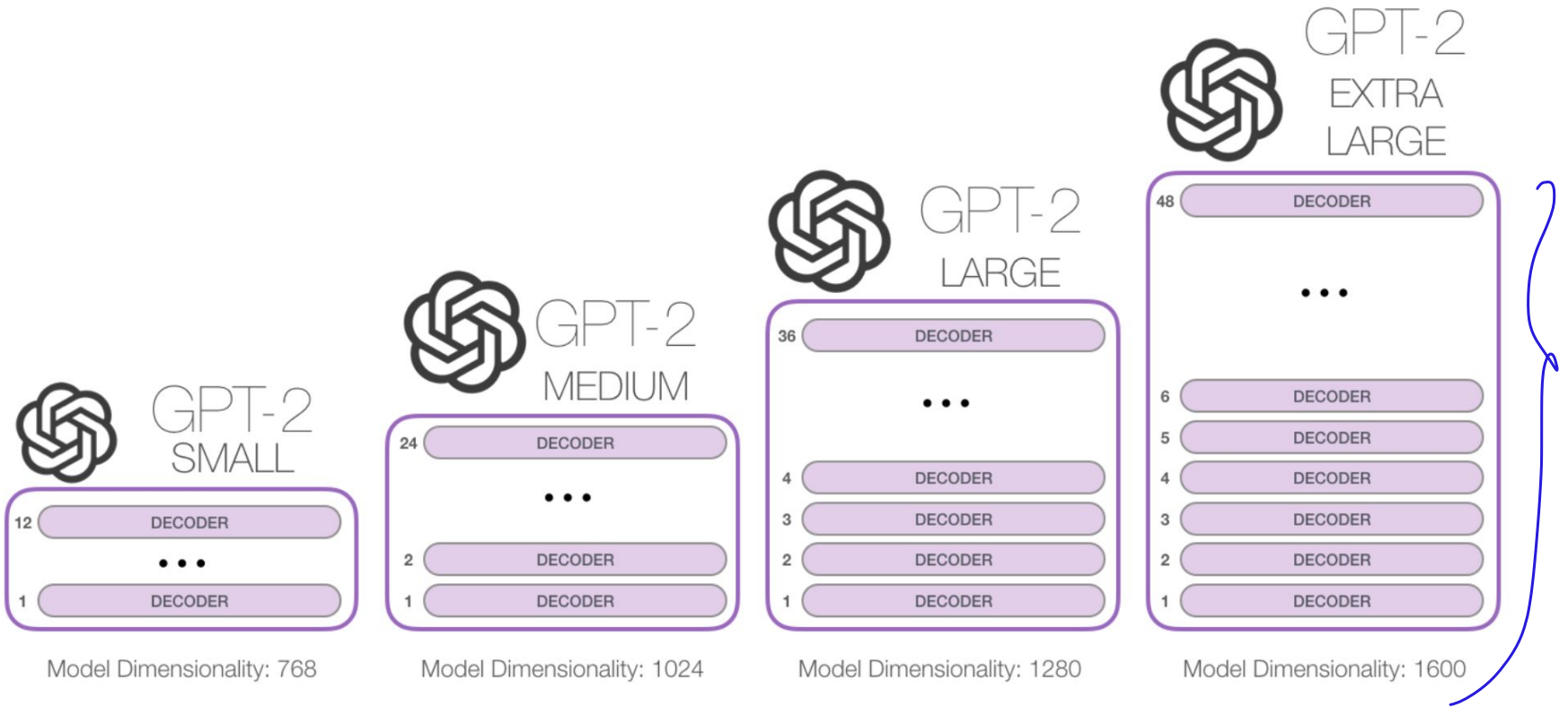
Image & Text together

Generative Pretrained Transformer (GPT)

- Transformer decoder with 12 layers, 117M parameters.
- 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.

Vocabulary larger than BERT

Pretrained Decoders



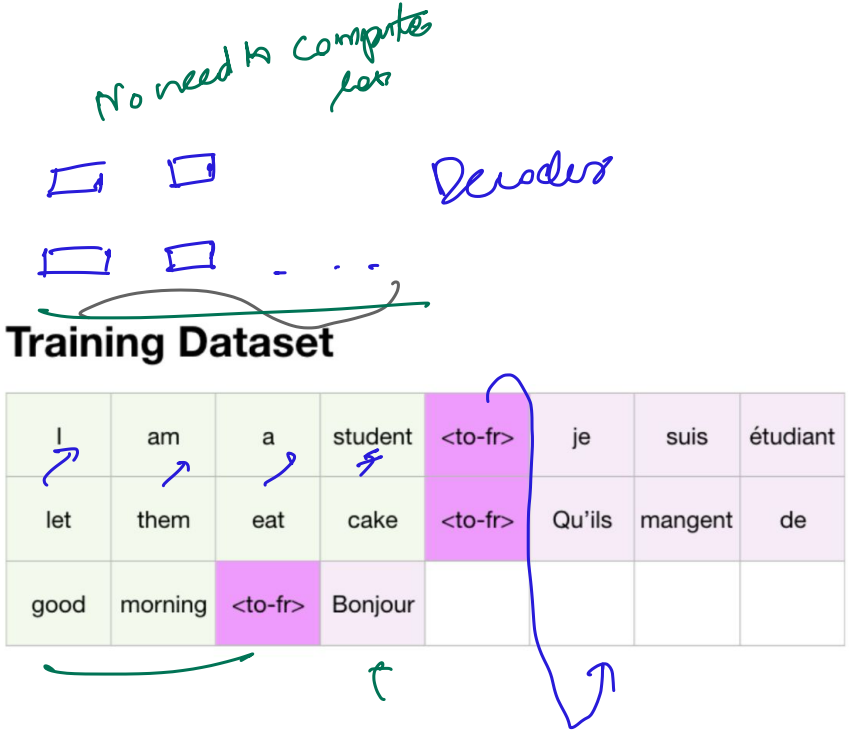
GPT-2: Beyond Language Modeling

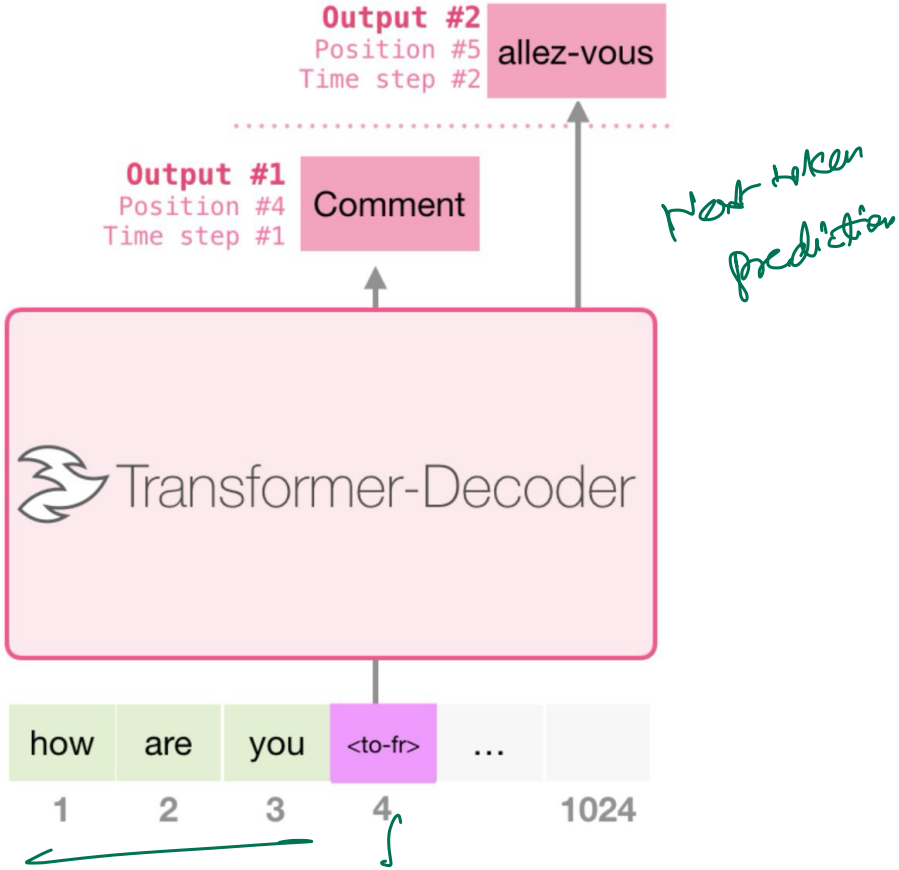
Basic Idea

The decoder can work with a prompt!

Any NLP task can be expressed in a probabilistic framework as estimating a conditional distribution $p(output|input)$.

GPT-2: Machine Translation

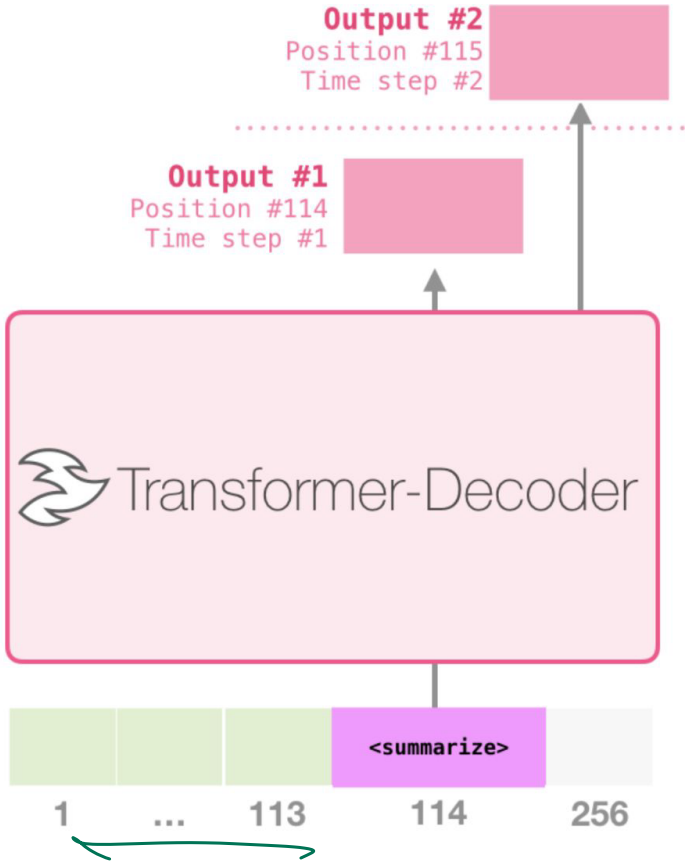




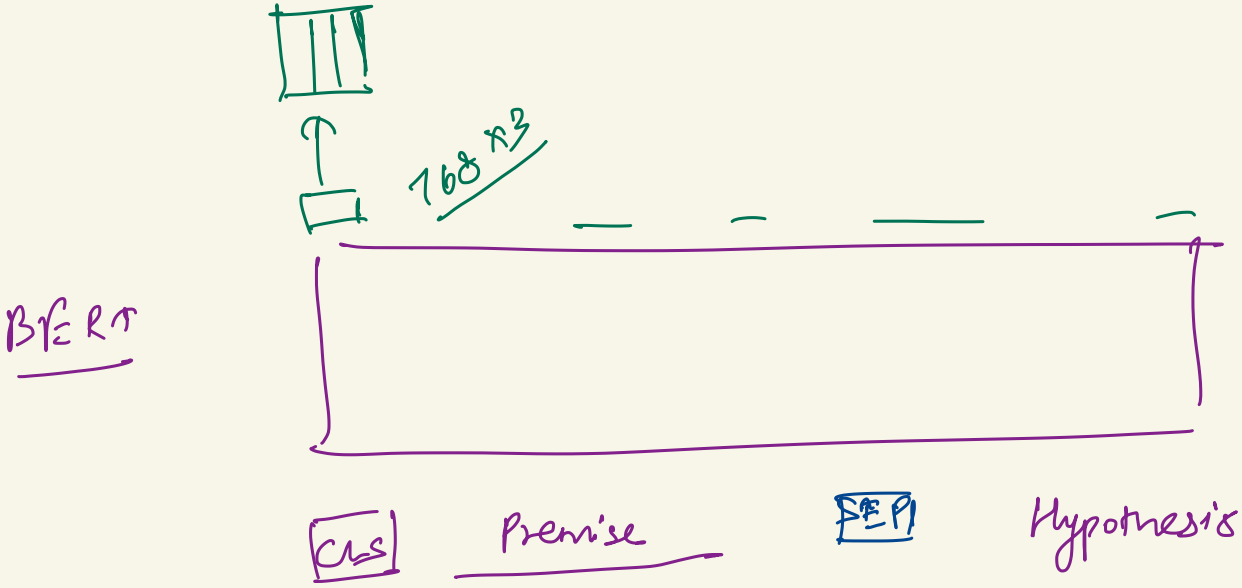
GPT-2: Summarization

Training Dataset

Article #1 tokens	<summarize>	Article #1 Summary
Article #2 tokens	<summarize>	Article #2 Summary
padding		
Article #3 tokens	<summarize>	Article #3 Summary




final
self
only our output tokens
masked self-attention
doc
sum
summary


BERT
768 x 3
CLS
Premise
SEP
Hypothesis

How to format inputs?

Natural Language Inference

Premise: *The man is in the doorway*
Hypothesis: *The person is near the door*

} **entailment** / contradict / neutral

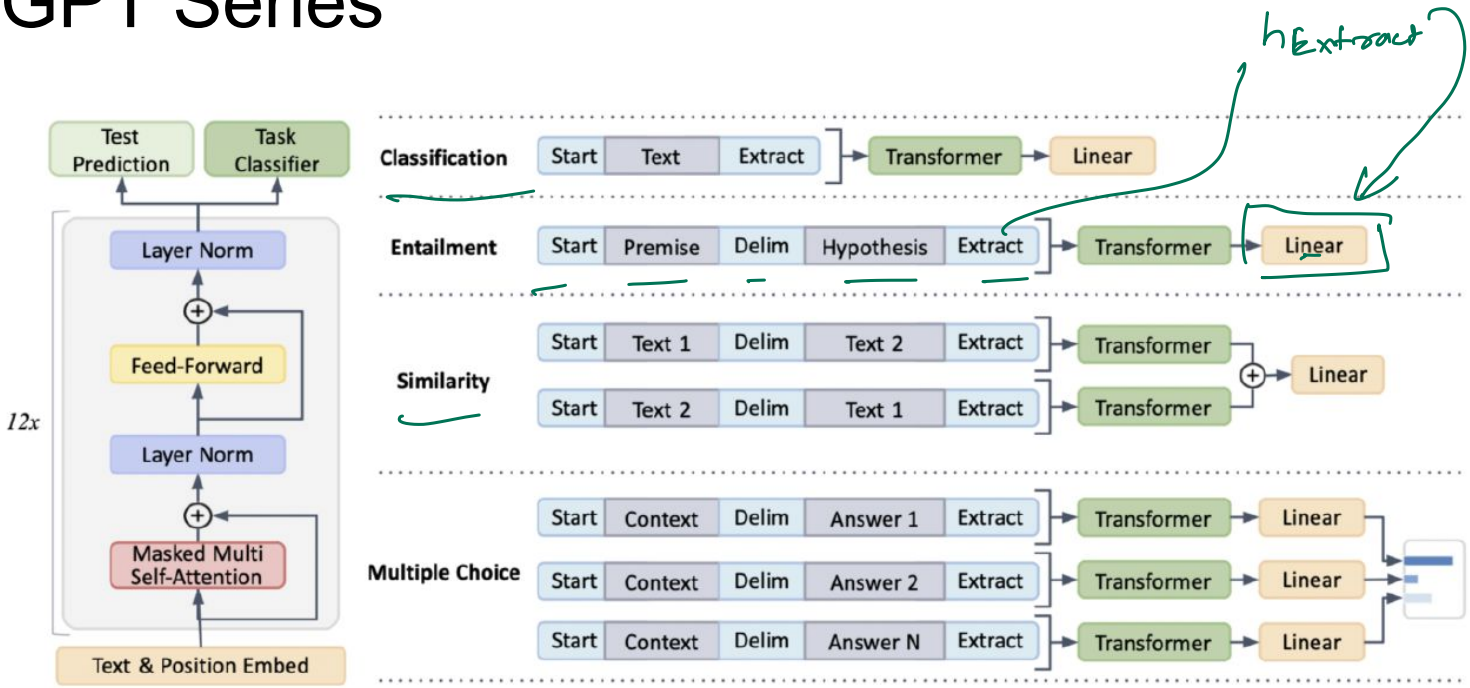
Radford et al., 2018 evaluate on natural language inference.

Here's roughly how the input was formatted, as a sequence of tokens for the decoder.

[START] *The man is in the doorway* [DELIM] *The person is near the door* [EXTRACT]

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

GPT Series



The GPT-1 architecture and designated training objectives employed for training. Structured inputs are converted into sequences of tokens for fine-tuning different tasks, which the pre-trained model processes, followed by implementing a linear layer with a softmax layer.

Why is "Extract" at the end?

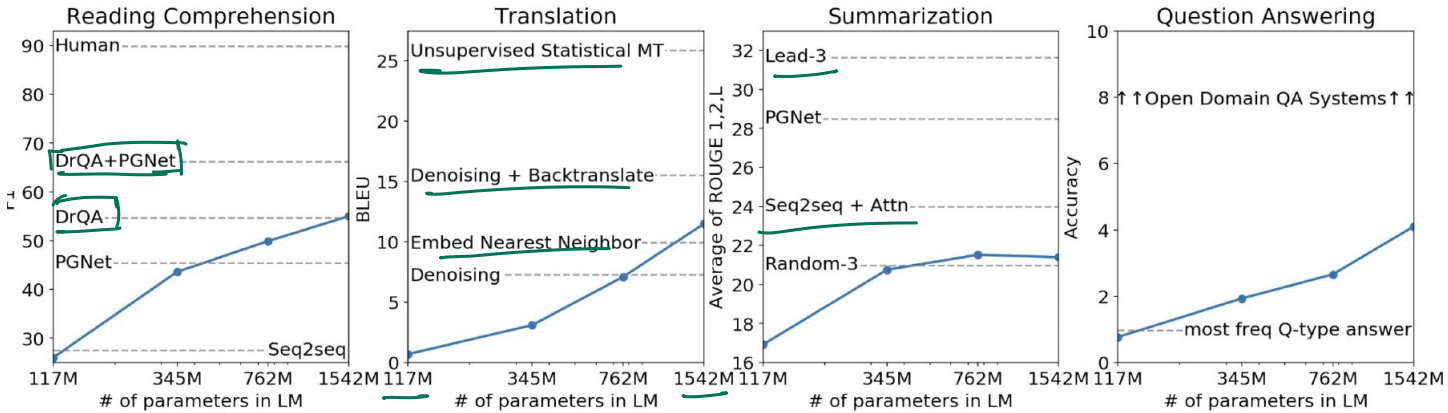
Image Source: "Large Language Models: A Deep Dive"

GPT Series

GPT-2 (Radford et al. 2019) scaled the GPT model to a larger model.

Demonstrated that language models are able to do NLP tasks **without any explicit supervision (zero-shot)**.

- e.g., when conditioned on a document plus questions, the answers generated by GPT-2 reach 55 F1 on the CoQA dataset.



Zero-shot task performance of GPT-2 as a function of model size on many NLP tasks

In-context learning: Terminology

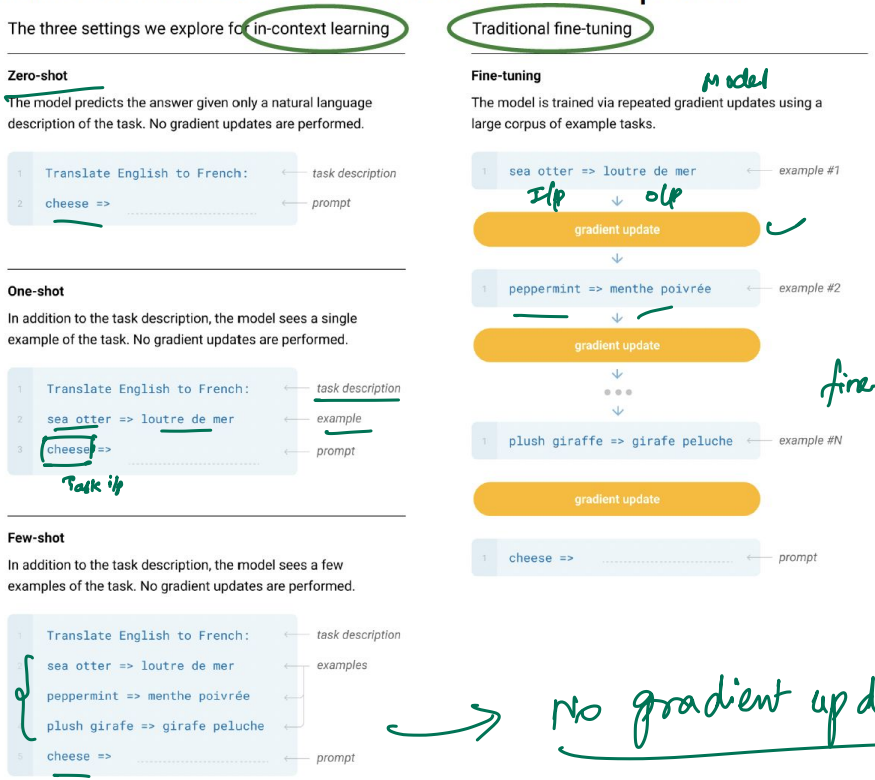
- In-context learning: A frozen LM performs a task only by conditioning on the prompt text.
- Few-shot in-context learning: (1) The prompt includes examples of the intended behavior, and (2) no examples of the intended behavior were seen in training.
  - We are unlikely to be able to verify (2).
  - "Few-shot" is also used in supervised learning with the sense of "training on few examples". The above is different.
- Zero-shot in-context learning: (1) The prompt includes no examples of the intended behavior (but it can contain other instructions), and (2) no examples of the intended behavior were seen in training.
  - We are unlikely to be able to verify (2).
  - Formatting and other instructions seem like a gray area, but we will allow them in this category.

GPT Series

Generative Pretrained Transformer series of models from OpenAI

GPT-3 (Brown et al. 2020) further scaled the GPT-2 model to 175 Billion (100 times larger compared to the largest GPT-2 model).

In addition to improved zero-shot performance, GPT-3 also exhibits strong few-shot **"in-context learning" ability**

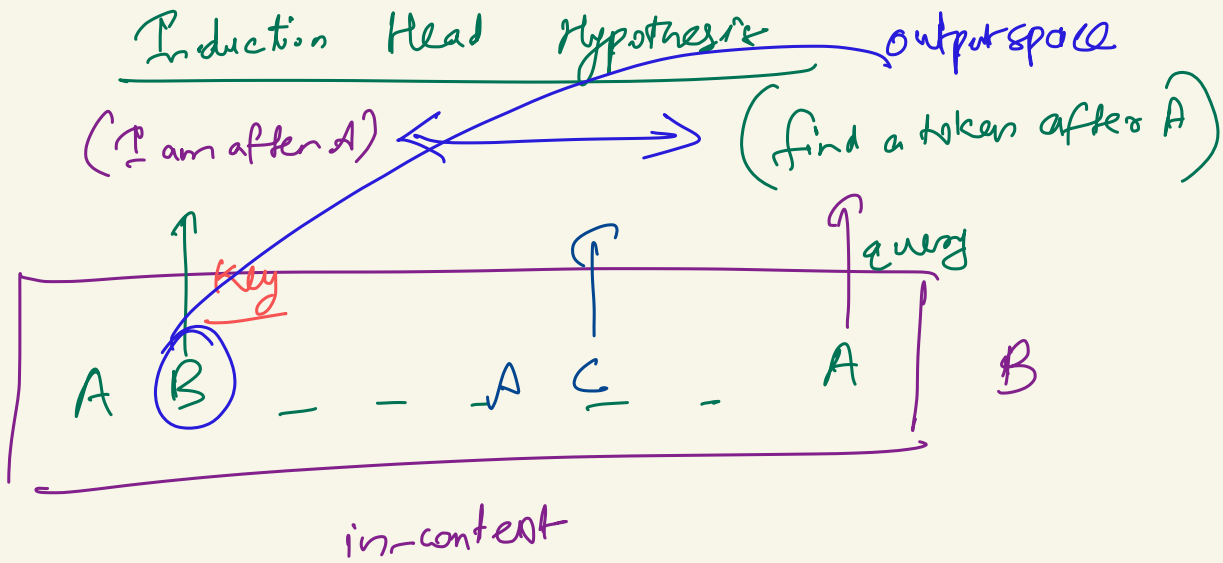


In in-context learning, we append the task description and a few demonstrations of the task to the original input as the input for the language model, which produces the prediction by auto-regressively predicting next tokens

Comparison of GPT series

Characteristic	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 Size	768	1600	12288
Batch Size	64	512	3.2M

Image Source: “Large Language Models: A Deep Dive”



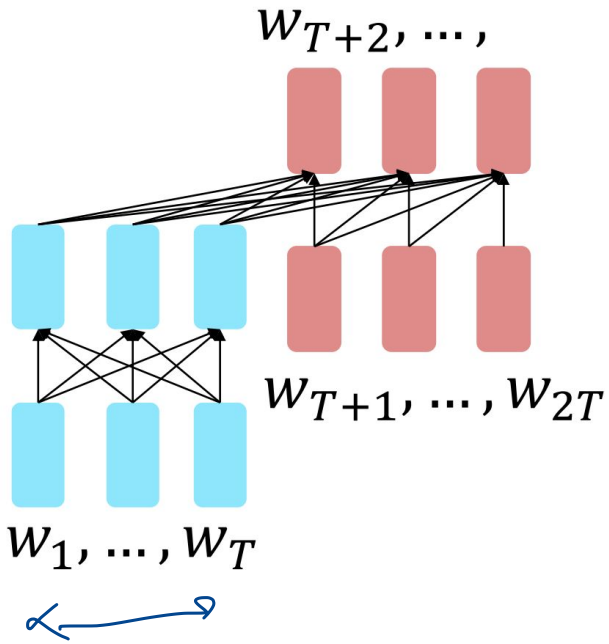
pattern matching

& copying beh^{r}

Pretraining encoder-decoders

What pretraining objective to use?

For encoder-decoders, we could do something like language modeling, but where a prefix of every input is provided to the encoder and is not predicted.



The encoder portion benefits from bidirectional context; the decoder portion is used to train the whole model through language modeling.

T5: A New Training Objective

Span Corruption

decode out the spans that were corrupted

Original text

Thank you ~~for inviting~~ me to your party ~~last~~ week.

Inputs

Thank you <X> me to your party <Y> week.

Targets

<X> for inviting <Y> last <Z>

Various architectures considered by T5

Encoder-Decoder ✓

Decoder

Prefix LM

Various objectives considered by T5

Objective	Inputs	Targets
Prefix language modeling	Thank you for inviting	me to your party last week .
BERT-style Devlin et al. (2018)	Thank you <M> <M> me to your party apple week .	*(original text)*
Deshuffling	party me for your to . last fun you inviting week Thank	*(original text)*
MASS-style Song et al. (2019)	Thank you <M> <M> me to your party <M> week .	*(original text)*
I.i.d. noise, replace spans	Thank you <X> me to your party <Y> week .	<X> for inviting <Y> last <Z>
I.i.d. noise, drop tokens	Thank you me to your party week .	for inviting last
Random spans	Thank you <X> to <Y> week .	<X> for inviting me <Y> your party last <Z>

Objective	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
Prefix language modeling	80.69	18.94	77.99	65.27	**26.86**	39.73	**27.49**
BERT-style (Devlin et al., 2018)	**82.96**	**19.17**	**80.65**	**69.85**	**26.78**	**40.03**	**27.41**
Deshuffling	73.17	18.59	67.61	58.47	26.11	39.30	25.62

Objective	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
BERT-style (Devlin et al., 2018)	82.96	19.17	**80.65**	69.85	26.78	**40.03**	27.41
MASS-style (Song et al., 2019)	82.32	19.16	80.10	69.28	26.79	**39.89**	27.55
★ Replace corrupted spans	83.28	**19.24**	**80.88**	**71.36**	**26.98**	39.82	**27.65**
Drop corrupted tokens	**84.44**	**19.31**	**80.52**	68.67	**27.07**	39.76	**27.82**

Various architectures considered by T5

Architecture	Objective	Params	Cost	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Encoder-decoder	Denoising	$2P$	M	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Enc-dec, shared	Denoising	P	M	82.81	18.78	**80.63**	**70.73**	26.72	39.03	**27.46**
Enc-dec, 6 layers	Denoising	P	$M/2$	80.88	18.97	77.59	68.42	26.38	38.40	26.95
Language model	Denoising	P	M	74.70	17.93	61.14	55.02	25.09	35.28	25.86
Prefix LM	Denoising	P	M	81.82	18.61	78.94	68.11	26.43	37.98	27.39
Encoder-decoder	LM	$2P$	M	79.56	18.59	76.02	64.29	26.27	39.17	26.86
Enc-dec, shared	LM	P	M	79.60	18.13	76.35	63.50	26.62	39.17	27.05
Enc-dec, 6 layers	LM	P	$M/2$	78.67	18.26	75.32	64.06	26.13	38.42	26.89
Language model	LM	P	M	73.78	17.54	53.81	56.51	25.23	34.31	25.38
Prefix LM	LM	P	M	79.68	17.84	76.87	64.86	26.28	37.51	26.76

T5 can be used for various tasks

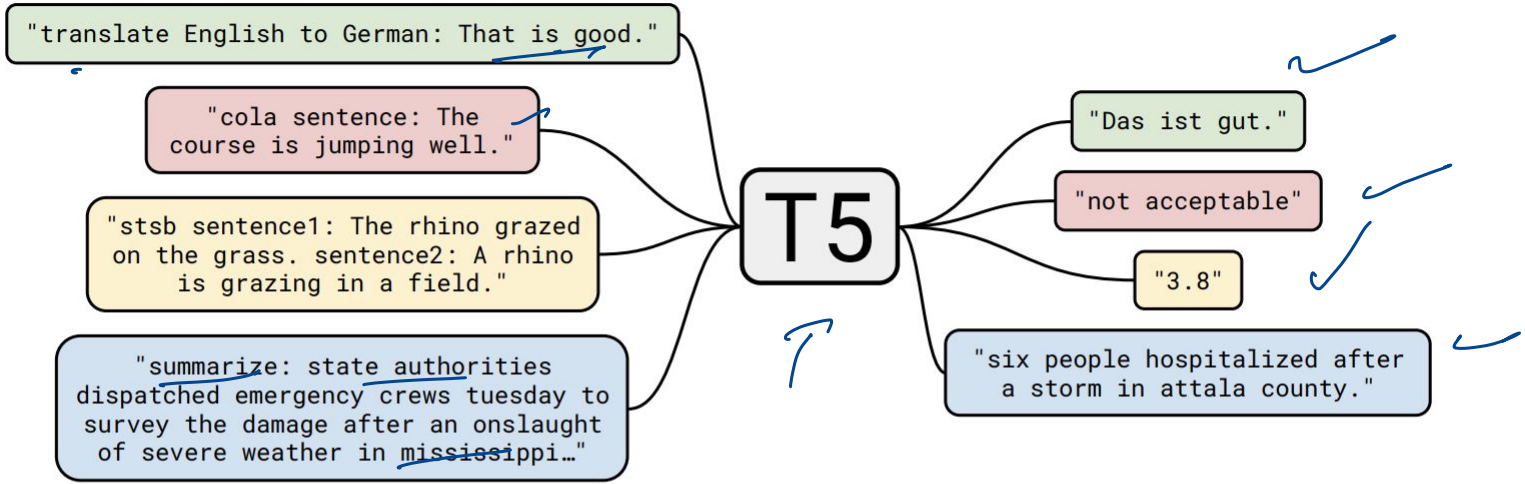
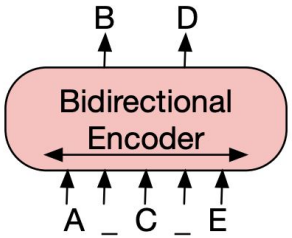
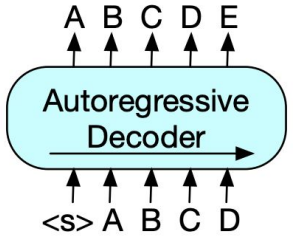


Figure 1: A diagram of our text-to-text framework. Every task we consider—including translation, question answering, and classification—is cast as feeding our model text as input and training it to generate some target text. This allows us to use the same model, loss function, hyperparameters, etc. across our diverse set of tasks. It also provides a standard testbed for the methods included in our empirical survey. "T5" refers to our model, which we dub the "**T**ext-**t**o-**T**ext **T**ransfer **T**ransformer".

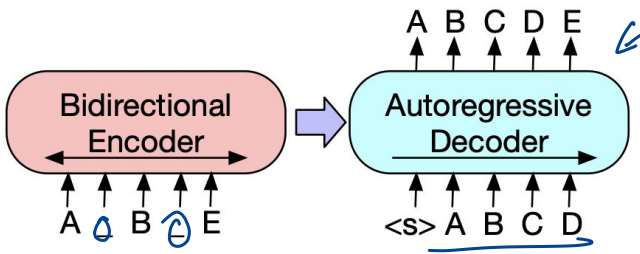
Bi-directional and Auto-Regressive Transformers (BART)



(a) BERT: Random tokens are replaced with masks, and the document is encoded bidirectionally. Missing tokens are predicted independently, so BERT cannot easily be used for generation.



(b) GPT: Tokens are predicted auto-regressively, meaning GPT can be used for generation. However words can only condition on leftward context, so it cannot learn bidirectional interactions.



(c) BART: Inputs to the encoder need not be aligned with decoder outputs, allowing arbitary noise transformations. Here, a document has been corrupted by replacing spans of text with mask symbols. The corrupted document (left) is encoded with a bidirectional model, and then the likelihood of the original document (right) is calculated with an autoregressive decoder. For fine-tuning, an uncorrupted document is input to both the encoder and decoder, and we use representations from the final hidden state of the decoder.

How to fine-tune pretrained encoder-decoder?

No additional parameters required!

Just fine-tune the encoder decoder model through additional task-specific training data

Loss computed at decoder is used to update all the encoder and decoder parameters