LLMs 1

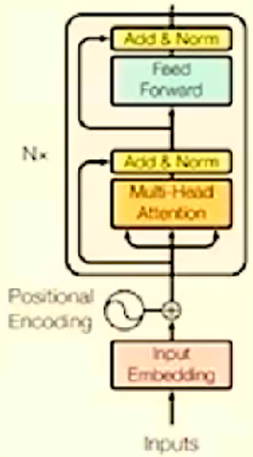
Intro to NLP

Rahul Mishra
IIIT-Hyderabad
Apr 05 & 12, 2024

LLMs

BERT

[*] [*] [sat_] [*] [the_] [*]



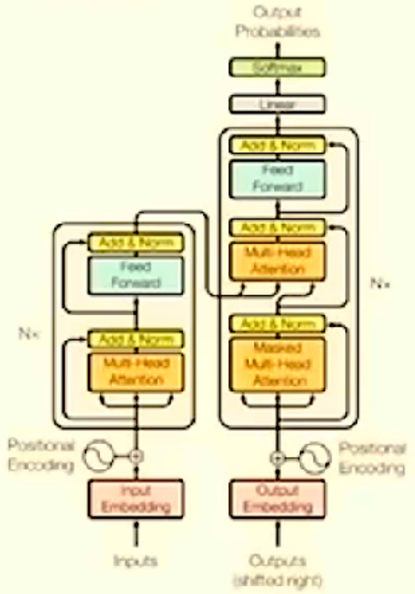
[The_] [cat_] [MASK] [on_] [MASK] [mat_]

T5

Das ist gut.

A storm in Attala caused 6 victims.

This is not toxic.



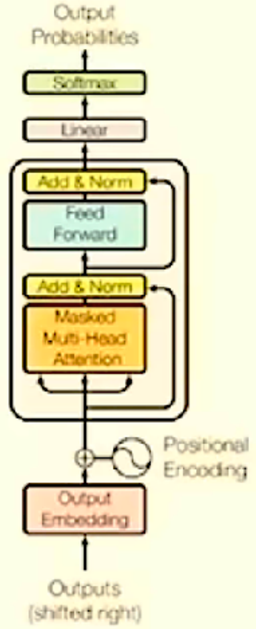
Translate EN-DE: This is good.

Summarize: state authorities dispatched…

Is this toxic: You look beautiful today!

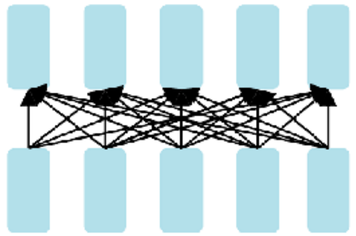
GPT

[sat_]



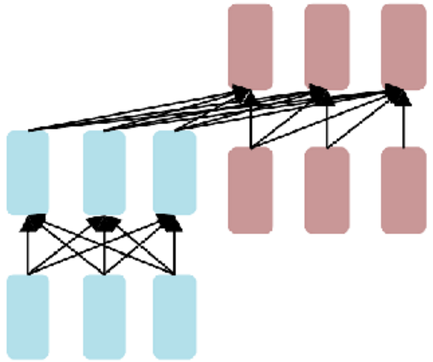
[START] [The_] [cat_]

LLMs



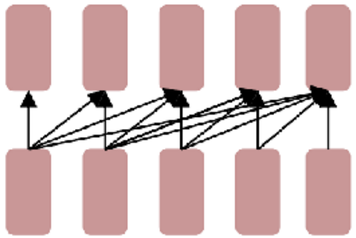
Encoders

- Gets bidirectional context – can condition on future!
- How do we train them to build strong representations?



Encoder-Decoders

- Good parts of decoders and encoders?
- What's the best way to pretrain them?



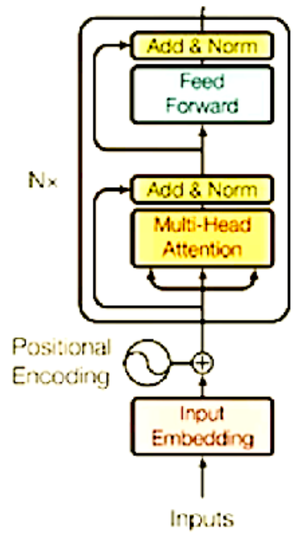
Decoders

- Language models! What we've seen so far.
- Nice to generate from; can't condition on future words

BERT, 2019

- *Bidirectional* Encoder Representations from Transformers
- Encoder-only (no attention masking)
- 110M params
- 15% of all words masked out
- Was great, now dated

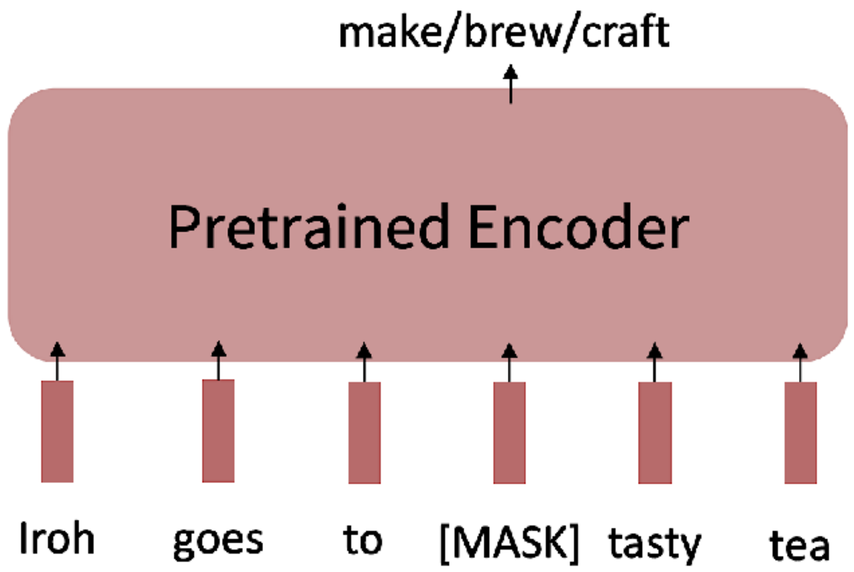
[*] [*] **[sat_]** [*] **[the_]** [*]

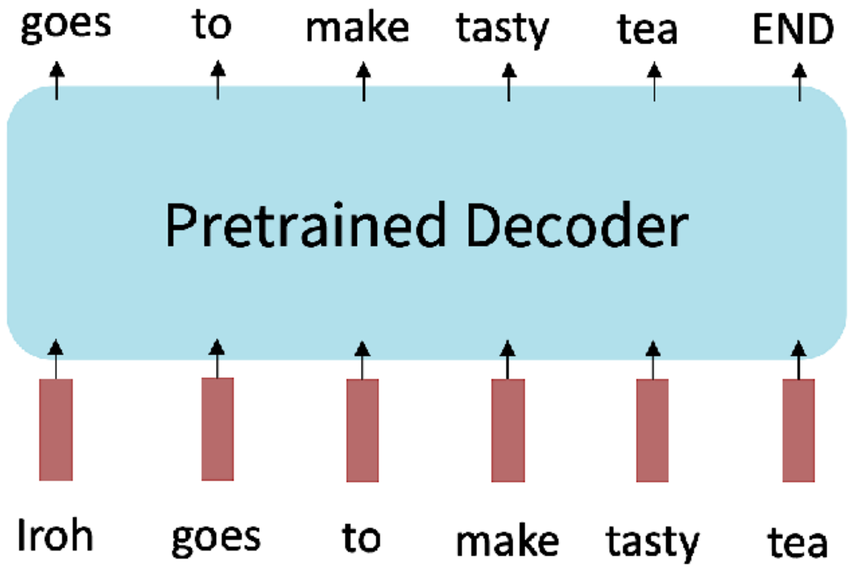


[The_] [cat_] **[MASK]** [on_] **[MASK]** [mat_]

BERT, 2019

If your task involves generating sequences, consider using a pretrained decoder; BERT and other pretrained encoders don't naturally lead to nice autoregressive (1-word-at-a-time) generation methods.

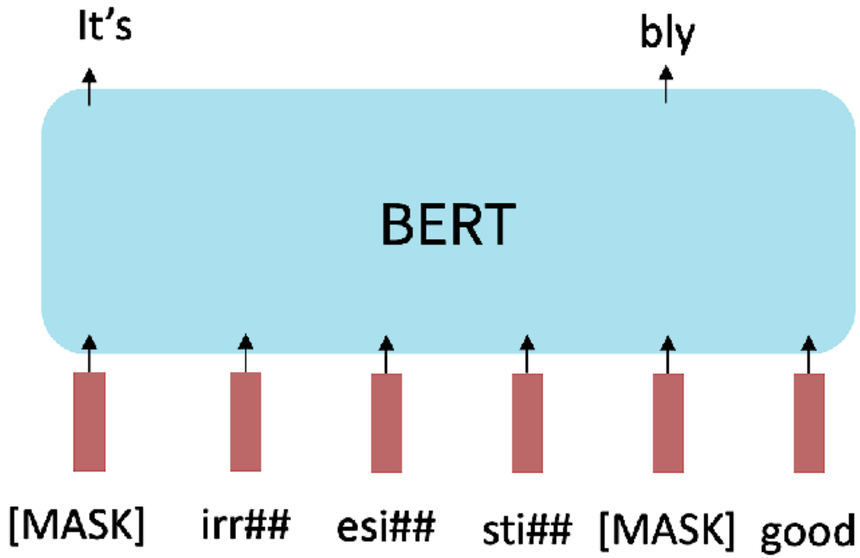


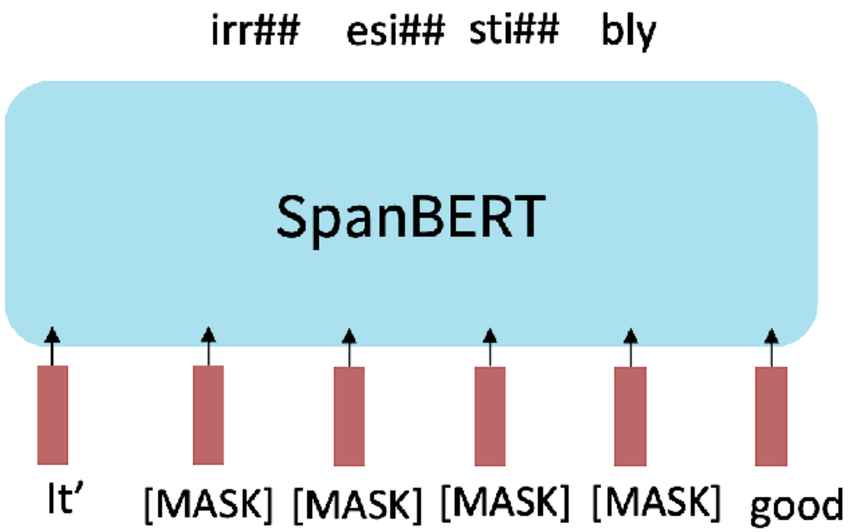


BERT, 2019

Some generally accepted improvements to the BERT pretraining formula:

- RoBERTa: mainly just train BERT for longer and remove next sentence prediction!
- SpanBERT: masking contiguous spans of words makes a harder, more useful pretraining task





T5, text-to-text, 2020

- Unsupervised pre-training on Colossal Clean Crawled Corpus (C4)
  - Start with Common Crawl (over 50TB of compressed data, 10B+ web pages)
  - Filtered down to ~800GB, or ~160B tokens
- Also trained on academic supervised tasks

- We discarded any page with fewer than 5 sentences and only retained lines that contained at least 3 words.
- We removed any page that contained any word on the "List of Dirty, Naughty, Obscene or Otherwise Bad Words".[6]
- Some pages inadvertently contained code. Since the curly bracket "{" appears in many programming languages (such as Javascript, widely used on the web) but not in natural text, we removed any pages that contained a curly bracket.
- To deduplicate the data set, we discarded all but one of any three-sentence span occurring more than once in the data set.

https://paperswithcode.com/dataset/c4

Sentence acceptability judgment

CoLA Warstadt et al., 2018

Sentiment analysis

SST-2 Socher et al., 2013

Paraphrasing/sentence similarity

MRPC Dolan and Brockett, 2005

STS-B Cer et al., 2017

QQP Iyer et al., 2017

Natural language inference

MNLI Williams et al., 2017

QNLI Rajpurkar et al., 2016

RTE Dagan et al., 2005

CB De Marneff et al., 2019

Sentence completion

COPA Roemmele et al., 2011

Word sense disambiguation

WiC Pilehvar and Camacho-Collados, 2018

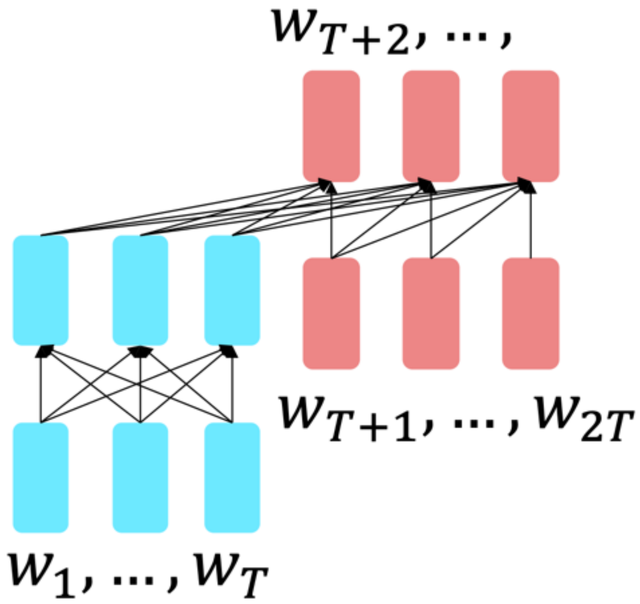
Question answering

MultiRC Khashabi et al., 2018

ReCoRD Zhang et al., 2018

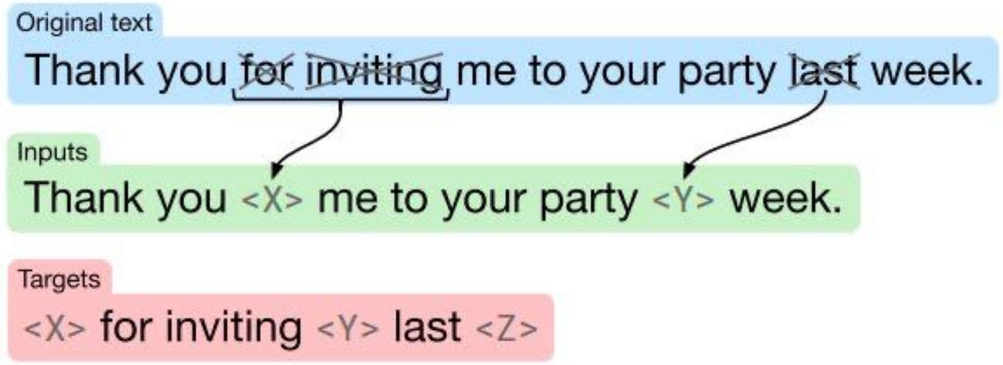
BoolQ Clark et al., 2019

T5, text-to-text, Pretraining



[Raffel et al., 2018]

- Span corruption
- Split Sentence



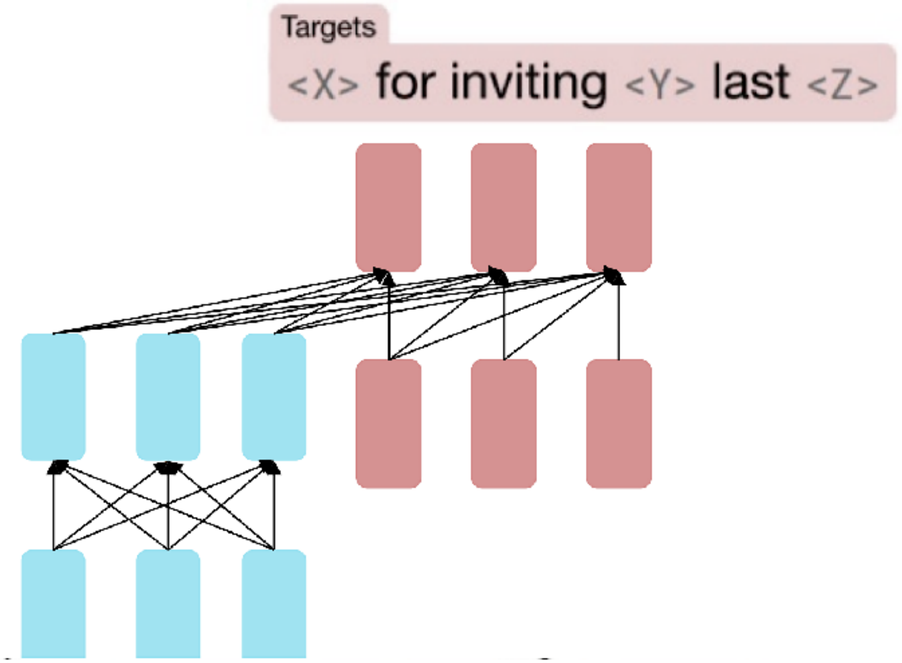
T5, text-to-text, Pretraining

Replace different-length spans from the input with unique placeholders; decode out the spans that were removed!

Original text

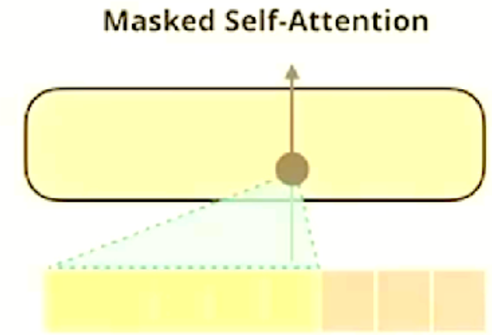
Thank you ~~for inviting~~ me to your party ~~last~~ week.

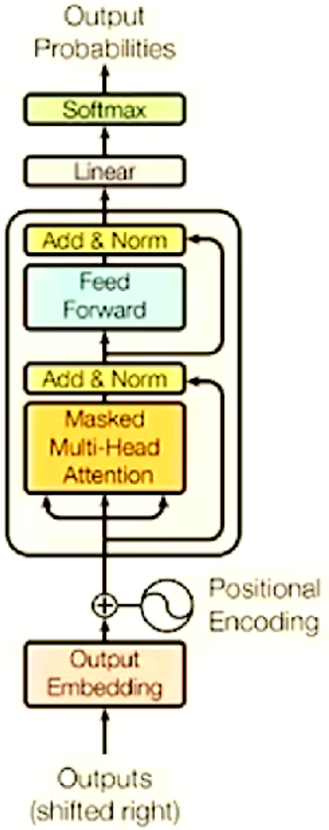
This is implemented in text preprocessing: it's still an objective that looks like **language modeling** at the decoder side.



GPT/GPT 2, 2019

- Generative Pre-trained Transformer
- Decoder-only (uses masked self-attention)
- Largest model is 1.5B





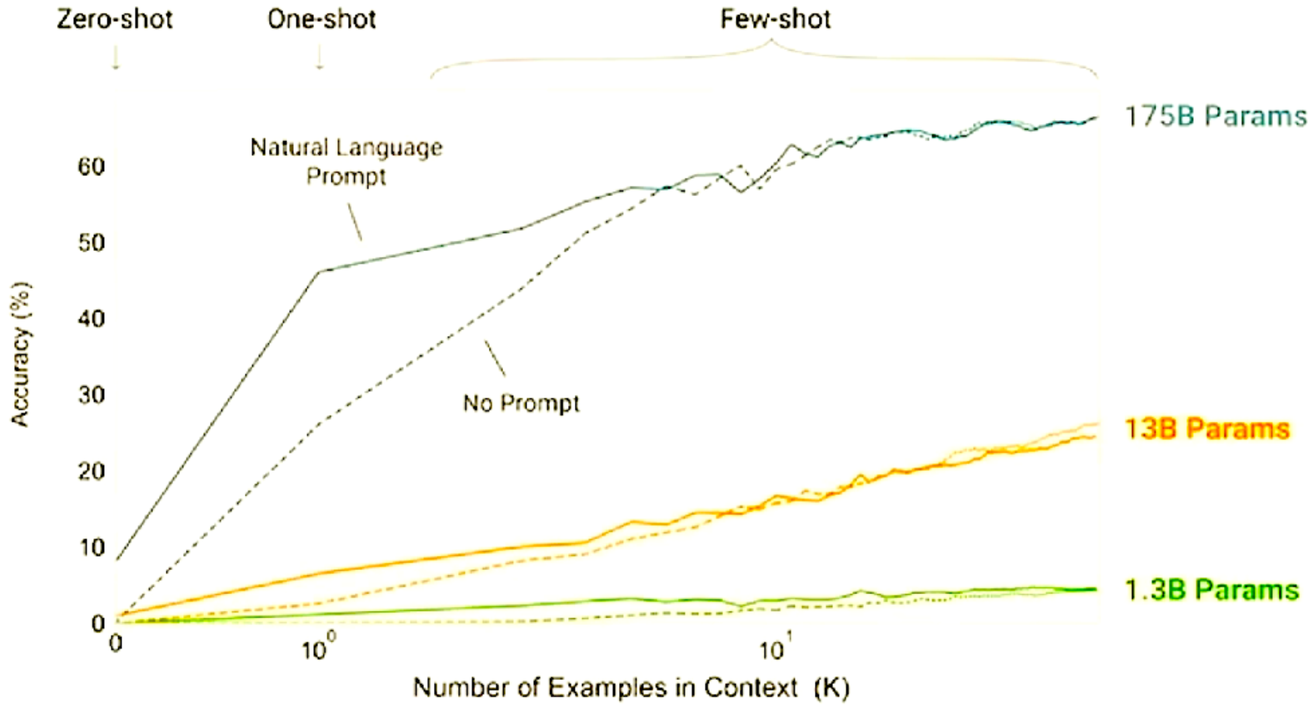
[START] [The_] [cat_]

GPT 2, Training

- Found that Common Crawl has major data quality issues
- Formed the WebText dataset
  - scraped all outbound links (45M) from Reddit which received at least 3 karma
- After de-duplication and some heuristic filtering, left with 8M documents for a total of 40GB of text

GPT 3

- Just like GPT-2, but 100x larger (175B params)
- Exhibited unprecedented few-shot and zero-shot learning



Zero-shot

The model predicts the answer given only a natural language discription of the task. No gradient updates are performed.

```
Translate English to French:        task description
cheese =>                           prompt
```

One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

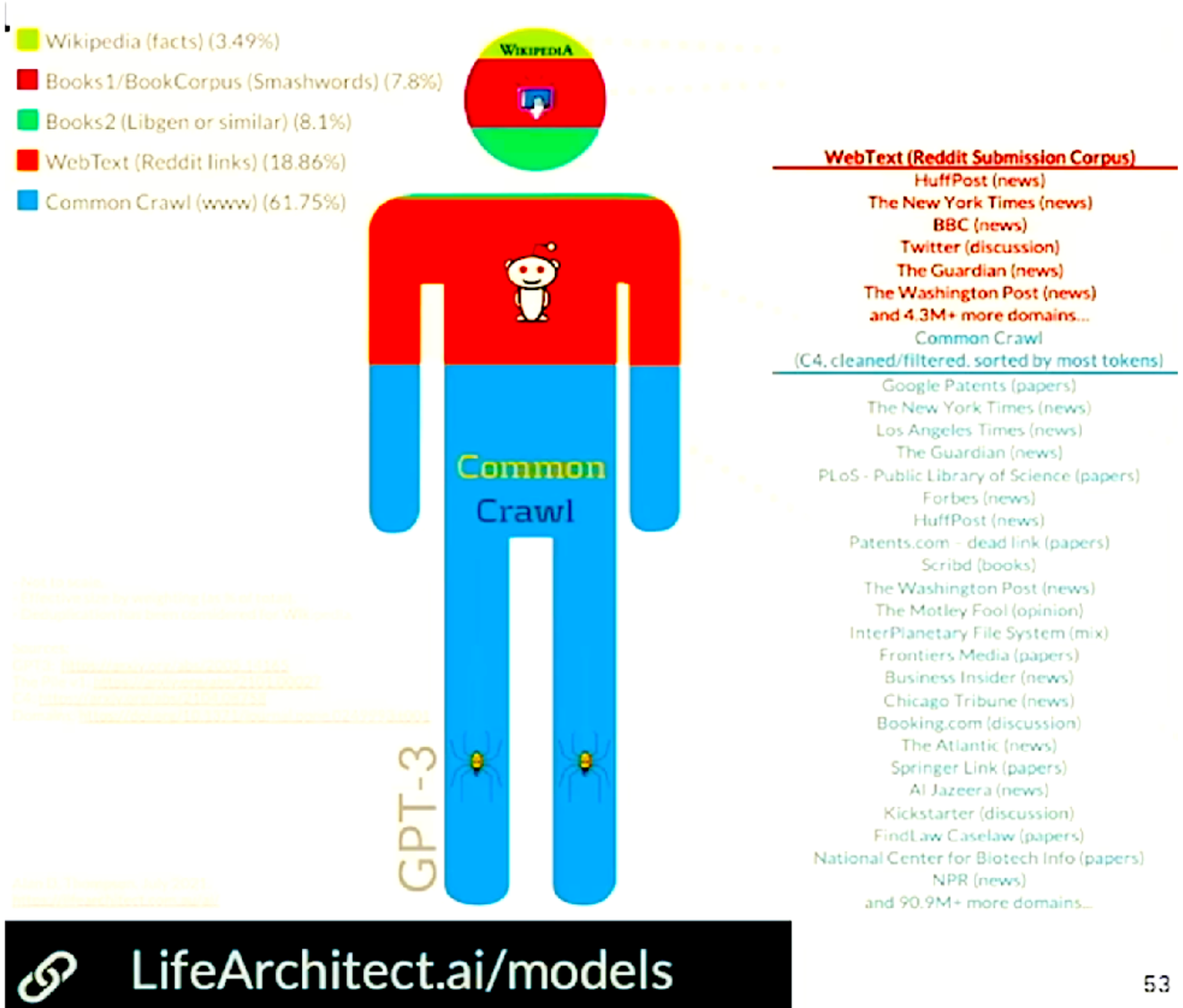
```
Translate English to French:        task description
sea otter => loutre de mer          example
cheese =>                           prompt
```

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
Translate English to French:        task description
sea otter => loutre de mer          examples
peppermint => menthe poivrée
plush girafe => girafe peluche
  eese =>                           prompt
```

GPT 3


Wikipedia (facts) (3.49%)
Books1/BookCorpus (Smashwords) (7.8%)
Books2 (Libgen or similar) (8.1%)
WebText (Reddit links) (18.86%)
Common Crawl (www) (61.75%)
WIKIPEDIA
Common
Crawl
GPT-3
WebText (Reddit Submission Corpus)
HuffPost (news)
The New York Times (news)
BBC (news)
Twitter (discussion)
The Guardian (news)
The Washington Post (news)
and 4.3M+ more domains...
Common Crawl
(C4, cleaned/filtered, sorted by most tokens)
Google Patents (papers)
The New York Times (news)
Los Angeles Times (news)
The Guardian (news)
PLoS - Public Library of Science (papers)
Forbes (news)
HuffPost (news)
Patents.com - dead link (papers)
Scribd (books)
The Washington Post (news)
The Motley Fool (opinion)
InterPlanetary File System (mix)
Frontiers Media (papers)
Business Insider (news)
Chicago Tribune (news)
Booking.com (discussion)
The Atlantic (news)
Springer Link (papers)
Al Jazeera (news)
Kickstarter (discussion)
FindLaw Caselaw (papers)
National Center for Biotech Info (papers)
NPR (news)
and 90.9M+ more domains...
LifeArchitect.ai/models

GPT 3: In Context Learning

Very large language models seem to perform some kind of learning **without gradient steps** simply from examples you provide within their contexts.

The in-context examples seem to specify the task to be performed, and the conditional distribution mocks performing the task to a certain extent.

Input (prefix within a single Transformer decoder context):

“ thanks -> merci

hello -> bonjour

mint -> menthe

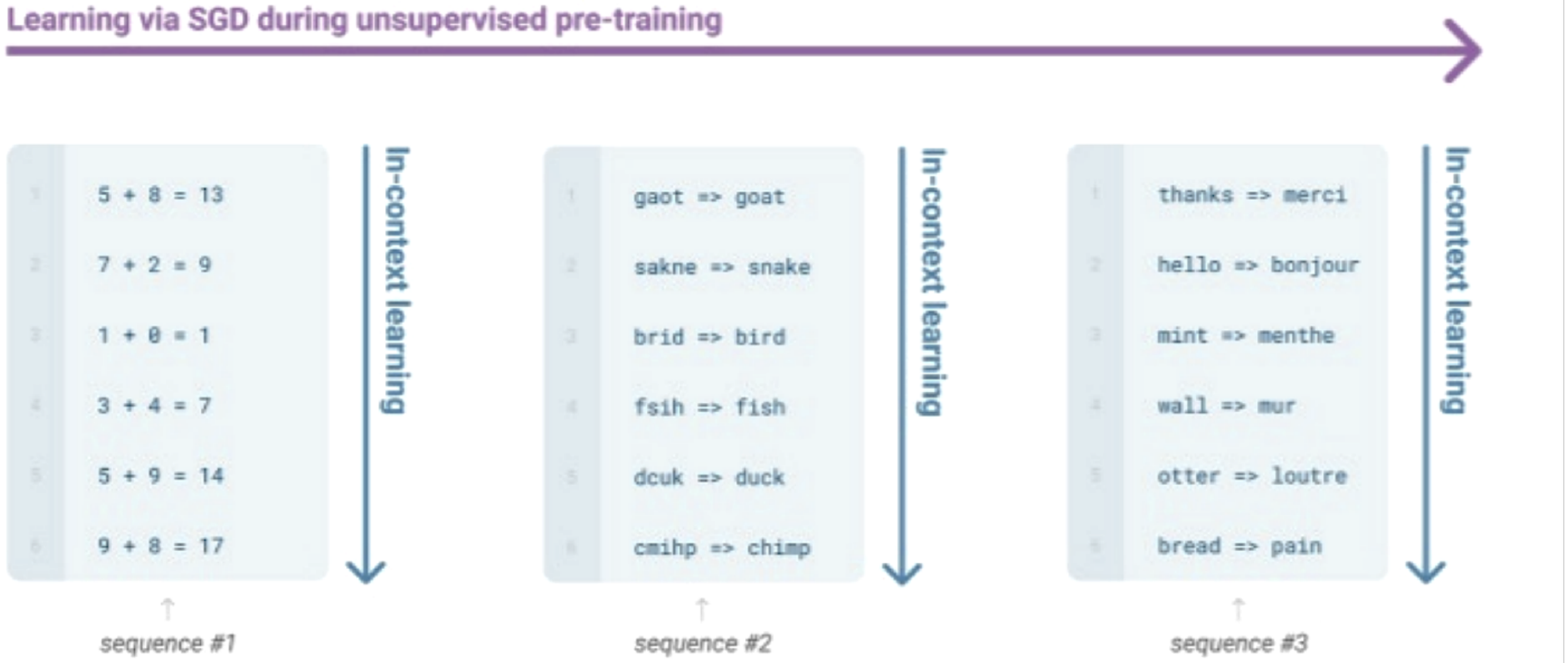
otter -> ”

Output (conditional generations):

loutre…”

GPT 3: In Context Learning

Very large language models seem to perform some kind of learning **without gradient steps** simply from examples you provide within their contexts.



GPT 3: In Context Learning

Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ✗

Chain-of-Thought Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9. The answer is 9. ✓

Chinchilla, 2022

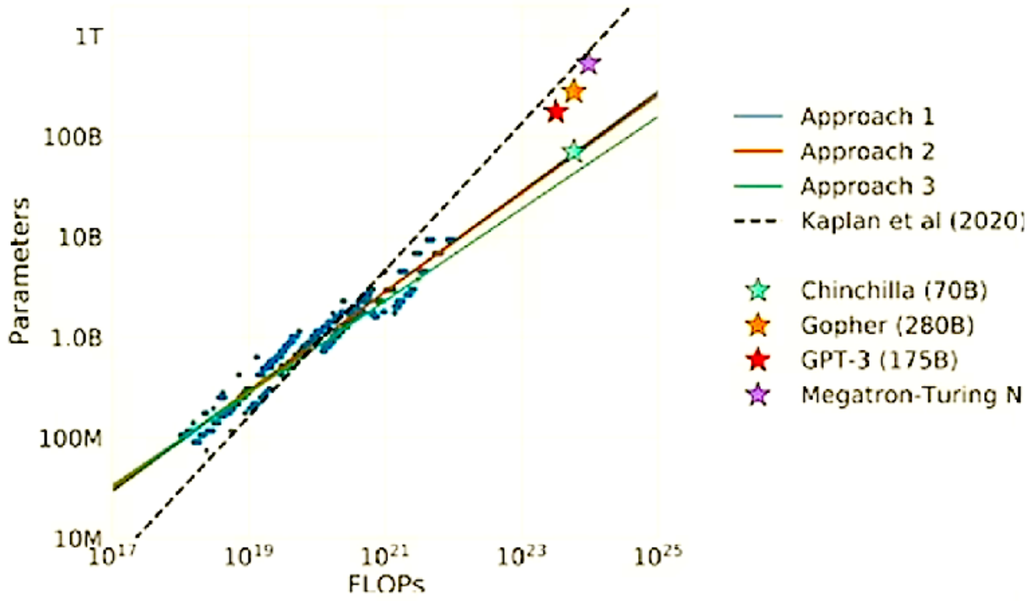
- Empirically derived formulas for optimal model and training set size given a fixed compute budget
- Found that most LLMs are "undertrained"
- Trained Chinchilla (70B) vs Gopher (280B) at the same compute budget, by using 4x fewer params and 4x more data

DeepMind

Training Compute-Optimal Large Language Models

Jordan Hoffmann*, Sebastian Borgeaud*, Arthur Mensch*, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals and Laurent Sifre*

*Equal contributions



Model	Size (# Parameters)	Training Tokens
LaMDA (Thoppilan et al., 2022)	137 Billion	168 Billion
GPT-3 (Brown et al., 2020)	175 Billion	300 Billion
Jurassic (Lieber et al., 2021)	178 Billion	300 Billion
Gopher (Rae et al., 2021)	280 Billion	300 Billion
MT-NLG 530B (Smith et al., 2022)	530 Billion	270 Billion
Chinch	70 Billion	1.4 Trillion

https://arxiv.org/pdf/2203.15556.pdf

LLaMA, 2023

- "Chinchilla-optimal" open-source LLMs from Meta
- Several sizes from 7B to 65B, trained on at least 1T tokens
- Benchmarks competitively against GPT-3 and other LLMs

params	dimension	n heads	n layers	learning rate	batch size	n tokens
6.7B	4096	32	32	$3.0e^{-4}$	4M	1.0T
13.0B	5120	40	40	$3.0e^{-4}$	4M	1.0T
32.5B	6656	52	60	$1.5e^{-4}$	4M	1.4T
65.2B	8192	64	80	$1.5e^{-4}$	4M	1.4T

Table 2: **Model sizes, architectures, and optimization hyper-parameters.**

		Humanities	STEM	Social Sciences	Other	Average
GPT-NeoX	20B	29.8	34.9	33.7	37.7	33.6
GPT-3	175B	40.8	36.7	50.4	48.8	43.9
Gopher	280B	56.2	47.4	71.9	66.1	60.0
Chinchilla	70B	63.6	54.9	79.3	**73.9**	67.5
PaLM	8B	25.6	23.8	24.1	27.8	25.4
	62B	59.5	41.9	62.7	55.8	53.7
	540B	**77.0**	**55.6**	**81.0**	69.6	**69.3**
LLaMA	7B	34.0	30.5	38.3	38.1	35.1
	13B	45.0	35.8	53.8	53.3	46.9
	33B	55.8	46.0	66.7	63.4	57.8
	65B	61.8	51.7	72.9	67.4	63.4

Table 9: **Massive Multitask Language Understanding (MMLU).** Five-shot accuracy.

LLaMA, 2023

- Custom quality-filtering of CommonCrawl + some C4 + Github + Wikipedia + Books + ArXiV + Stack Exchange
- RedPajama: open-source recreation

	RedPajama	LLaMA*
CommonCrawl	878 billion	852 billion
C4	175 billion	190 billion
Github	59 billion	100 billion
Books	26 billion	25 billion
ArXiv	28 billion	33 billion
Wikipedia	24 billion	25 billion
StackExchange	20 billion	27 billion
Total	1.2 trillion	1.25 trillion

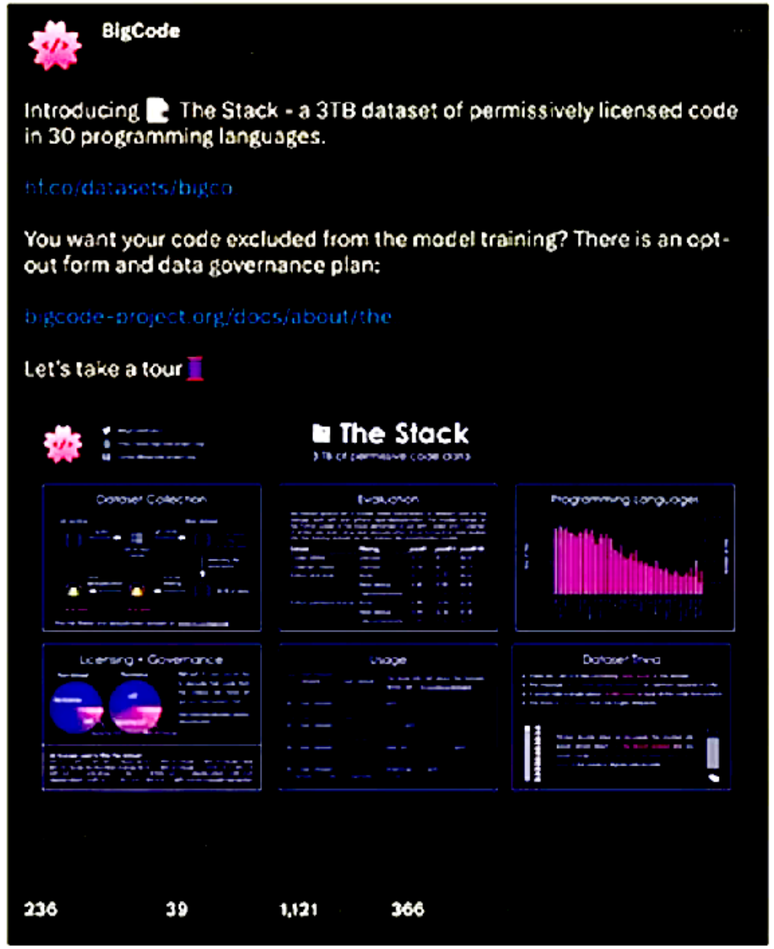
Dataset	Sampling prop.	Epochs	Disk size
CommonCrawl	67.0%	1.10	3.3 TB
C4	15.0%	1.06	783 GB
Github	4.5%	0.64	328 GB
Wikipedia	4.5%	2.45	83 GB
Books	4.5%	2.23	85 GB
ArXiv	2.5%	1.06	92 GB
StackExchange	2.0%	1.03	78 GB

Table 1: **Pre-training data.** Data mixtures used for pre-training, for each subset we list the sampling proportion, number of epochs performed on the subset when training on 1.4T tokens, and disk size. The pre-training runs on 1T tokens have the same sampling proportion.

https://arxiv.org/pdf/2302.13971.pdf

Code in training?

- T5 and GPT-3 (2020) specifically removed code. But most recent models are trained on ~5% code. Why?
- Code-specific models such as OpenAI Codex (2021) was GPT-3 further trained on public GitHub code.
- Empirically, this improved performance on non-code tasks!
- Open-source dataset: The Stack (3TB of permissively licensed source code)



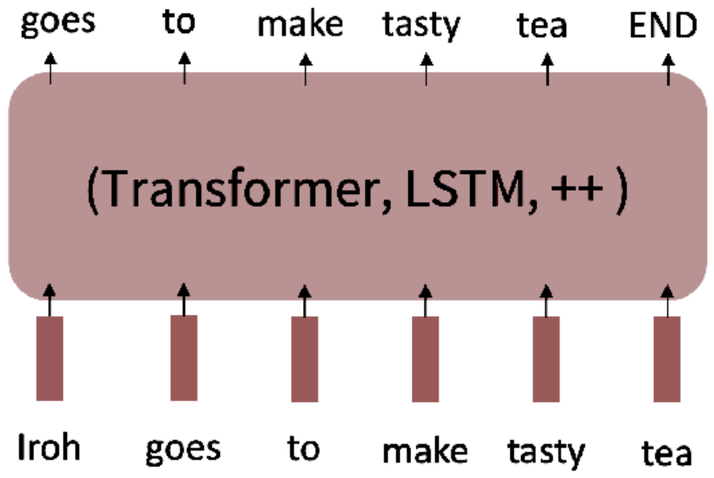
Yao Fu et al. How does GPT Obtain its Ability?

Fine-tuning LLM

Pretraining can improve NLP applications by serving as parameter initialization.

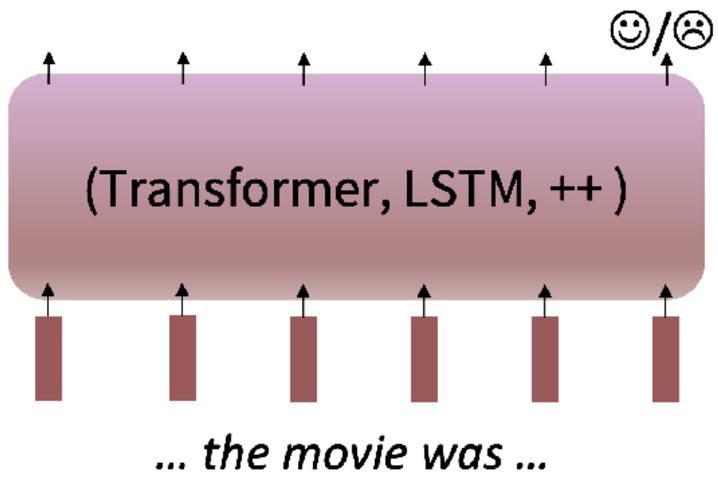
Step 1: Pretrain (on language modeling)

Lots of text; learn general things!



Step 2: Finetune (on your task)

Not many labels; adapt to the task!



Fine-tuning LLM

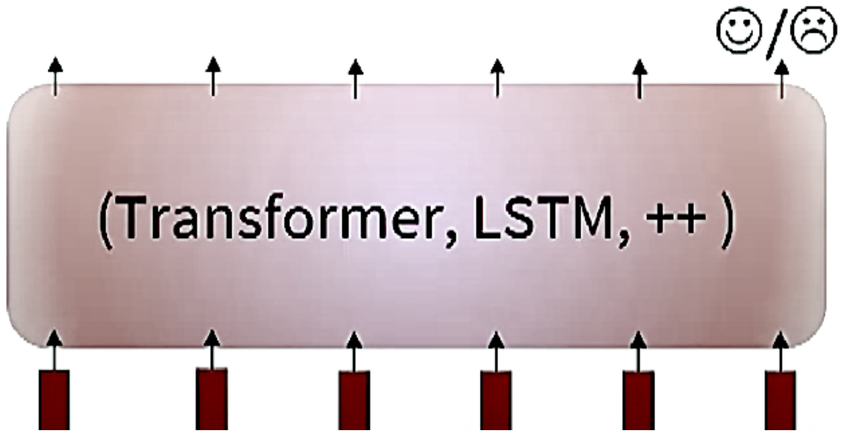
Finetuning every parameter in a pretrained model works well, but is memory-intensive.

But **lightweight** finetuning methods adapt pretrained models in a constrained way.

Leads to **less overfitting** and/or **more efficient finetuning and inference.**

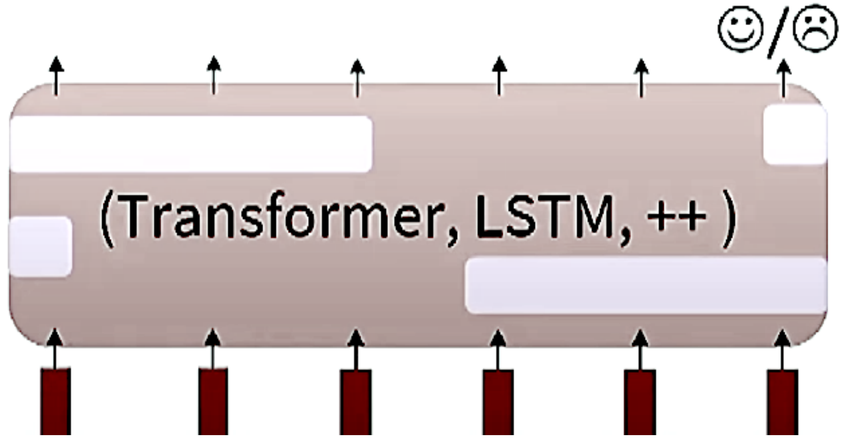
Full Finetuning

Adapt all parameters



Lightweight Finetuning

Train a few existing or new parameters



Fine-tuning LLM