

Decoding, Pretrained Models, and Finetuning

CMSC 473/673 - NATURAL LANGUAGE PROCESSING

Learning Objectives

Consider when to use various sampling algorithms

Discuss the uses of finetuning

Differentiate between encoder model embeddings and older dense embeddings

Recognize useful encoder-only, encoder-decoder, and decoder-only models


Limitations of Recurrent architecture

Slow to train.

- Can't be easily parallelized.
- The computation at position t is dependent on first doing the computation at position $t-1$.

Difficult to access information from many steps back.

- If two tokens are K positions apart, there are K opportunities for knowledge of the first token to be erased from the hidden state before a prediction is made at the position of the second token.



Mostly fixed with
Transformer
architecture!

Review: Generating Text

Also sometimes called decoding



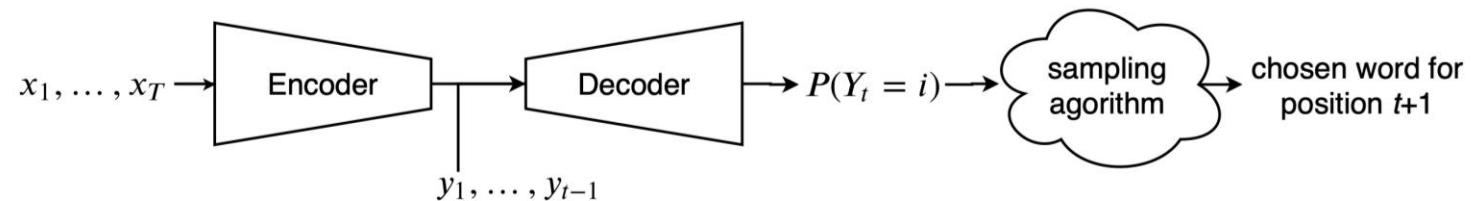
To generate text, we need an algorithm that selects tokens given the predicted probability distributions.

Examples:

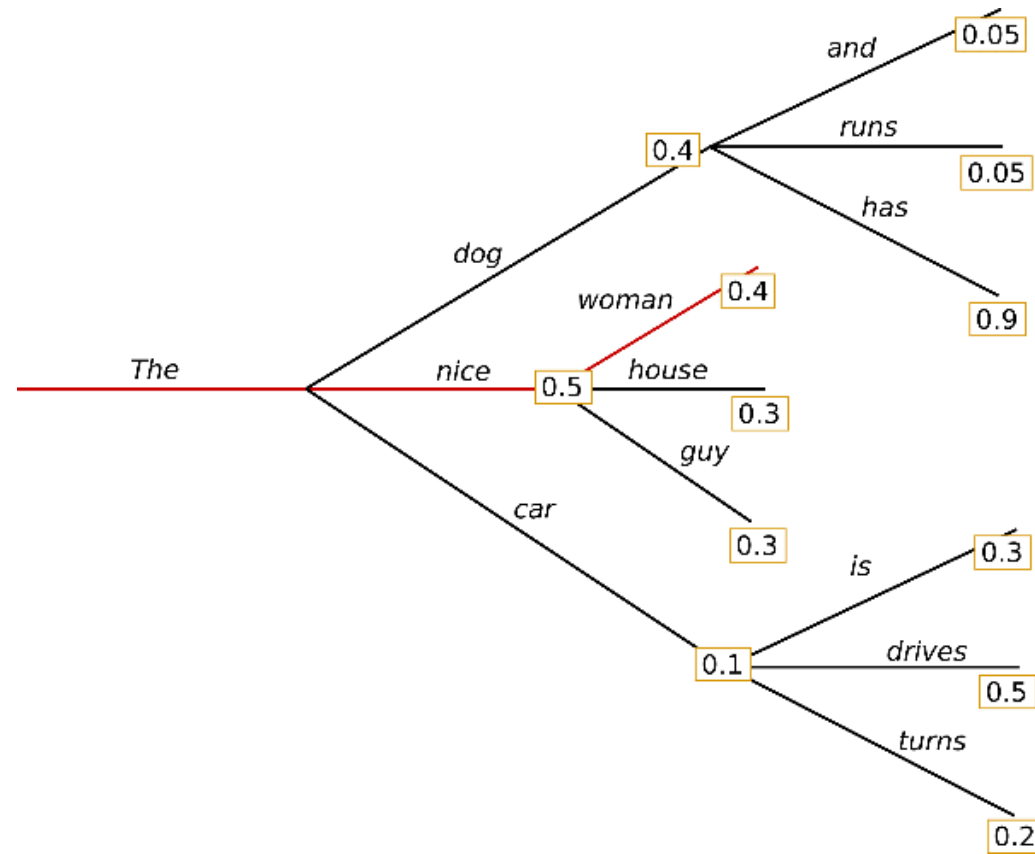
Argmax

Beam search

Random sampling



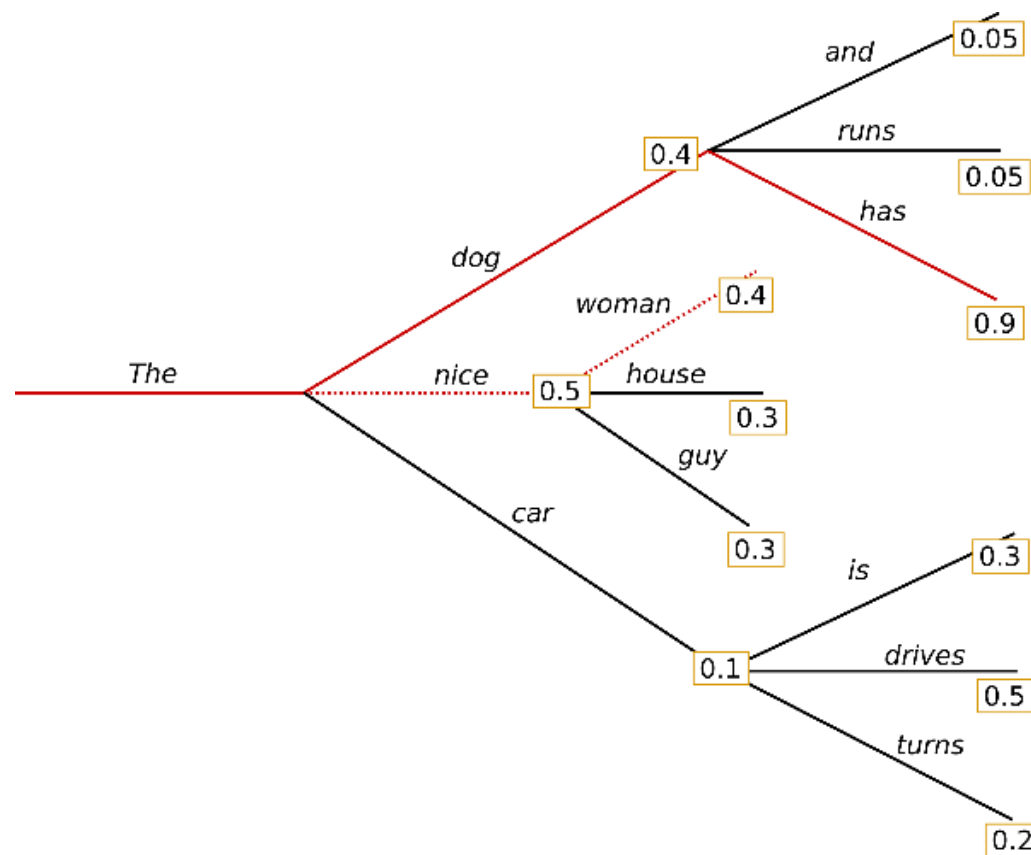
Greedy Search (Argmax)



<https://huggingface.co/blog/how-to-generate>

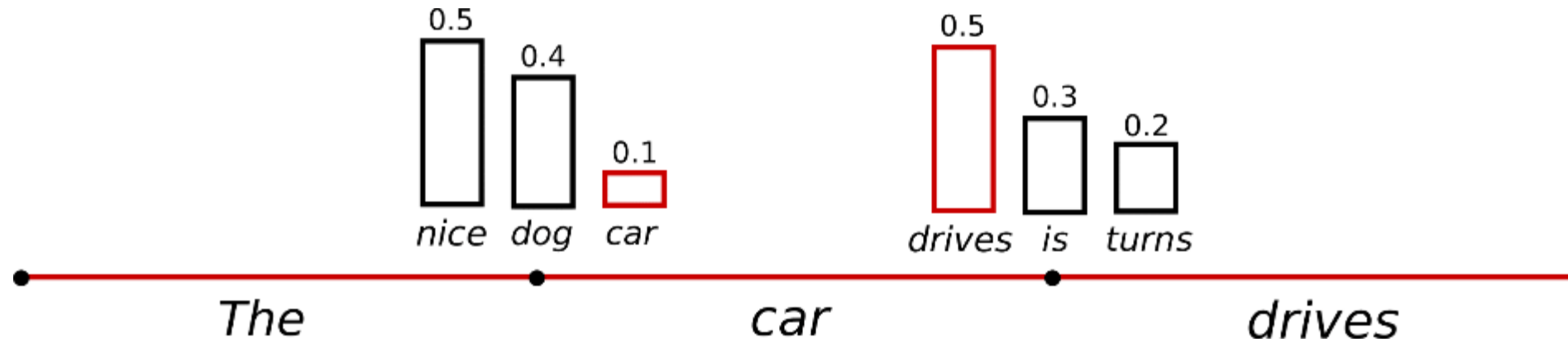
Beam Search

Number of
beams = 2



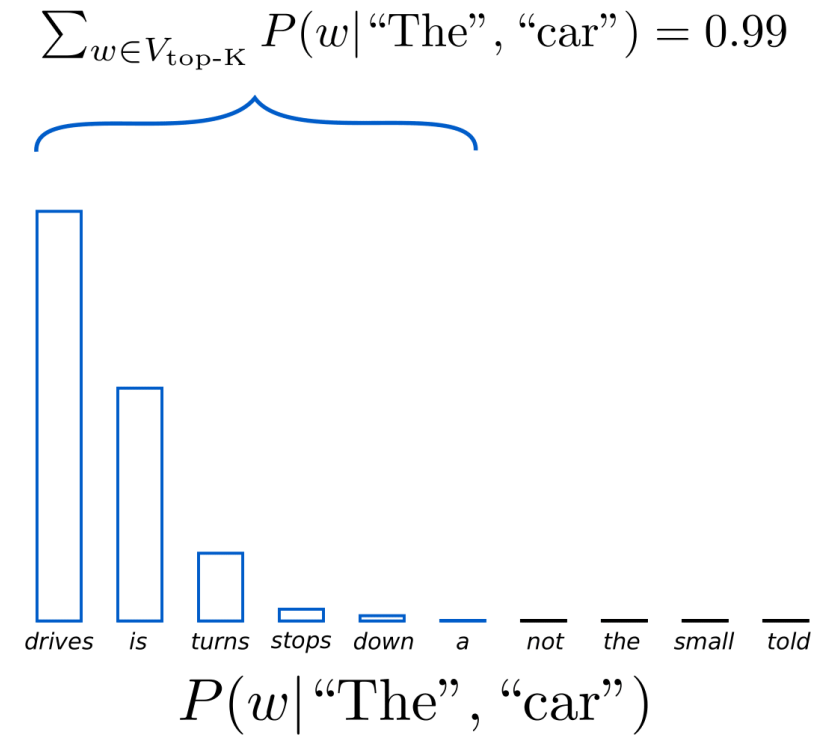
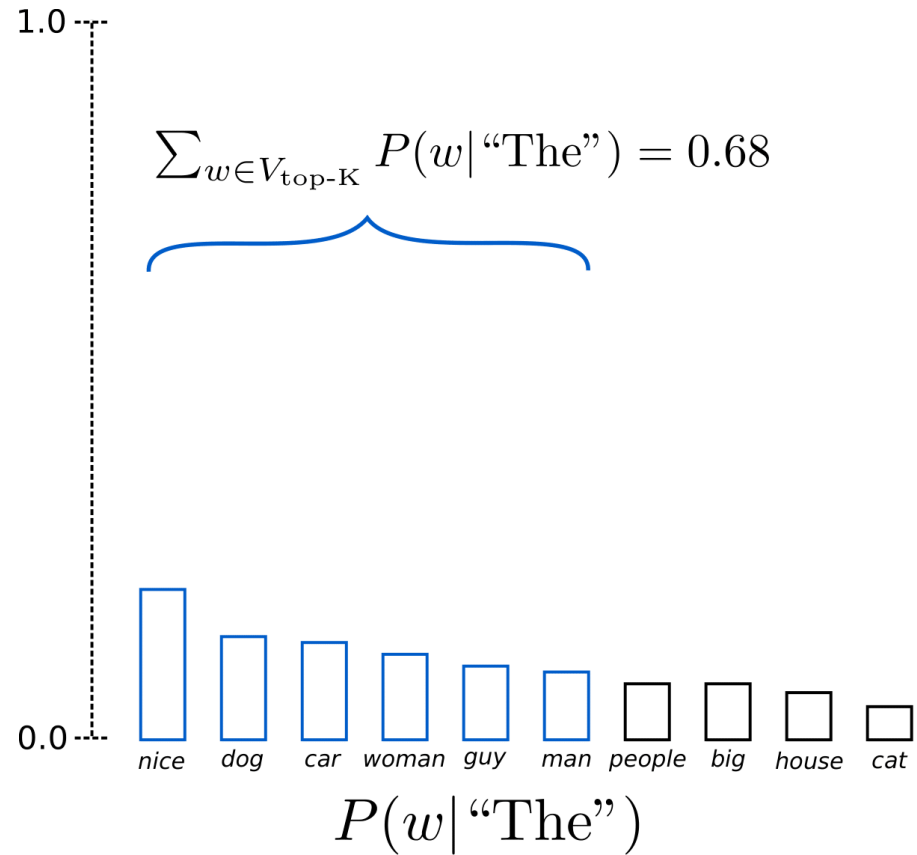
<https://huggingface.co/blog/how-to-generate>

Random Sampling



<https://huggingface.co/blog/how-to-generate>

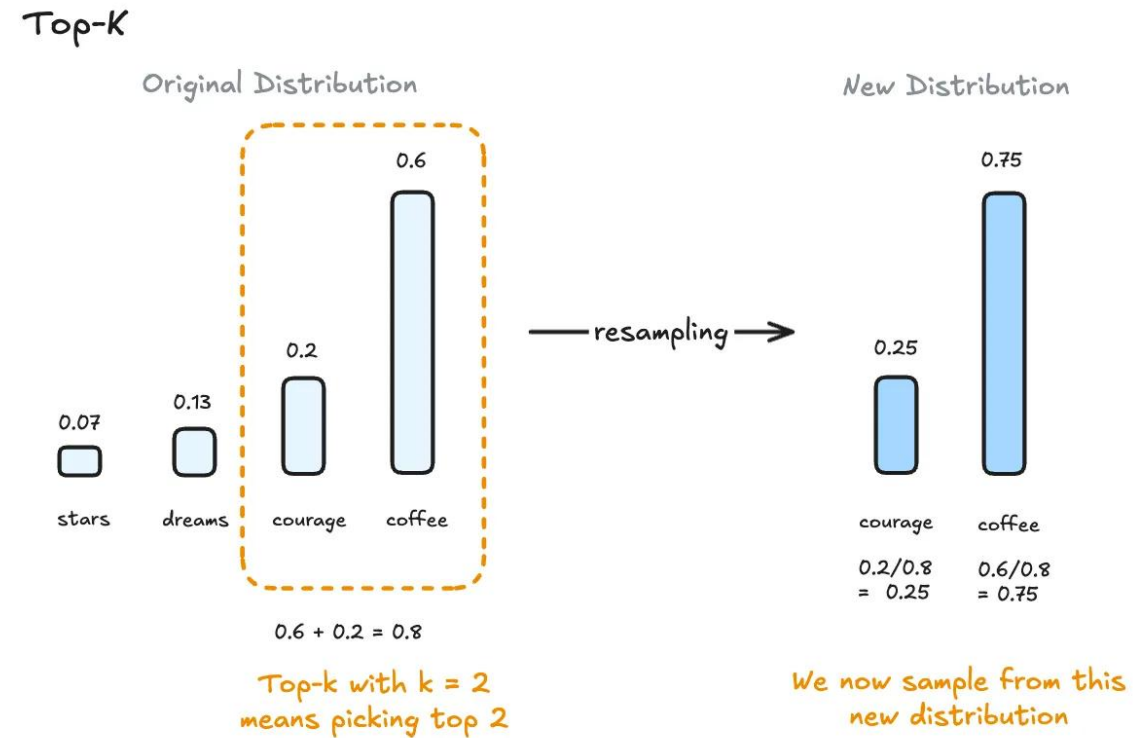
Top-K Sampling



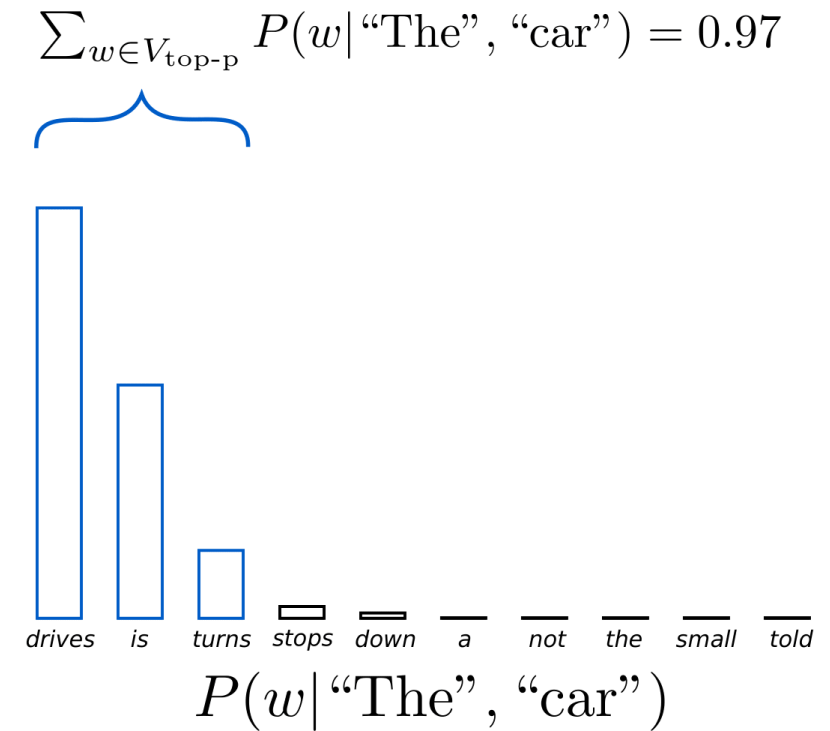
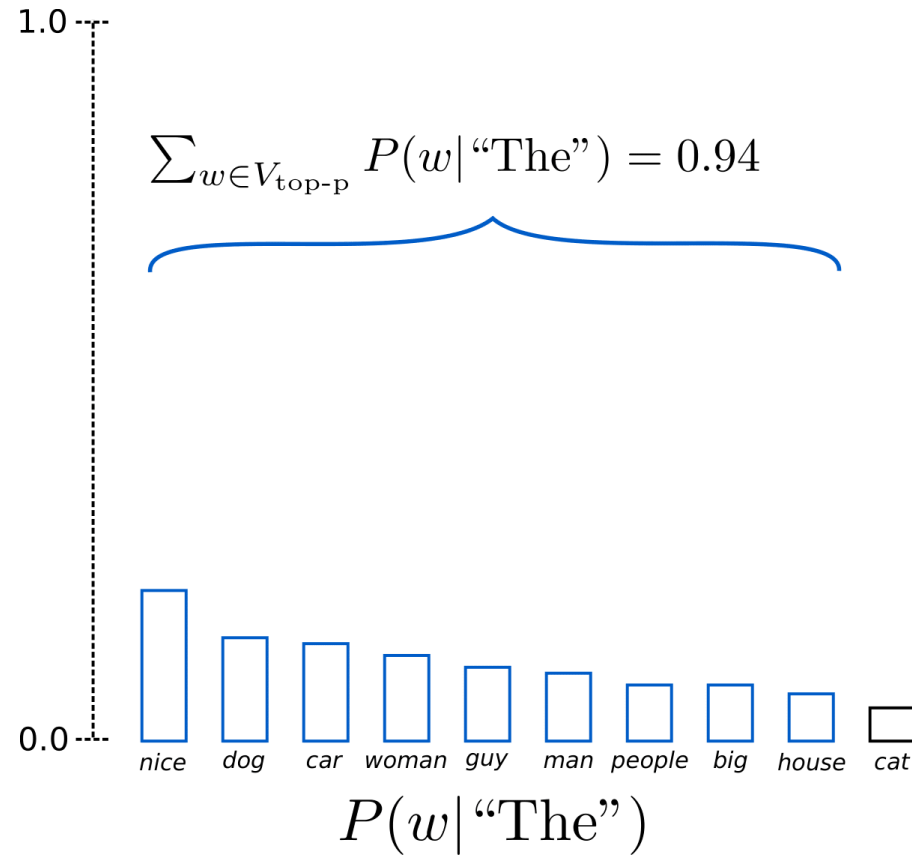
A. Holtzman, J. Buys, M. Forbes, and Y. Choi, "The Curious Case of Neural Text Degeneration," in *International Conference on Learning Representations (ICLR)*, 2020, p. 16.
<https://openreview.net/forum?id=rygGQyrFvH>

<https://huggingface.co/blog/how-to-generate>

Resampling

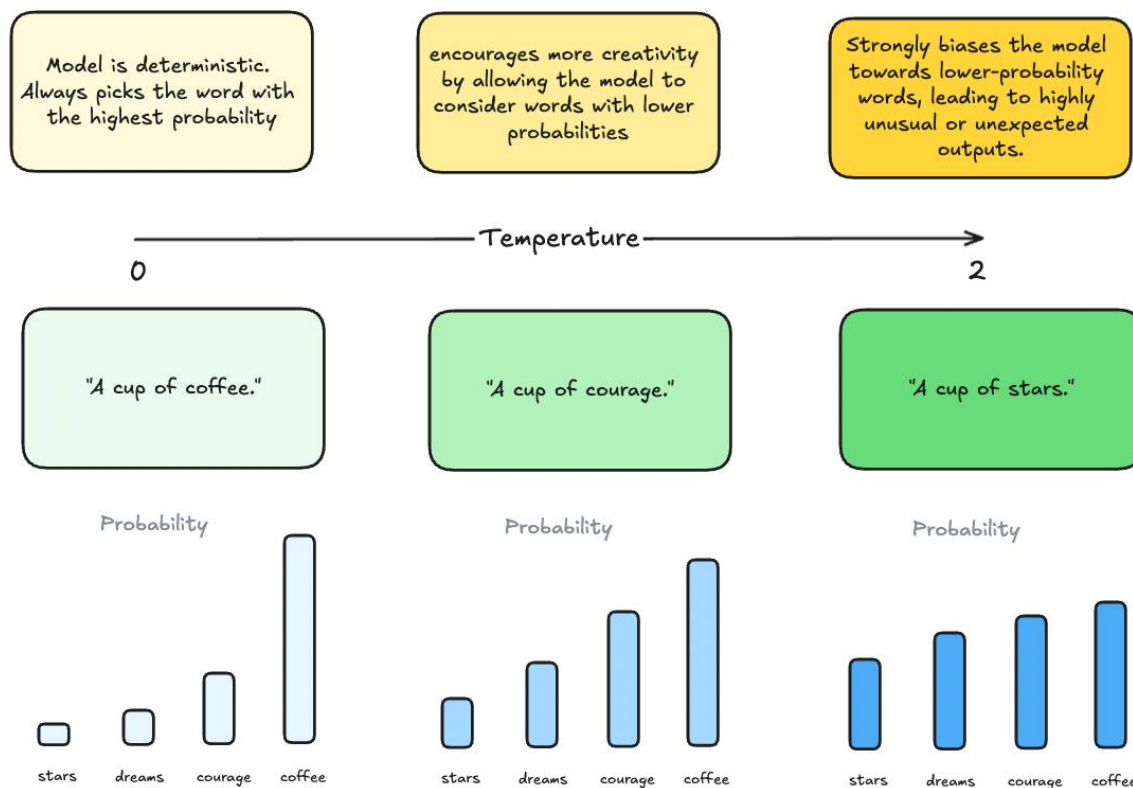


Top-P Sampling



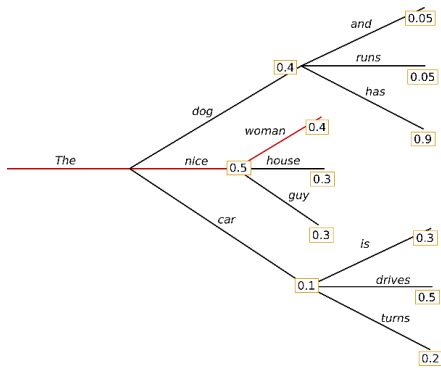
<https://huggingface.co/blog/how-to-generate>

“Temperature”

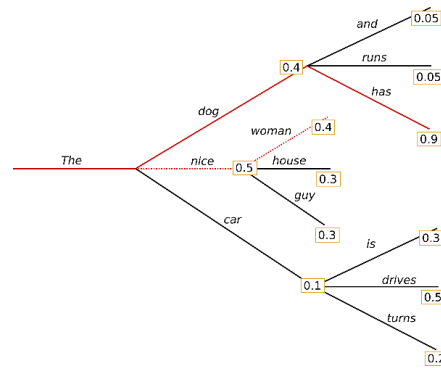


Think-Pair-Share

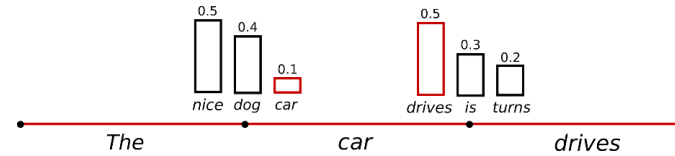
When might you want to use one sampling algorithm over the other?



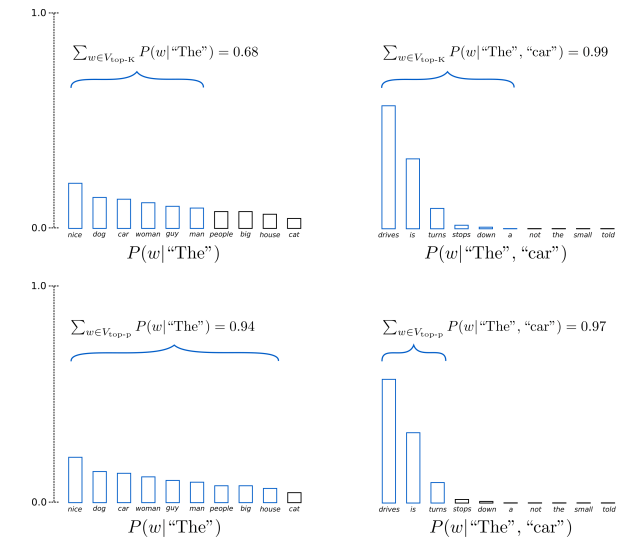
Greedy



Beam
Search



Random
Sampling

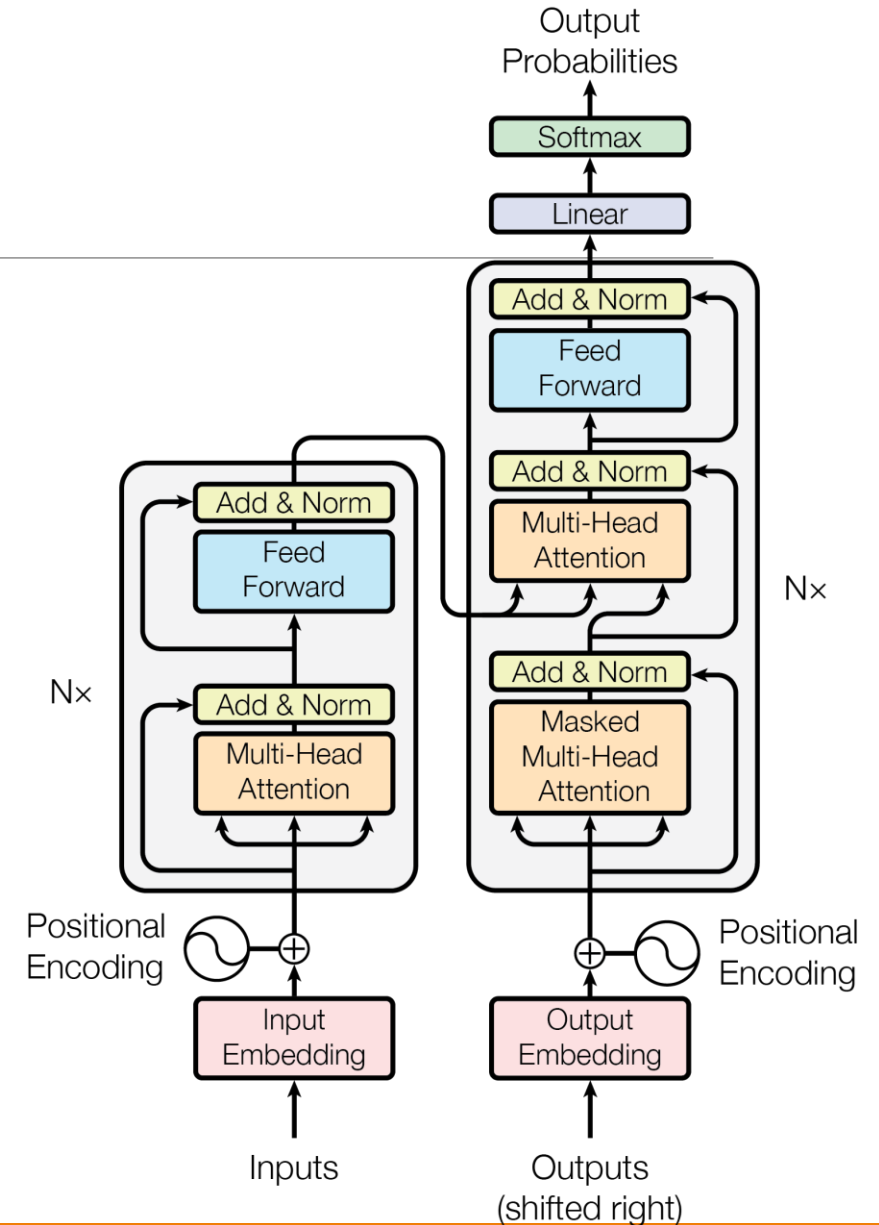


Top-K/P

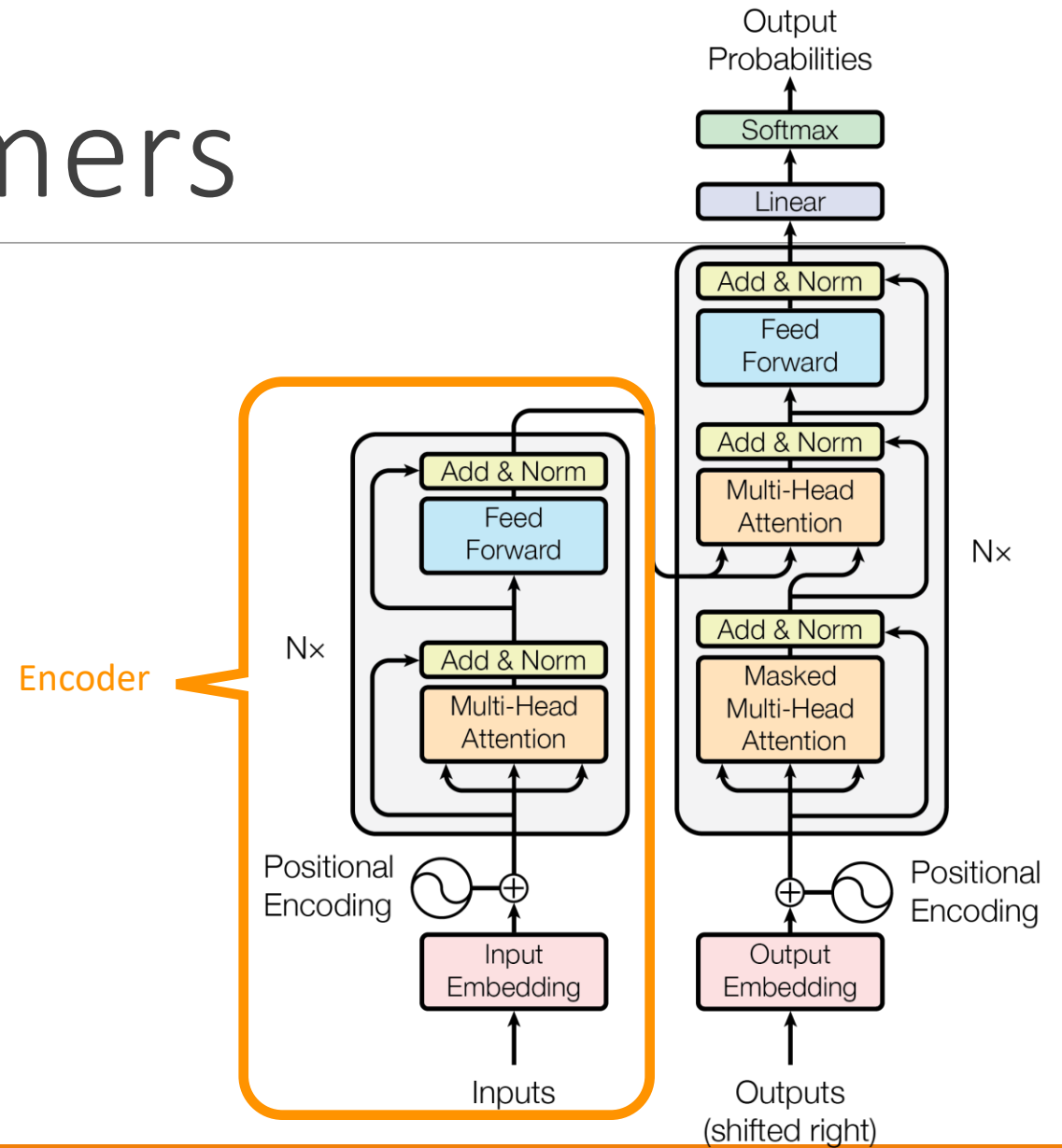
Review: Transformers

The Transformer is a **non-recurrent** non-convolutional (feed-forward) neural network designed for language understanding

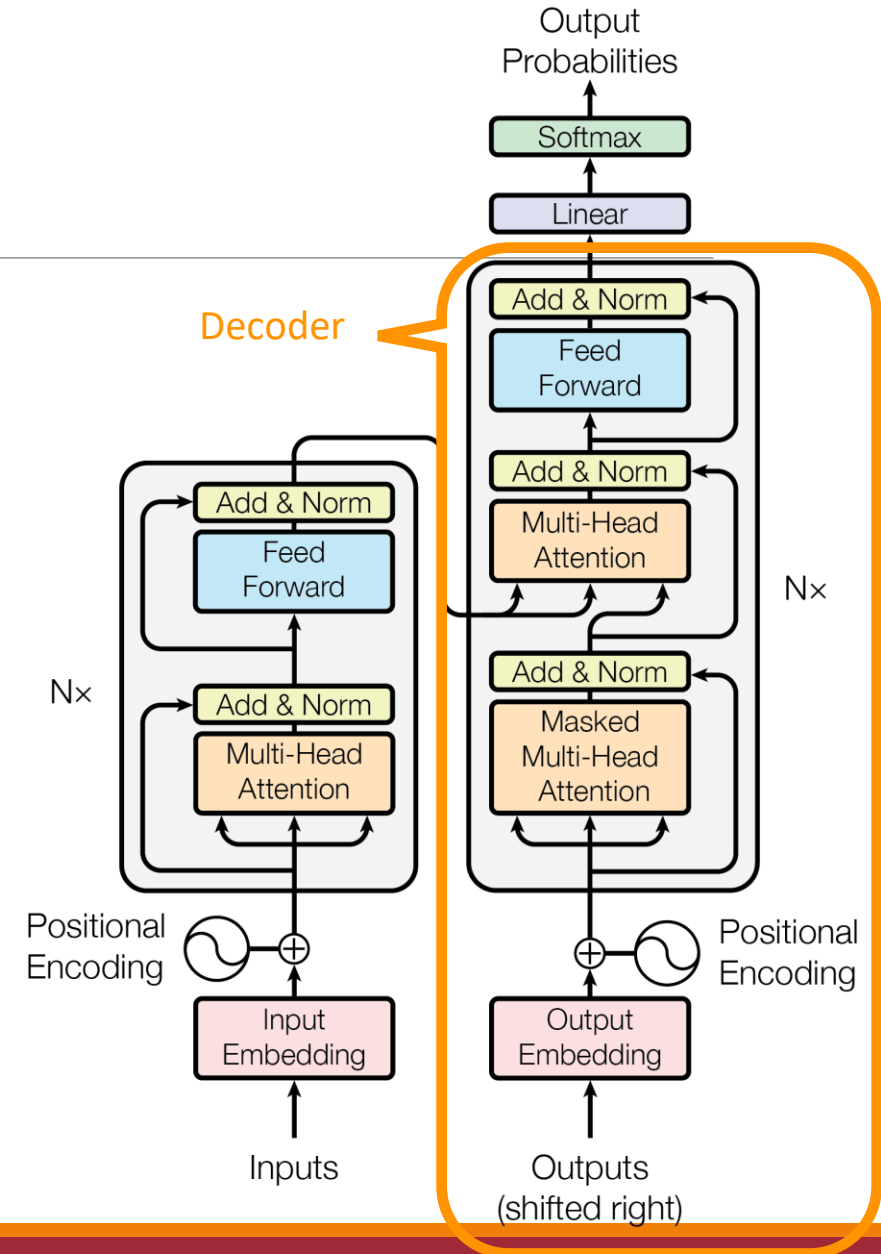
- introduces self-attention in addition to encoder-decoder attention



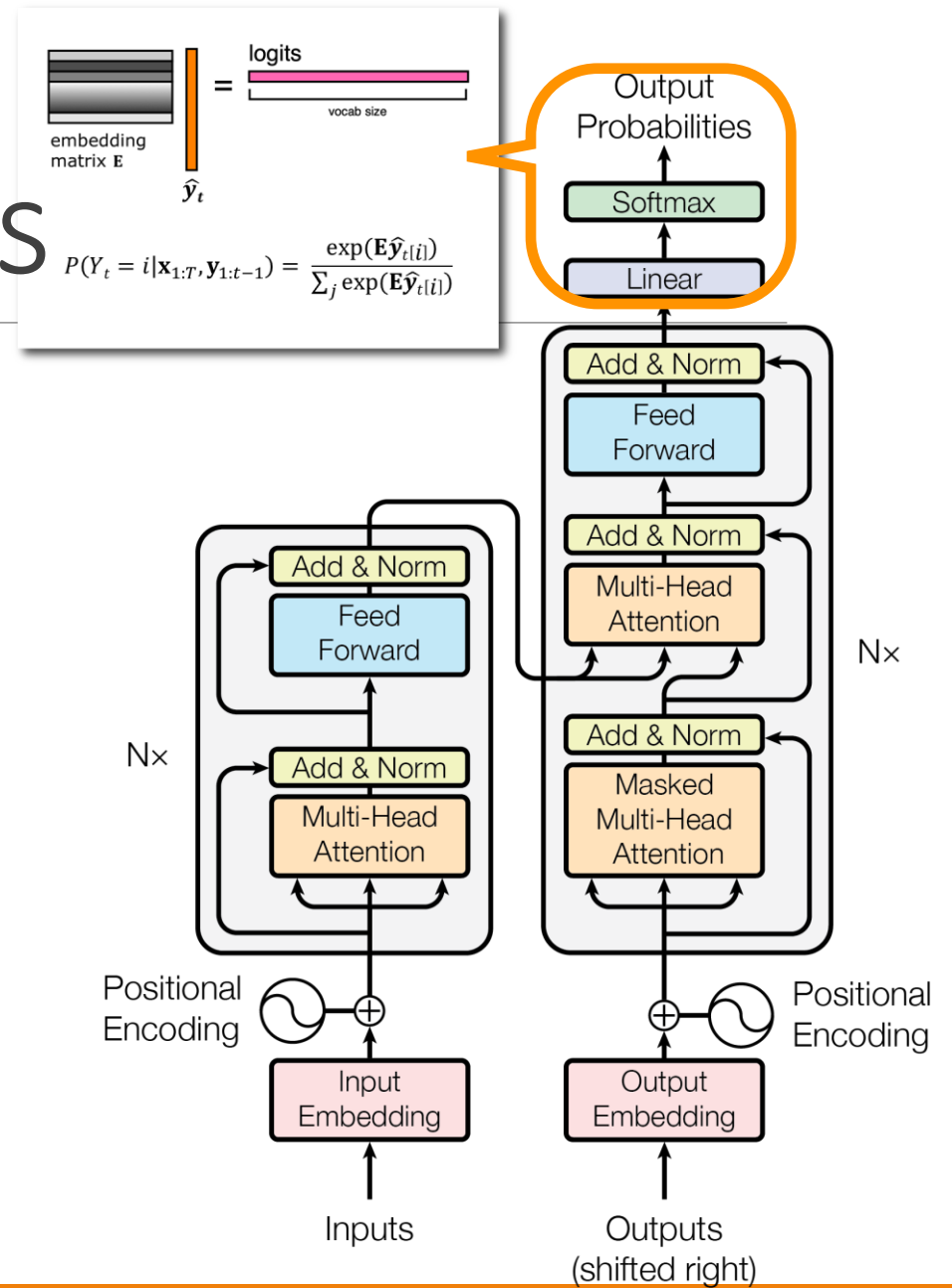
Review: Transformers



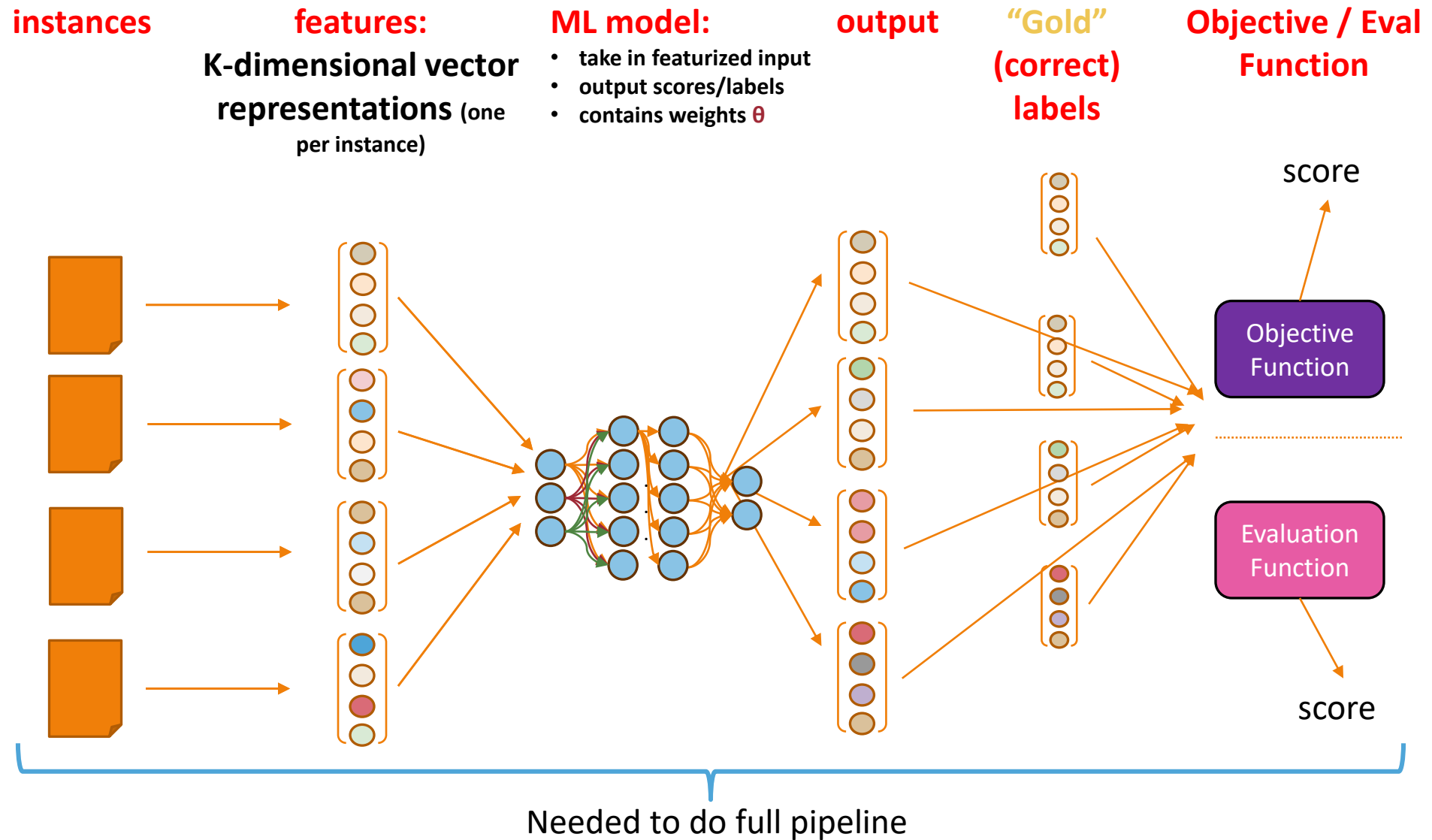
Review: Transformers



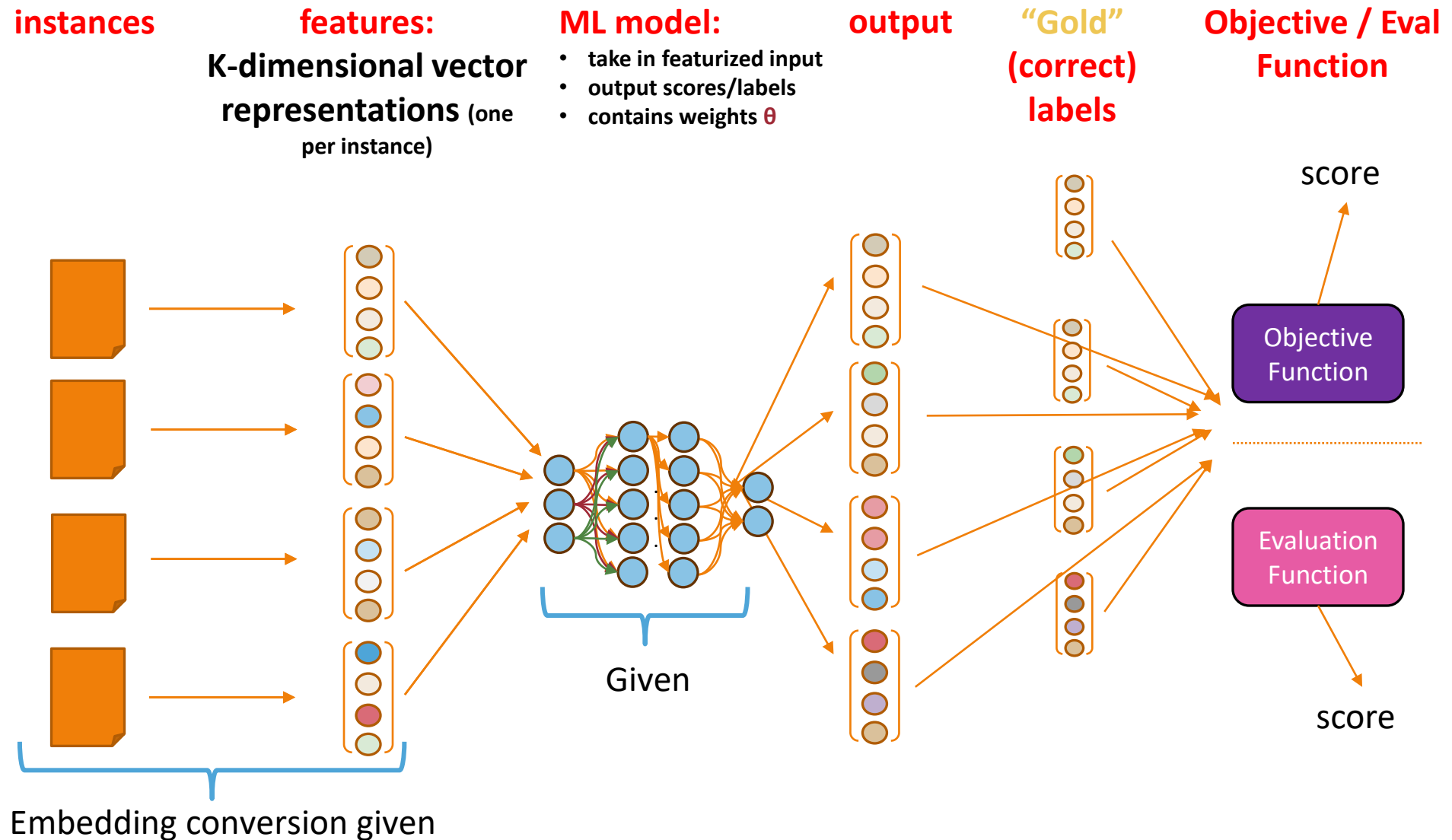
Review: Transformers



Pre-transformer Neural NLP



Transformer-based NLP



Fine-tuning

Start with pre-trained model

Freeze the model (don't touch it) except for the last layer

- Start with generalized “foundational” model
- Train on a new, small dataset for your specific task

GPT-2

Language Models are Unsupervised Multitask Learners

Alec Radford ^{*1} Jeffrey Wu ^{*1} Rewon Child ¹ David Luan ¹ Dario Amodei ^{**1} Ilya Sutskever ^{**1}

Abstract

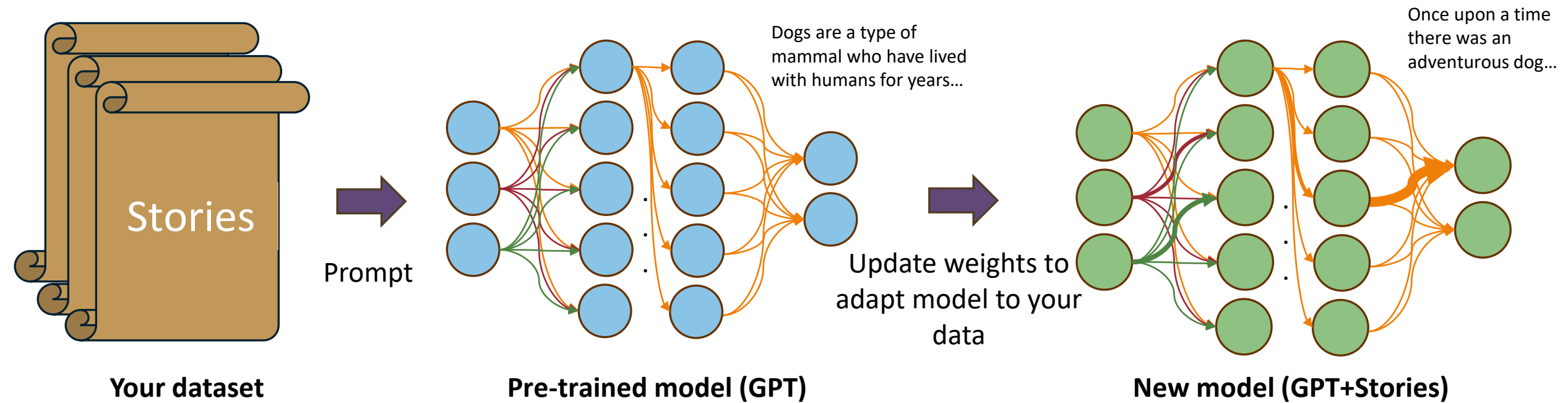
Natural language processing tasks, such as question answering, machine translation, reading comprehension, and summarization, are typically approached with supervised learning on task-specific datasets. We demonstrate that language models begin to learn these tasks without any explicit supervision when trained on a new dataset of millions of webpages called WebText. When conditioned on a document plus questions, the answers generated by the language model reach 55 F1 on the CoQA dataset - matching or exceeding the performance of 3 out of 4 baseline systems without using the 127,000+ training examples. The capacity of the language model is essential to the success of zero-shot task transfer and increasing it improves performance in a log-linear fashion across tasks. Our largest model, GPT-2, is a 1.5B parameter Transformer that achieves state of the art results on 7 out of 8 tested language modeling datasets in a zero-shot setting but still underfits WebText. Samples from the

competent generalists. We would like to move towards more general systems which can perform many tasks – eventually without the need to manually create and label a training dataset for each one.

The dominant approach to creating ML systems is to collect a dataset of training examples demonstrating correct behavior for a desired task, train a system to imitate these behaviors, and then test its performance on independent and identically distributed (IID) held-out examples. This has served well to make progress on narrow experts. But the often erratic behavior of captioning models (Lake et al., 2017), reading comprehension systems (Jia & Liang, 2017), and image classifiers (Alcorn et al., 2018) on the diversity and variety of possible inputs highlights some of the shortcomings of this approach.

Our suspicion is that the prevalence of single task training on single domain datasets is a major contributor to the lack of generalization observed in current systems. Progress towards robust systems with current architectures is likely to require training and measuring performance on a wide range of domains and tasks. Recently, several benchmarks

Finetuning



What types of things can go wrong with finetuning?

Underfitting – finetuning data is too different from what the foundational model was trained on → model can't learn it

Overfitting – overwrites what the model learned originally

Pre-trained models

Most LLMs people use today are pre-trained models

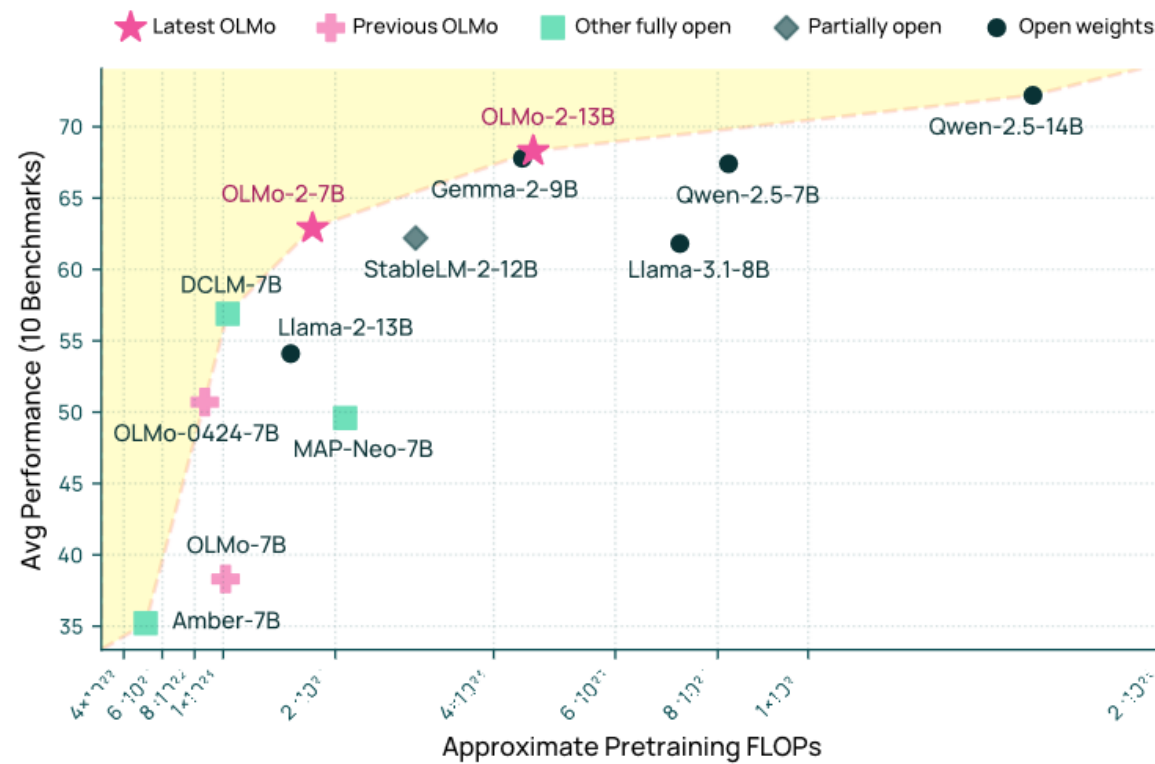
Trained on “the Internet” → Impossible to know all of what it’s train on

- Very few models release all the data. One example is OLMo 2.

Can then be finetuned on specific data

Why would you
want to “tweak” an
existing model?

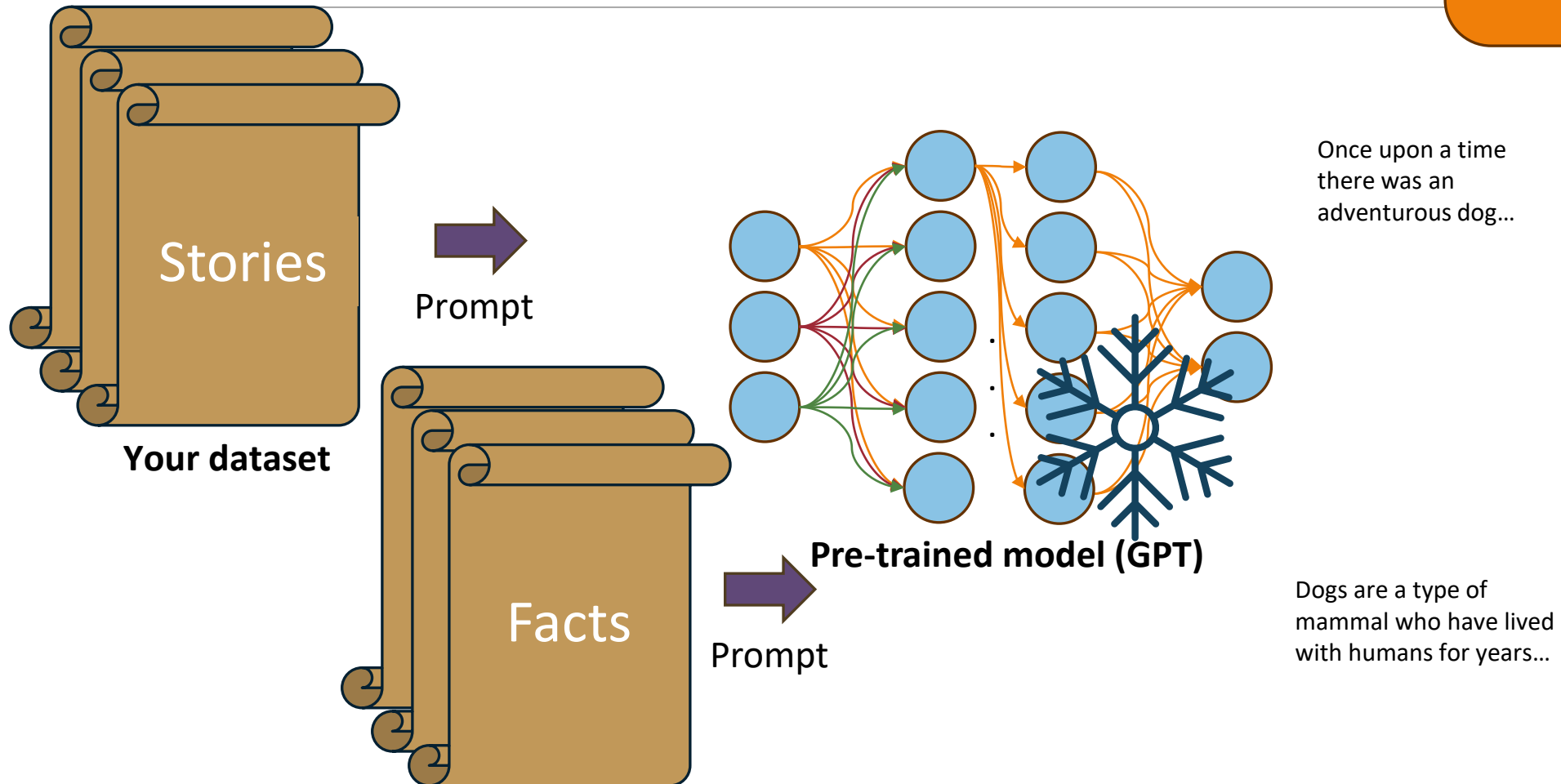
Open-Sourced Models



OLMo, T., Walsh, P., Soldaini, L., Groeneveld, D., Lo, K., Arora, S., Bhagia, A., Gu, Y., Huang, S., Jordan, M., Lambert, N., Schwenk, D., Tafjord, O., Anderson, T., Atkinson, D., Brahman, F., Clark, C., Dasigi, P., Dziri, N., ... Hajishirzi, H. (2024). *2 OLMo 2 Furious* (No. 2501.00656). arXiv. <https://doi.org/10.48550/arXiv.2501.00656>

Prompting

We'll talk about
this in a future
lecture



Types of Foundation Models

Encoder Only

Decoder Only

Encoder-Decoder Models

What is a foundation model?

A model that captures “foundation” or core information about a modality (e.g., text, speech, images)

Pretrained on a large amount of data & able to *be* finetuned on a particular task

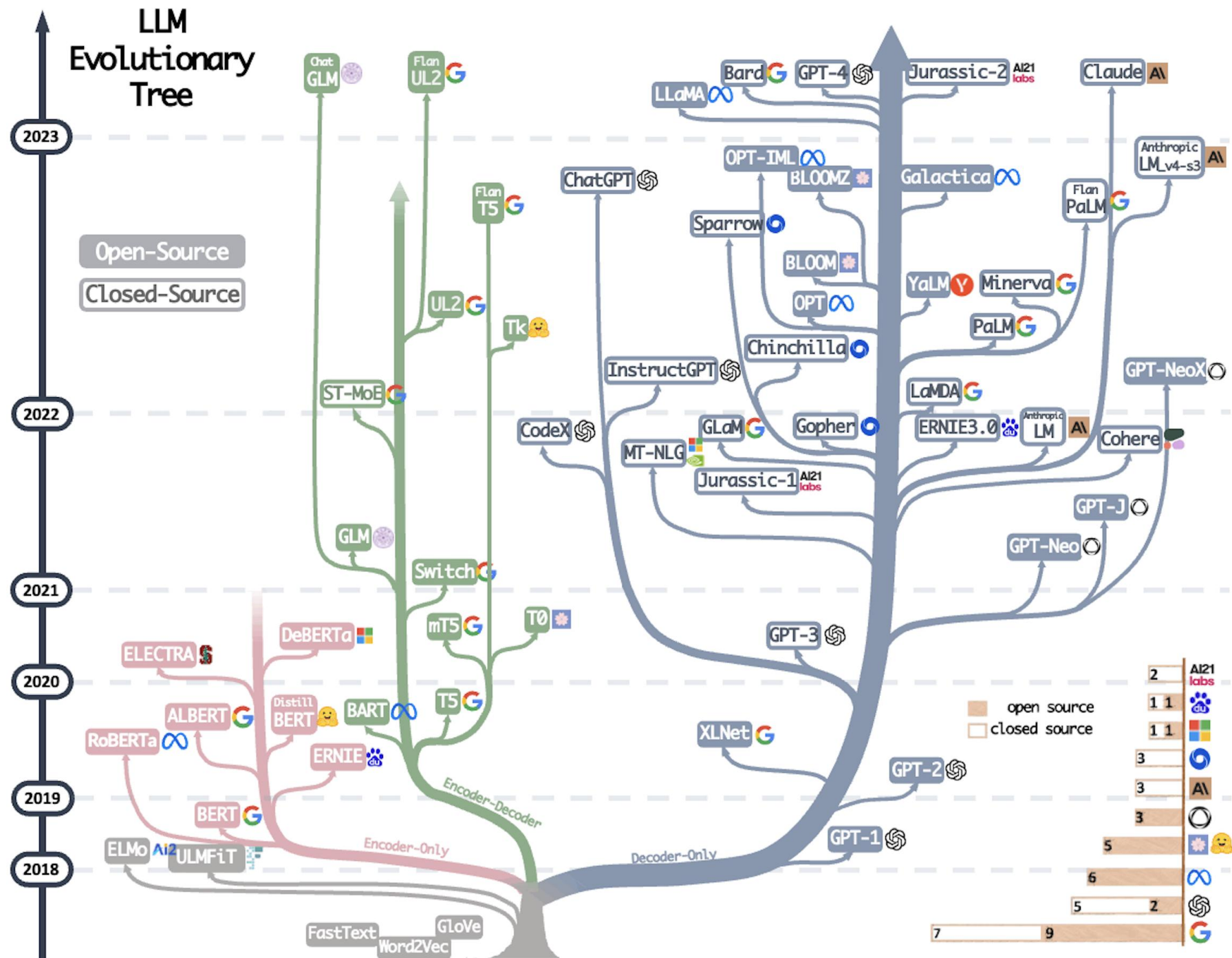
Self-supervised

All non-finetuned large language models (LLMs) are foundation models

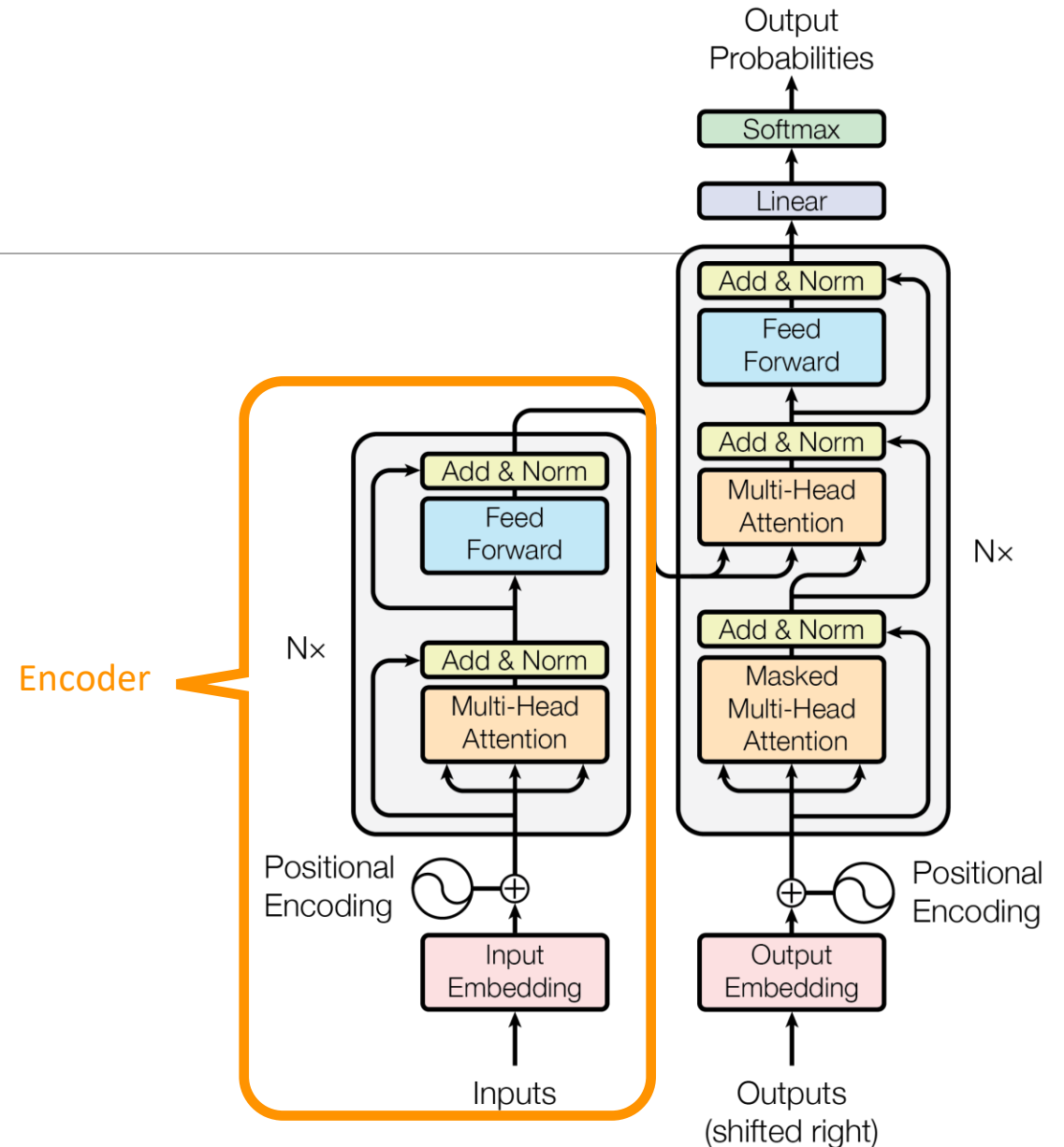
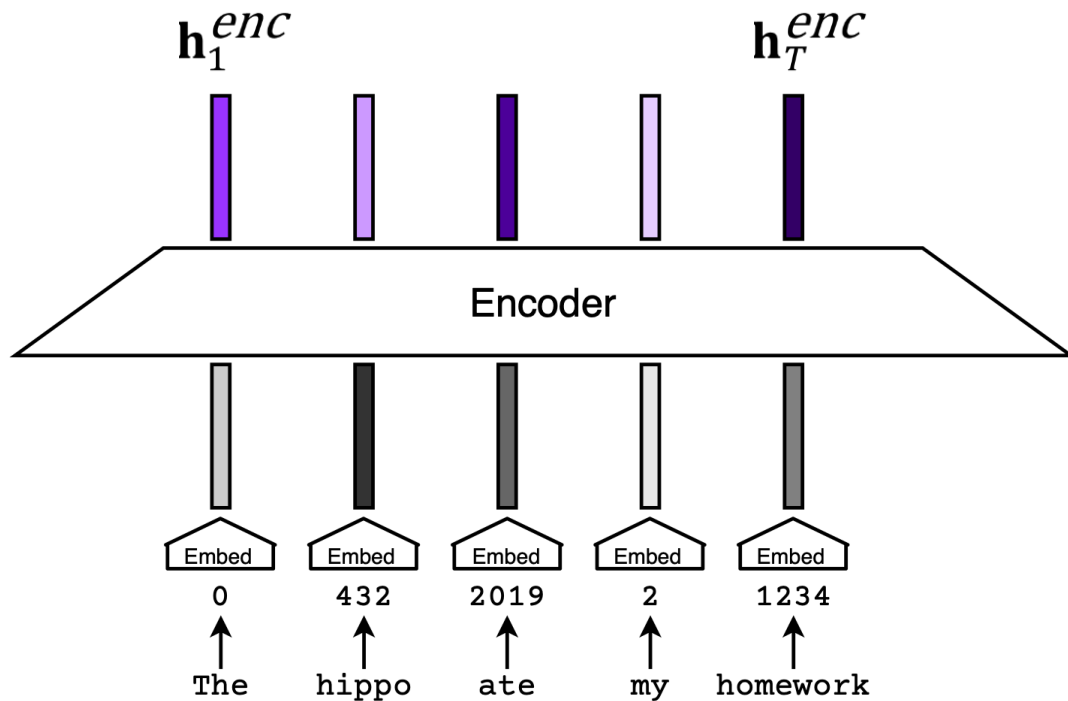
Some Models Come Fine-tuned

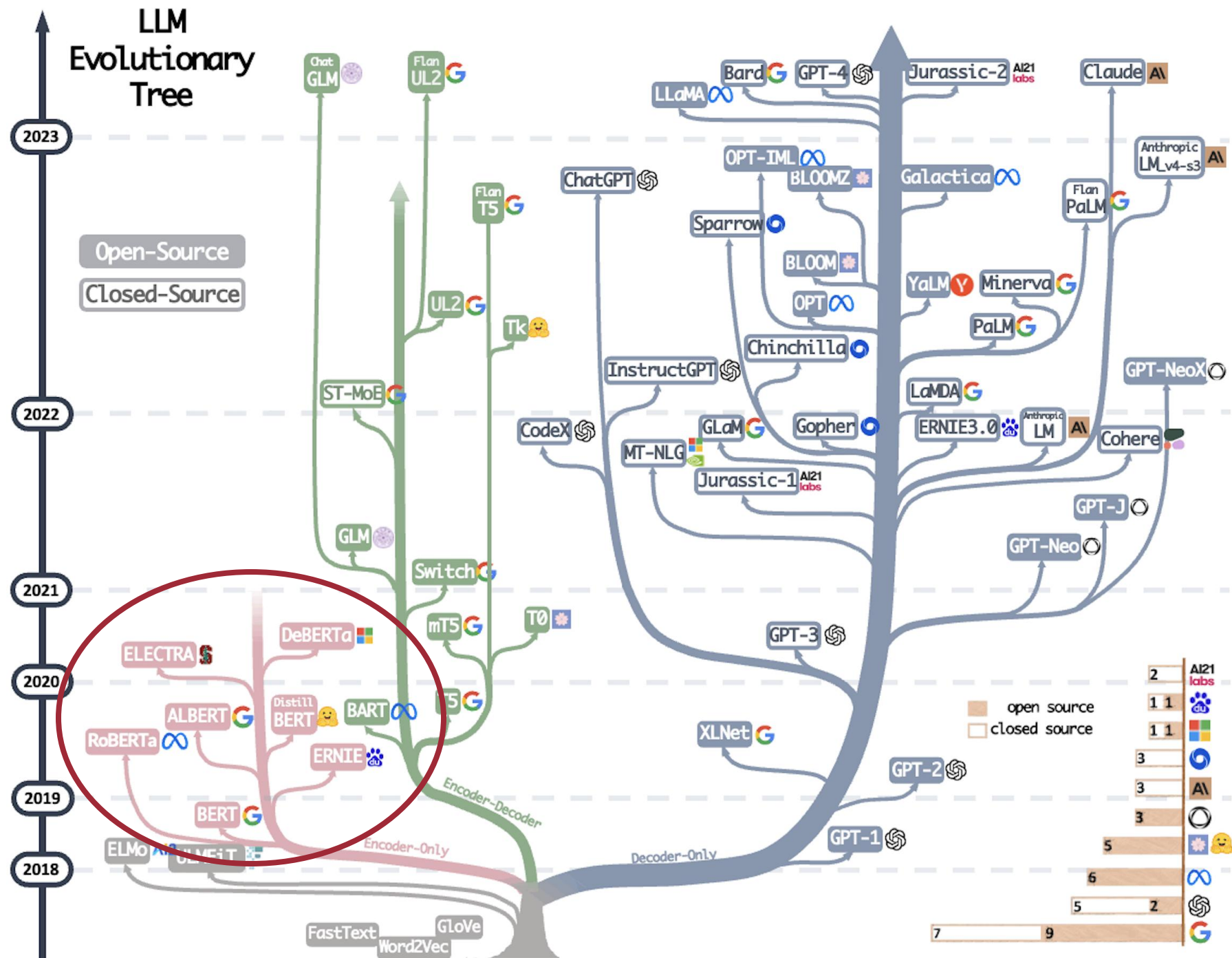
ChatGPT/InstructGPT

Most/all “Instruct” or “Chat” models



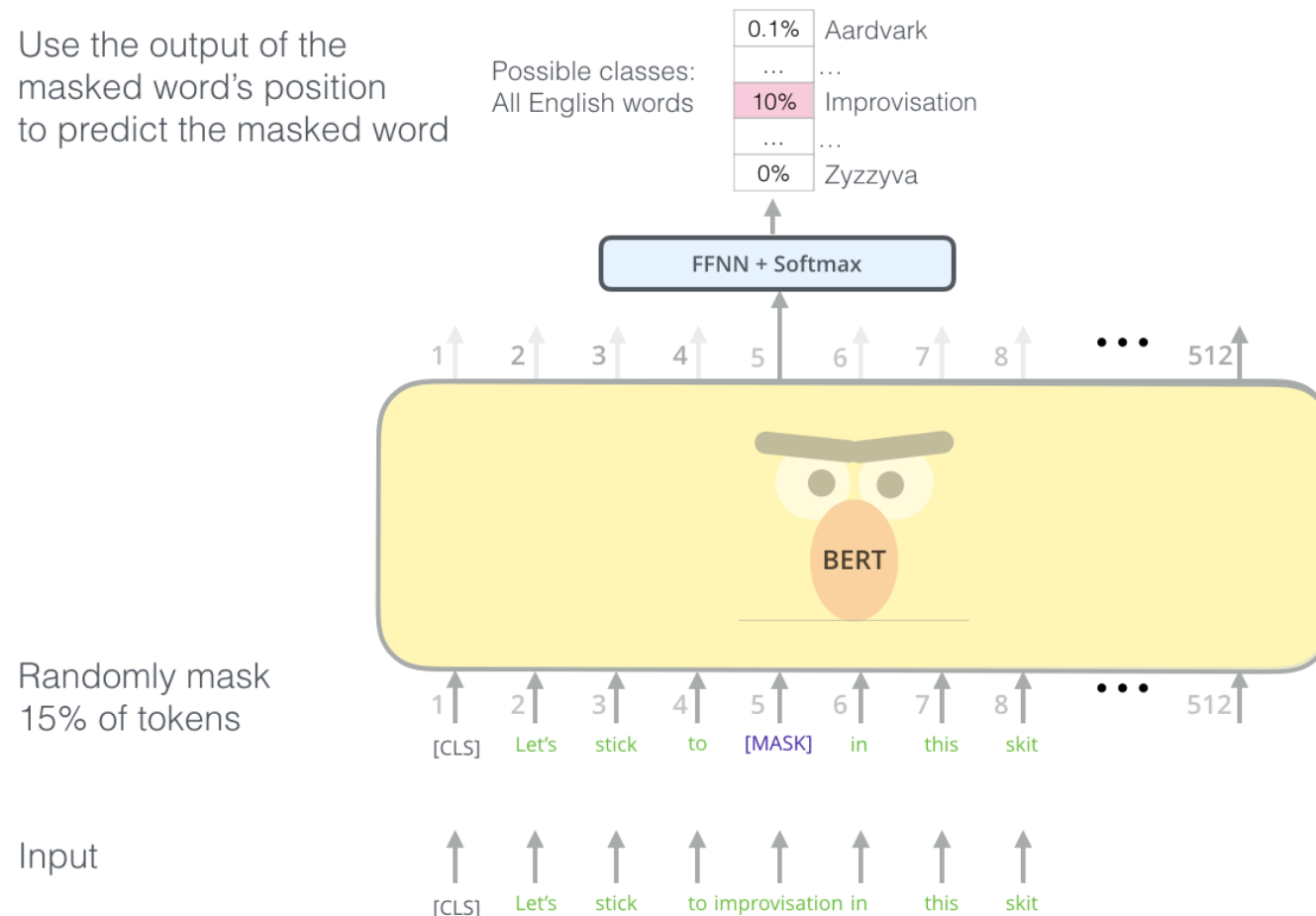
Encoder-only models





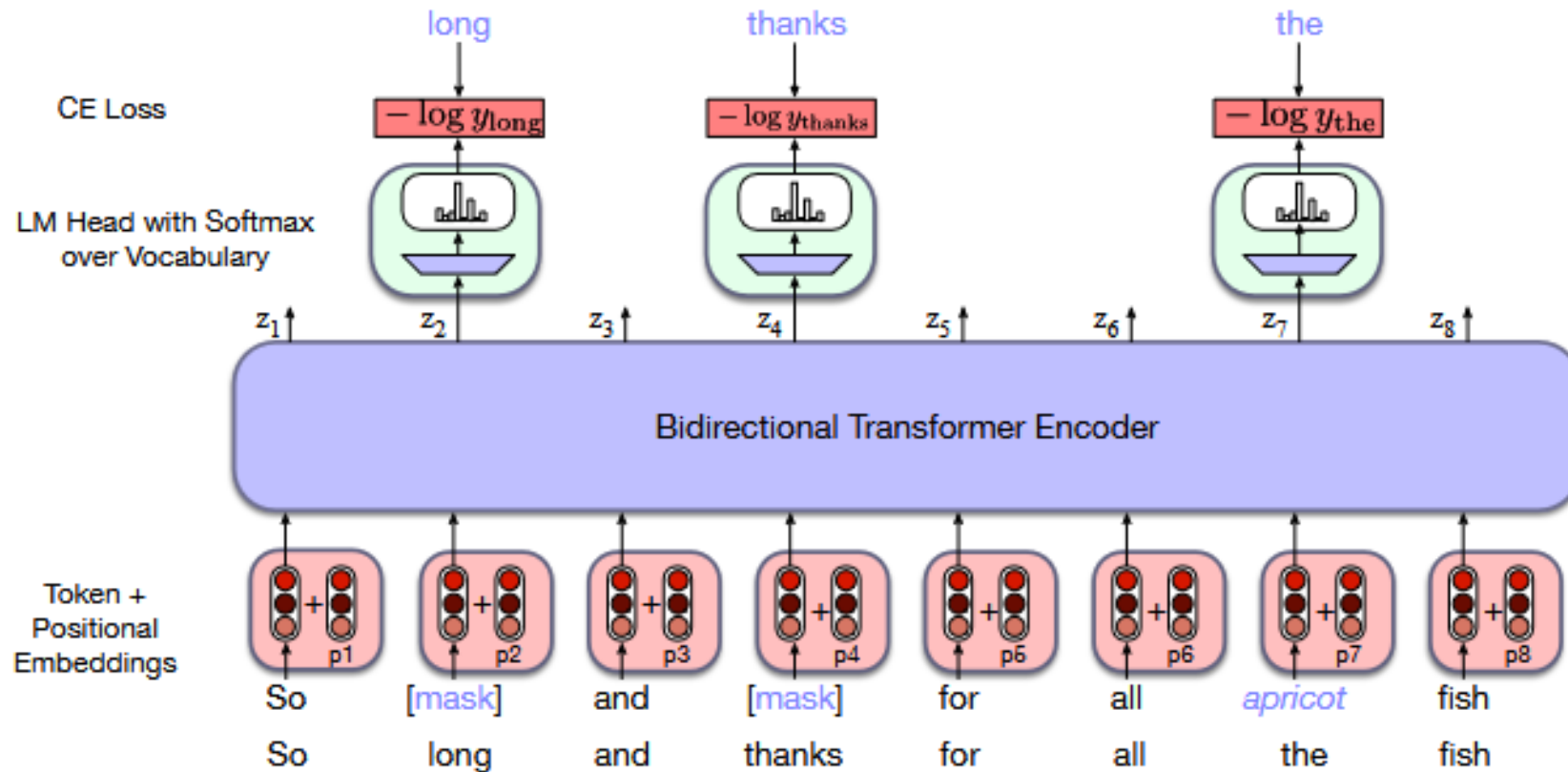
BERT (Devlin et al. 2019)

Use the output of the masked word's position to predict the masked word



<http://jalammar.github.io/illustrated-bert/>

Masked Language Models



From Jurafsky & Martin's *Speech and Language Processing*, 3rd Edition, Chapter 11

Contextual Embeddings



From Jurafsky & Martin's *Speech and Language Processing*, 3rd Edition, Chapter 11

Uses of Encoder-Only Models

Classification tasks

Sentence embeddings

Context-dependent word embeddings

Any type of fill-in-the-blank tasks

BERT Question

Consider the highlighted words. Which two words would contextual word embeddings from BERT say are closest?

- A. I am so excited to use my new bat at the baseball game tomorrow.
- B. The favorite food of this species of bat is mosquitoes.
- C. The cardinal isn't just a lawn decoration; the species makes themselves useful by eating mosquitoes.

PollEv.com/laramartin527



Remember: word2vec is a dense vector embedding

Word2Vec Question

Consider the highlighted words. Which two words would word2vec say are closest?

- A. I am so excited to use my new bat at the baseball game tomorrow.
- B. The favorite food of this species of bat is mosquitoes.
- C. The cardinal isn't just a lawn decoration; the species makes themselves useful by eating mosquitoes.

PollEv.com/laramartin527



BERT Family of Models

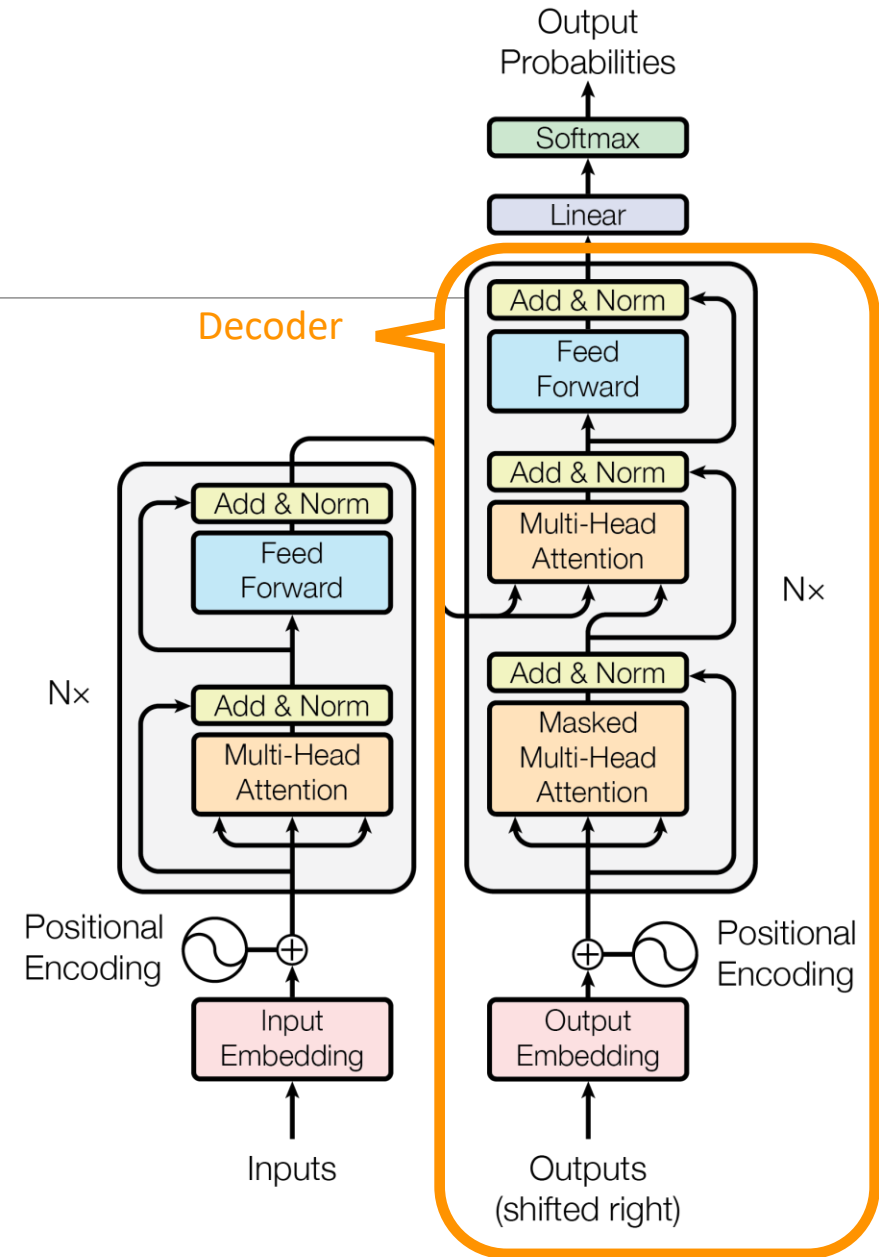
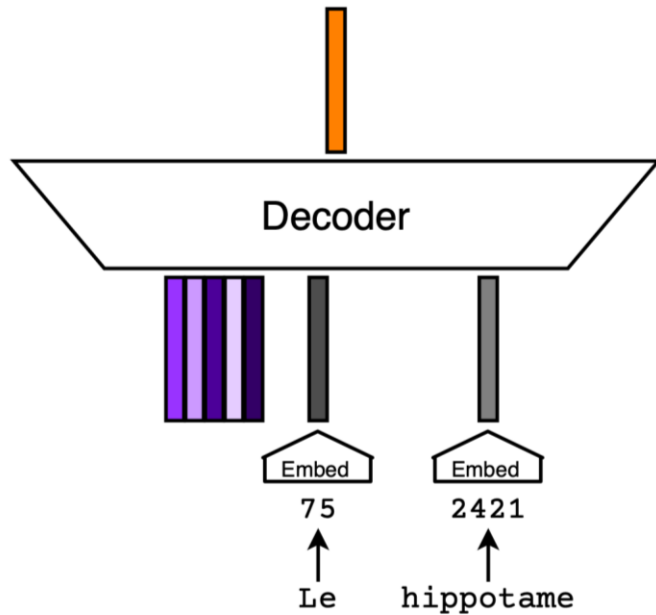
- Encoder-only
 - Input: Corrupted version of text sequence
 - Goal: Produce an uncorrupted version of text sequence
- How to use:
 - Finetune for a classification task
 - Extract word/sentence embeddings

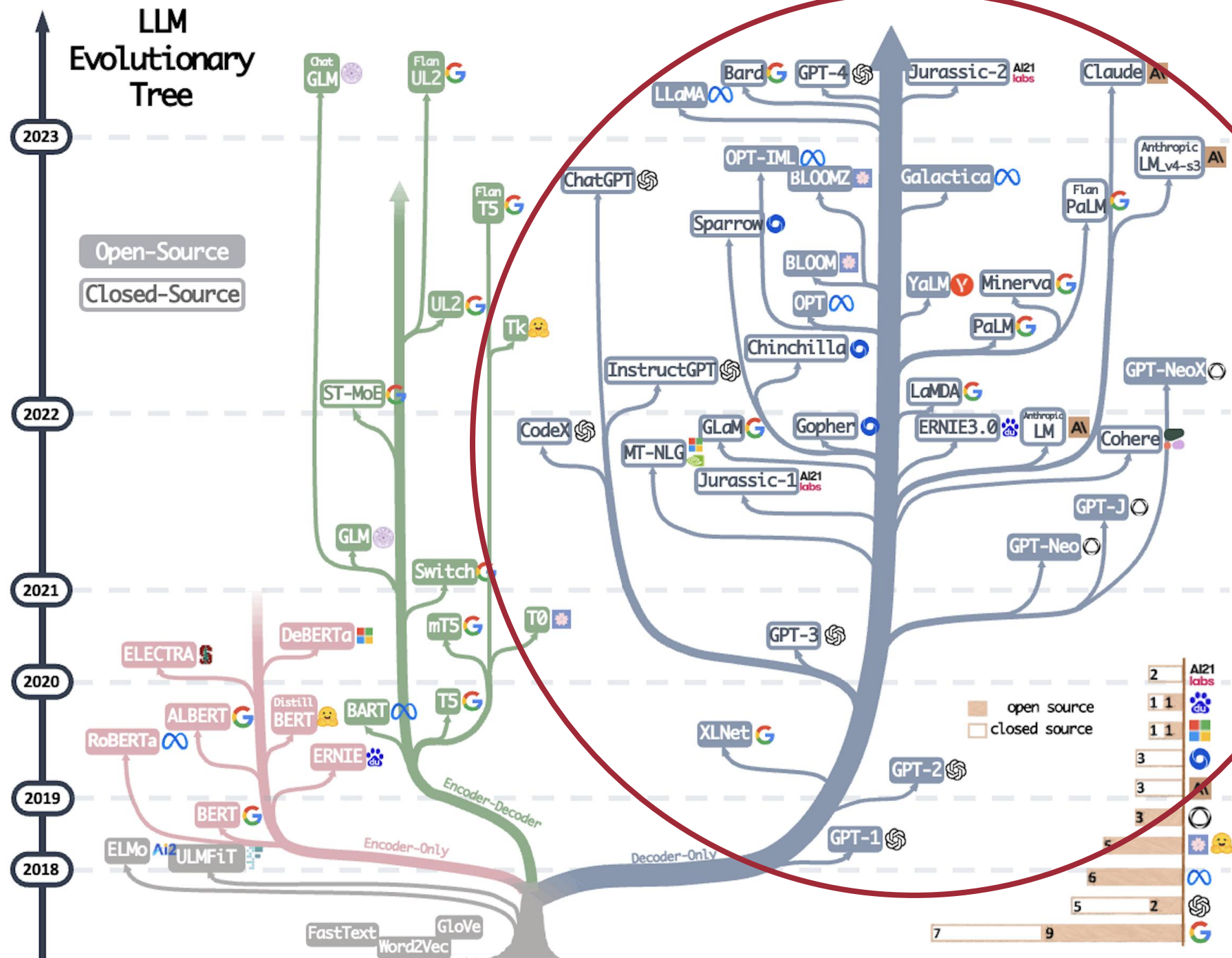
Some important BERT family members

(in my opinion)

- RoBERTa (better version of the original BERT) – Liu et al. 2019 (Facebook)
- Sentence-BERT (BERT fine-tuned to give good sentence embeddings) – Reimers & Gurevych 2019 (Technische Universität Darmstadt)
- DistilBERT (lite BERT) – Sanh et al. 2019
- ALBERT (lite BERT) – Lan et al. 2020
- HuBERT (BERT for speech embeddings) – Hsu et al. 2021

Decoder-Only Models





GPT Family

- Decoder-only
 - Input: Text sequence
 - Goal: Predict the next word given the previous ones
- How to use:
 - Ask GPT* to continue from a prompt.
 - Finetune smaller GPTs for more customized generation tasks.
 - ChatGPT cannot be finetuned since it is already finetuned
 - Use OpenAI's API to get them to fine-tune GPT* for you.
- Around GPT-2 was when pre-trained models became popular
- Around GPT-3 was when *just* prompting became a thing

Other Decoder-Only Models

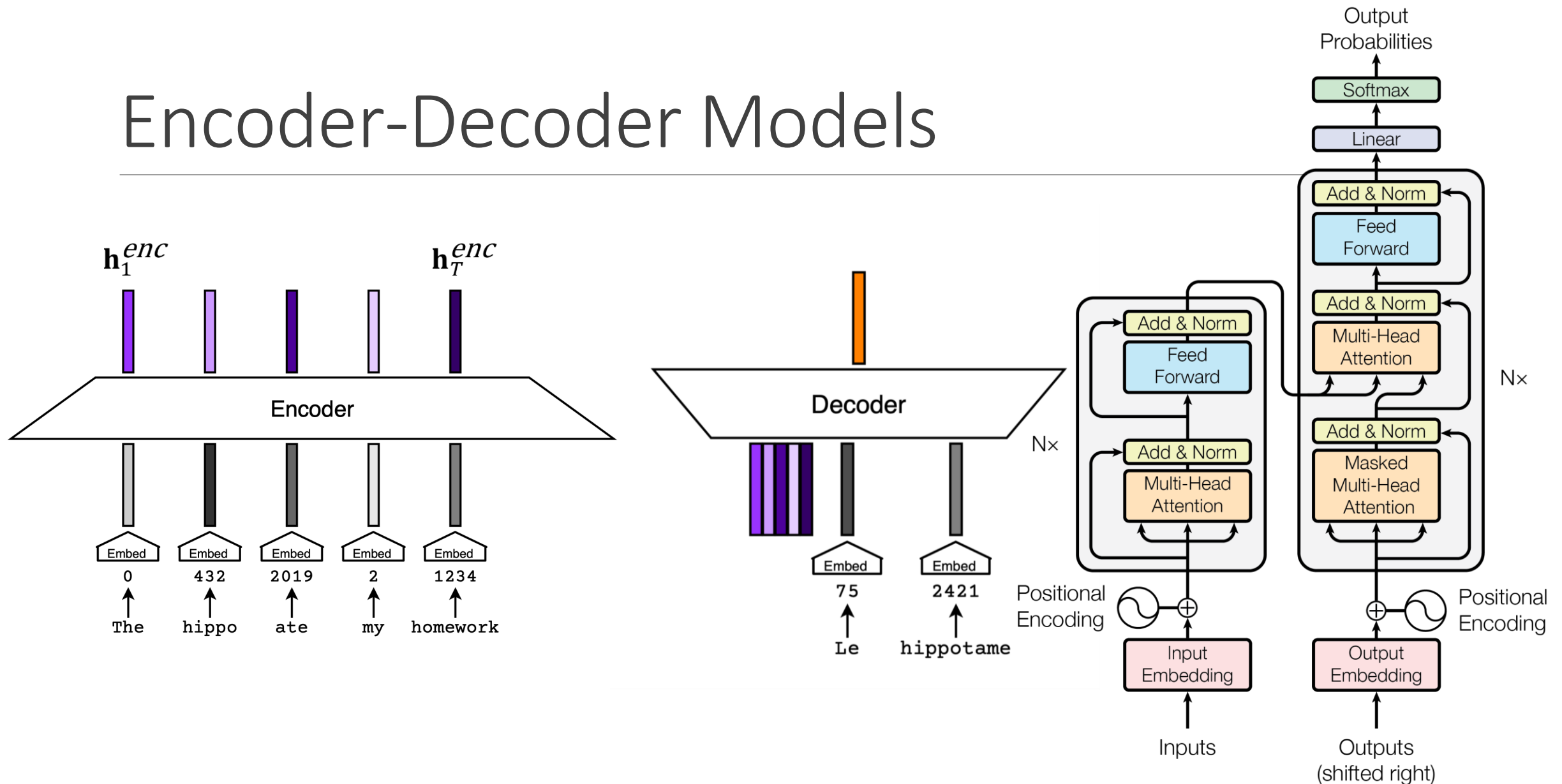
LLaMA 3/4 (Meta)

Claude 3 (Anthropic)

Gemma (Google)

OLMo 2 (AI2)

Encoder-Decoder Models



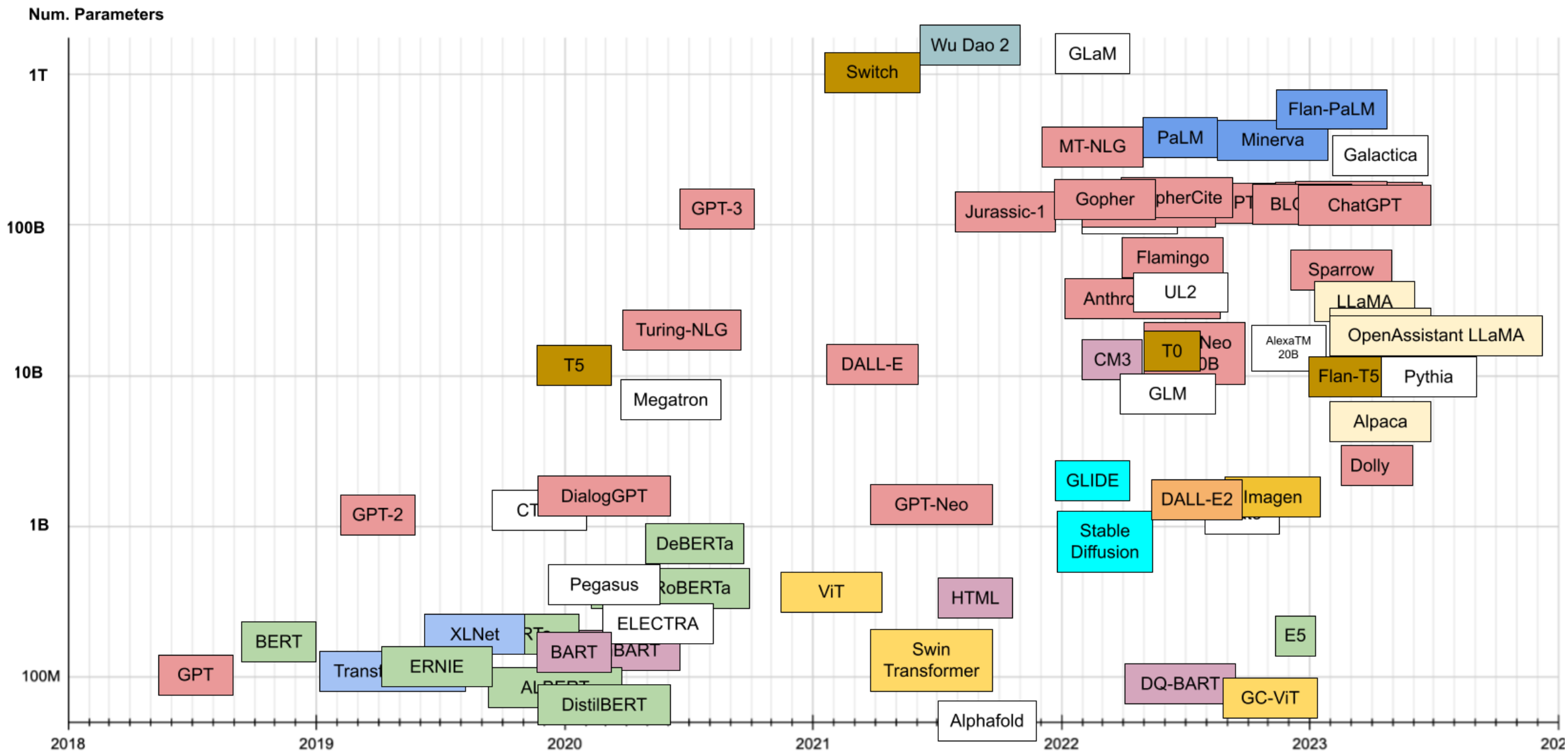


Enc-Dec Family of Models

- Encoder-decoder
 - Input: Text sequence with random word spans deleted
 - Goal: Generate the deleted word spans
- How to use:
 - Finetune smaller ones for either generation or classification tasks.
 - Prompt tuning (train a sequence of embedding which get prefixed to the input)

Some Enc-Dec family members

- T5 (Google)
- BART (combo of GPT and BERT) – (Facebook)
- DALL-E 2 (for caption prediction)



<https://amatriain.net/blog/transformer-models-an-introduction-and-catalog-2d1e9039f376/>