

Prompting & Decoding

Lara J. Martin (she/they)

<https://laramartin.net/neurosymbolic-text-gen/>

New Grader

Shaswati Saha

ssaha3@umbc.edu – grading questions only

TA is still Arya Honraopatil

ahonrao1@umbc.edu

Learning Objectives

Consider when to use various sampling algorithms

Distinguish between finetuning and prompting

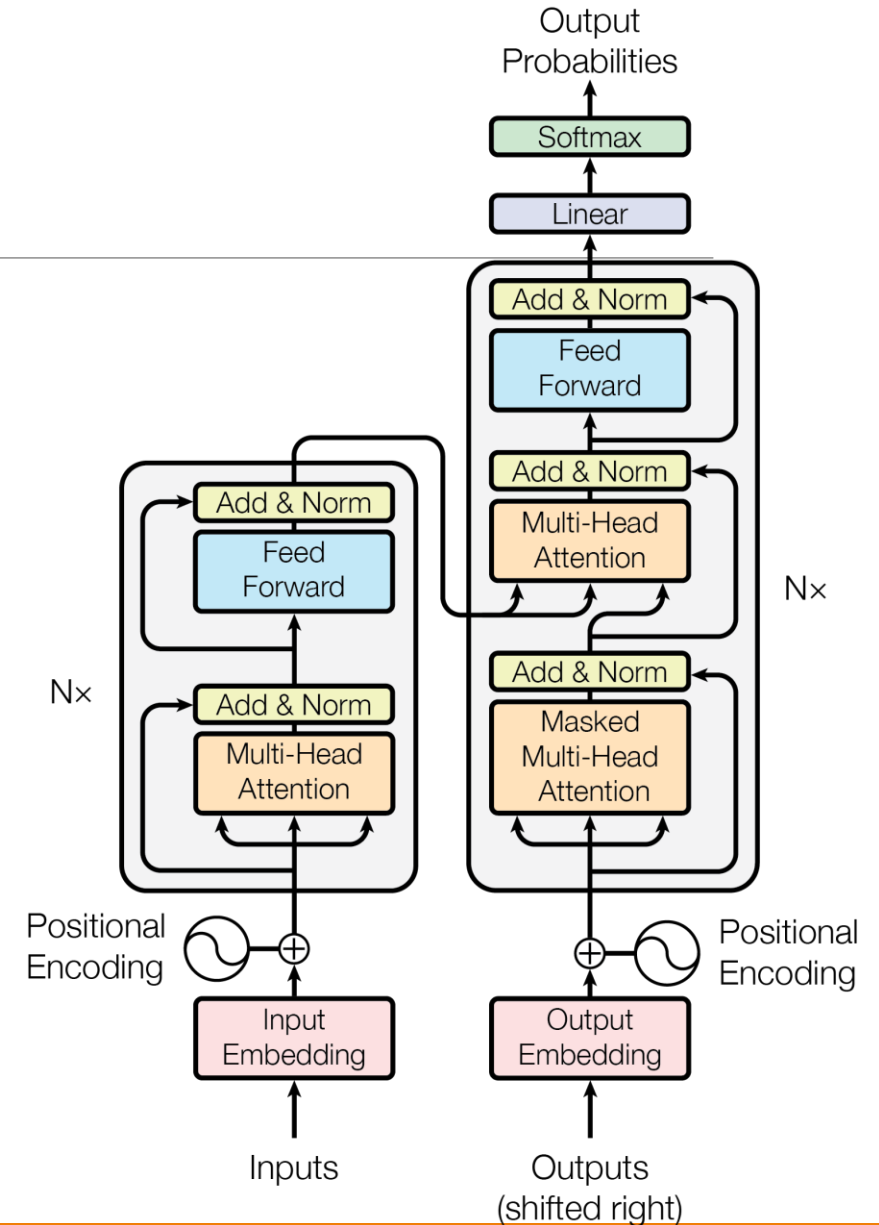
Distinguish between few-shot and zero-shot prompting

Try common prompting techniques like chain-of-thought

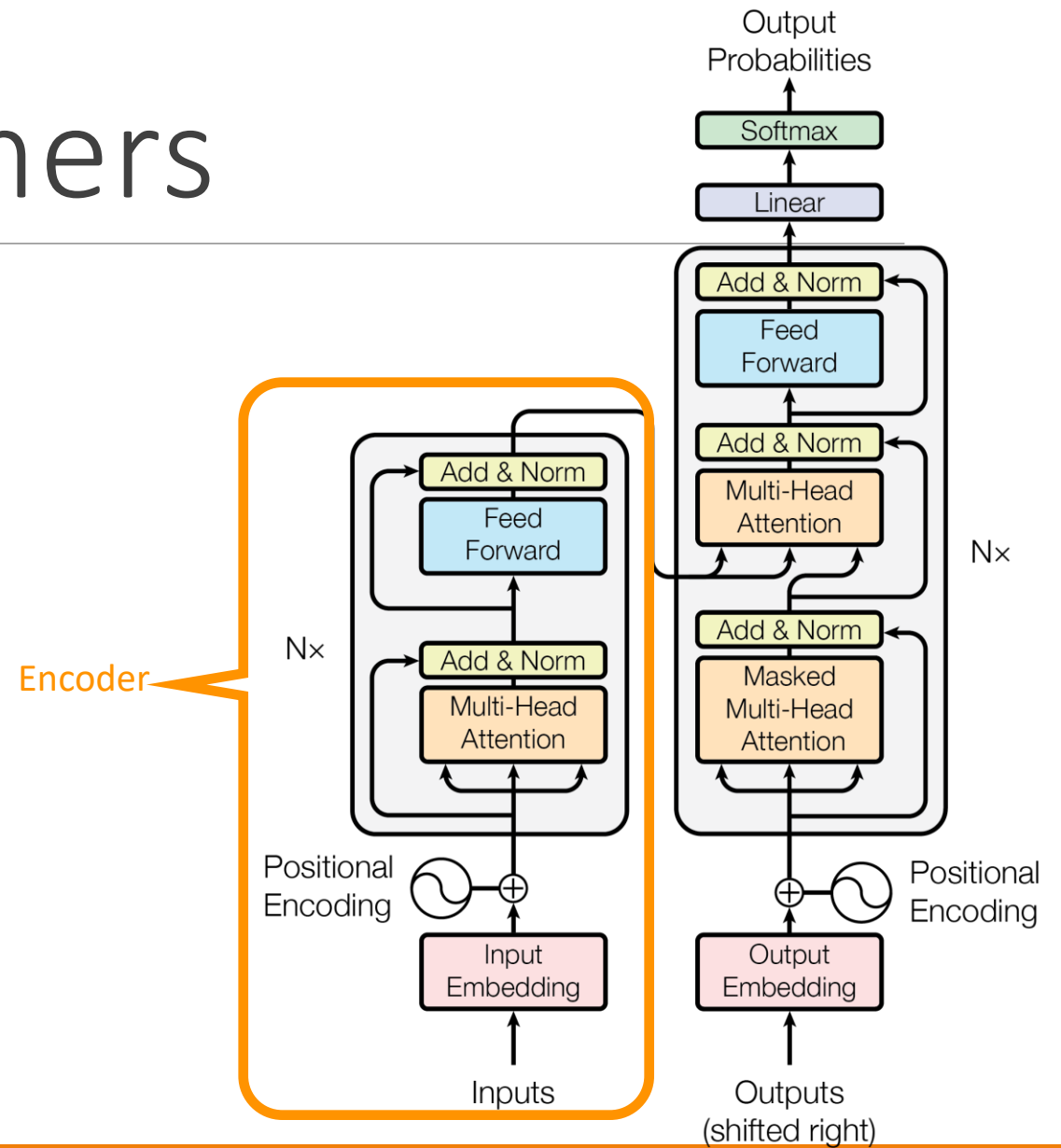
Review: Transformers

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

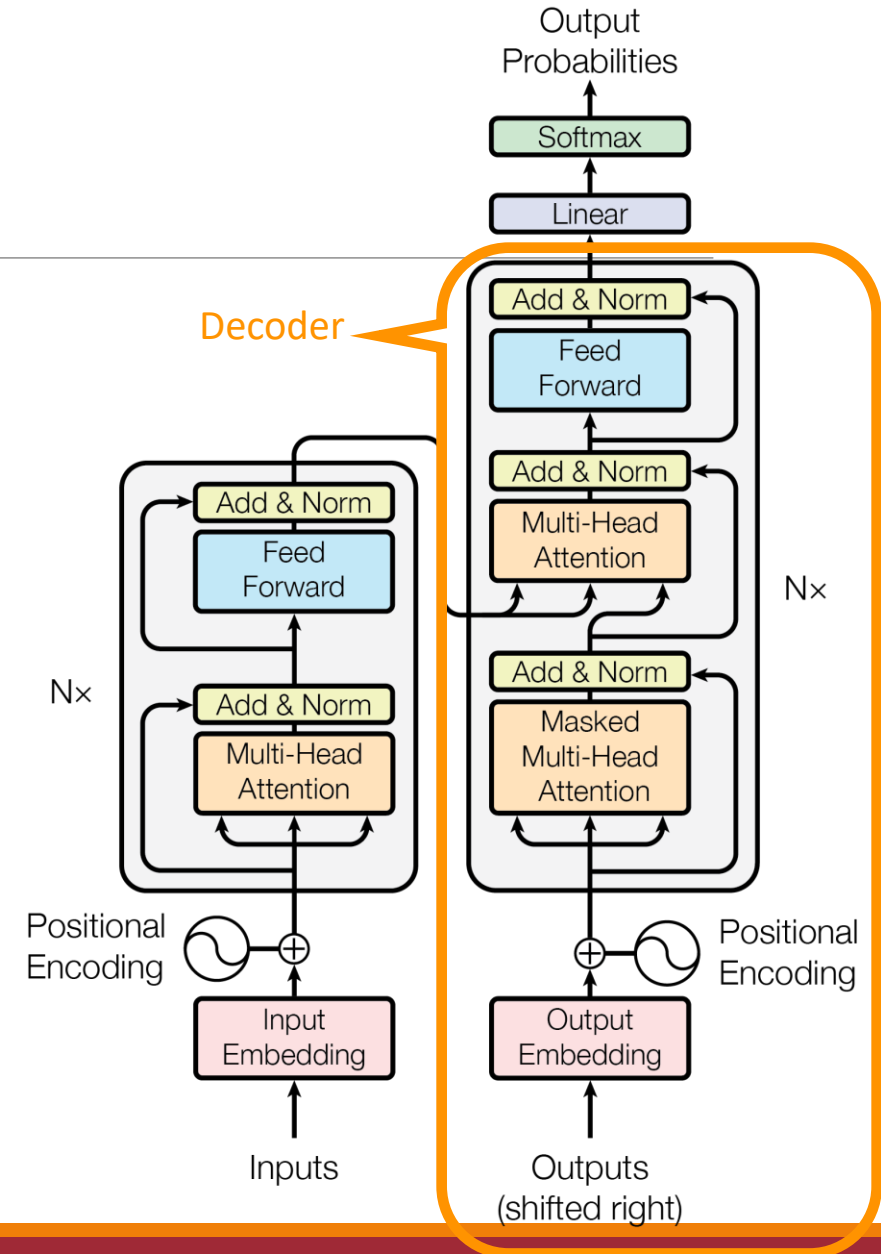
- introduces self-attention in addition to encoder-decoder attention



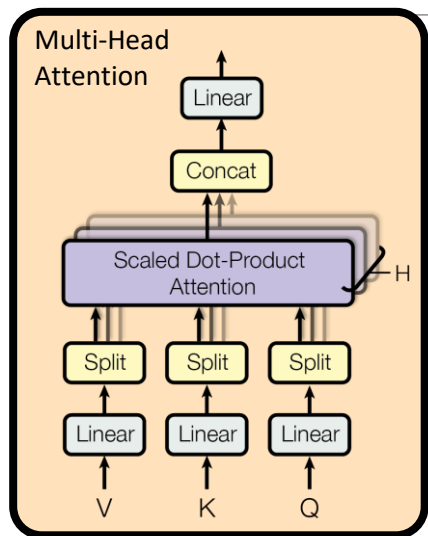
Review: Transformers



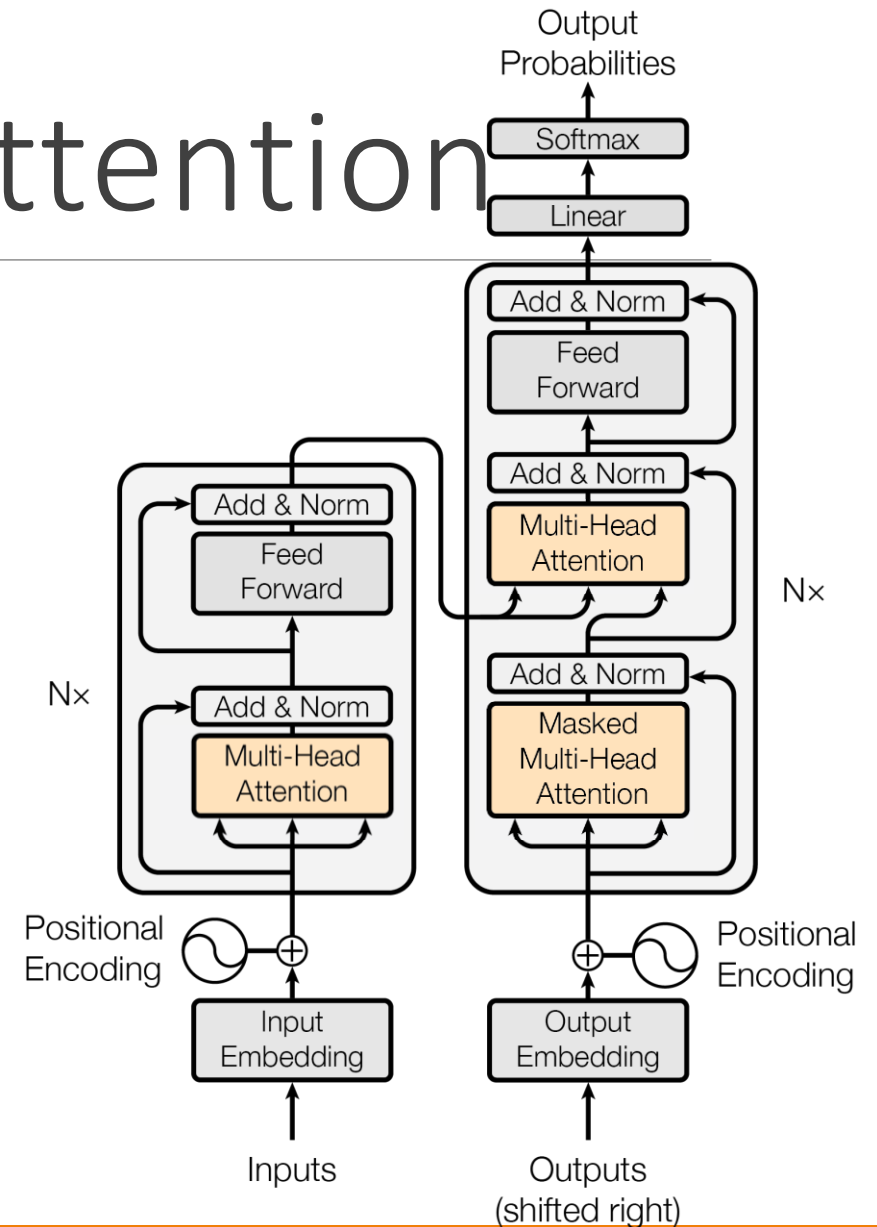
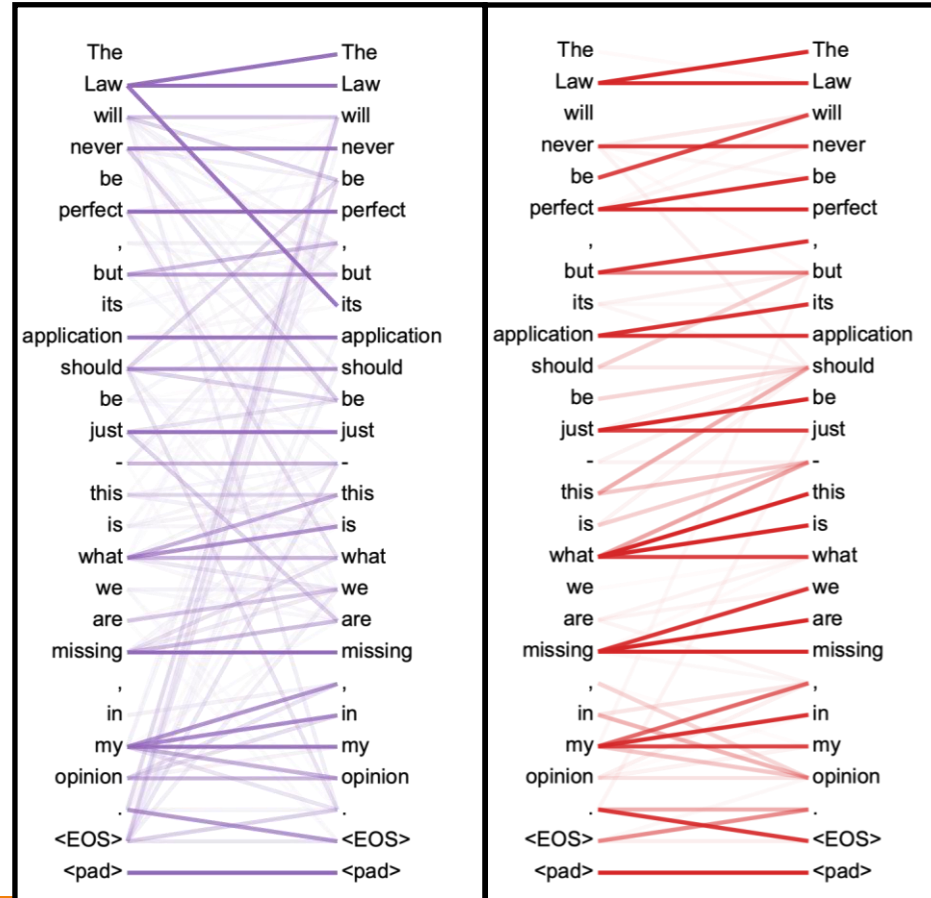
Review: Transformers



Review: Multi-Head Attention



Two different self-attention heads:



Review: Strengths of the Transformer Architecture

Training is easily parallelizable

- Larger models can be trained efficiently.

Does not “forget” information from earlier in the sequence.

- Any position can attend to any position.

Review: Weaknesses of the Transformer Architecture

We can use a lot of data to train → expensive (money, time)

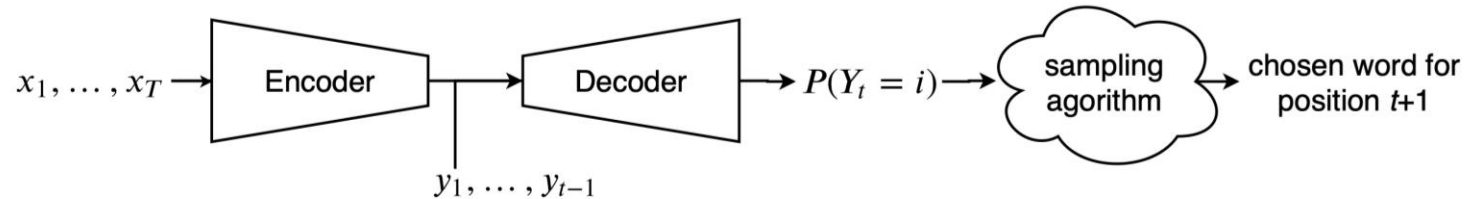
Can't actually remember things, just looks back

Review: Generating Text

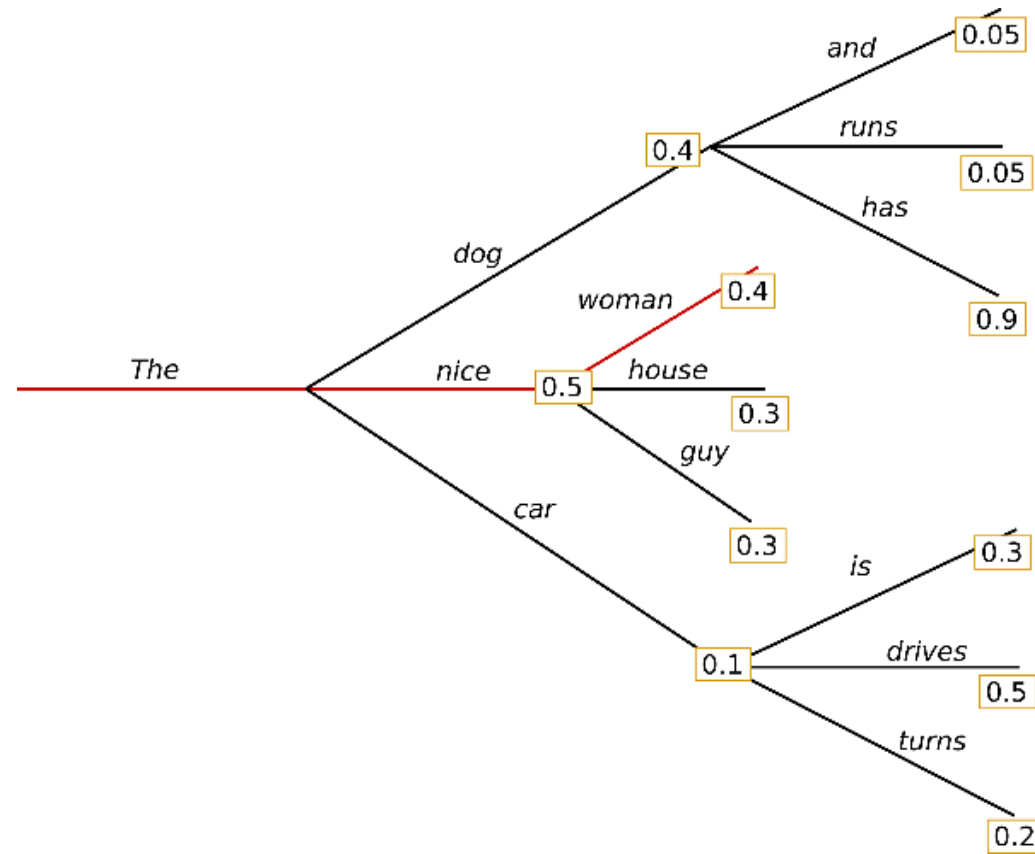
Also sometimes called decoding



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



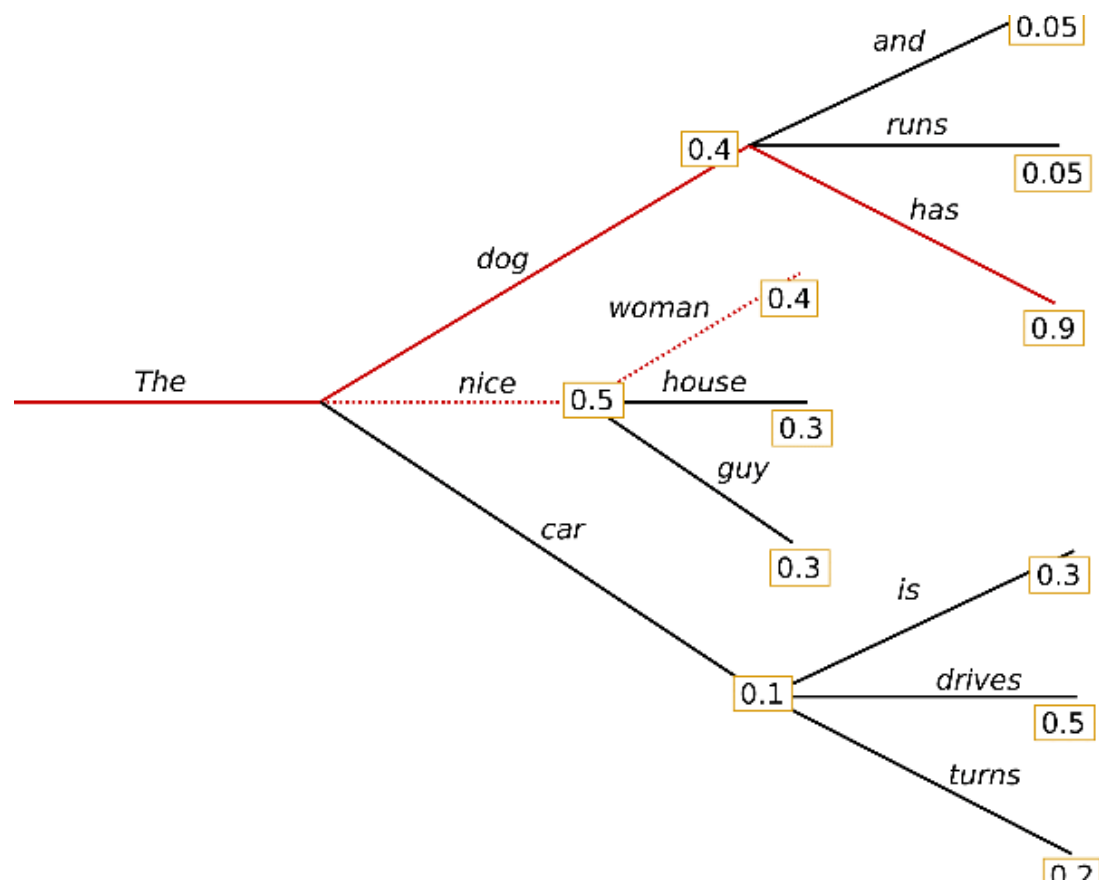
Greedy Search (Argmax)



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

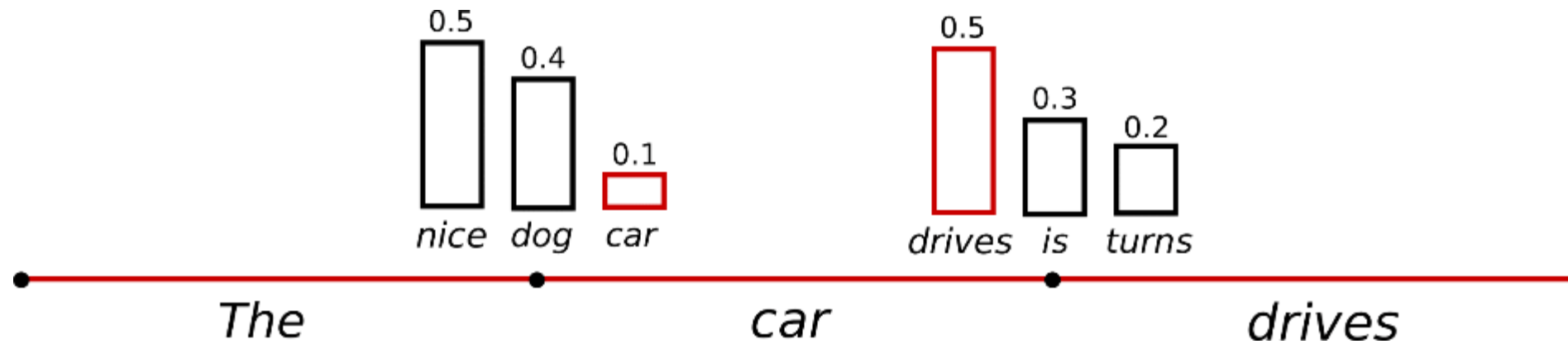
Beam Search

Number of Beams = 2



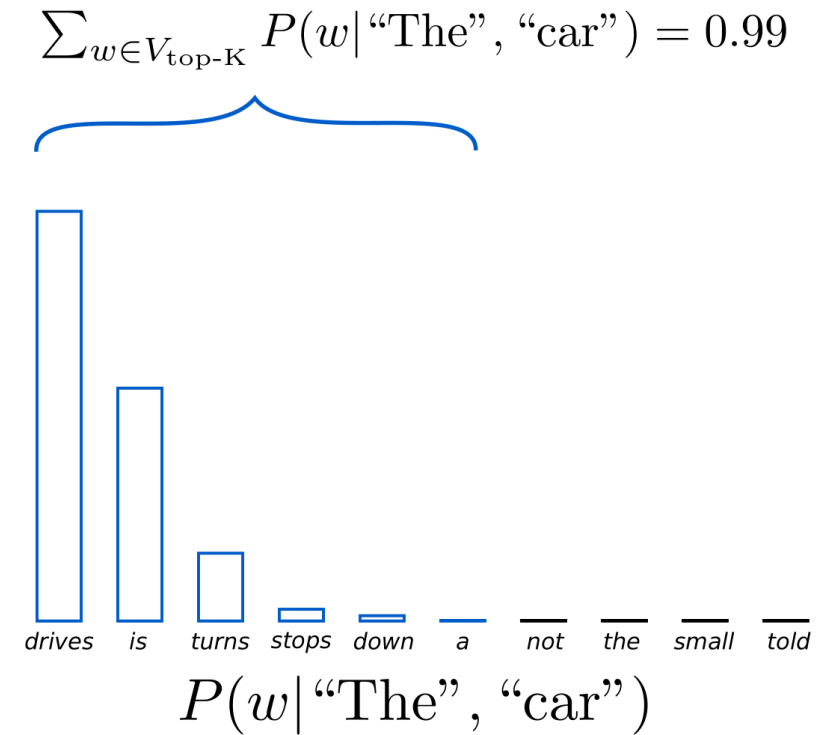
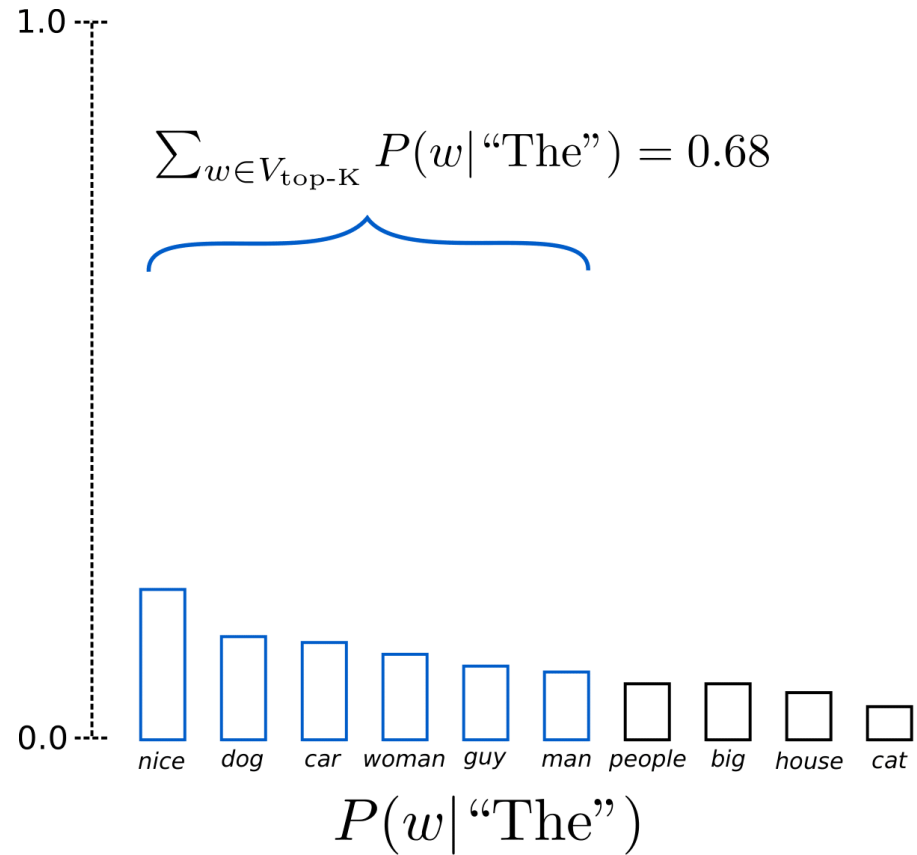
<https://huggingface.co/blog/how-to-generate>

Random Sampling



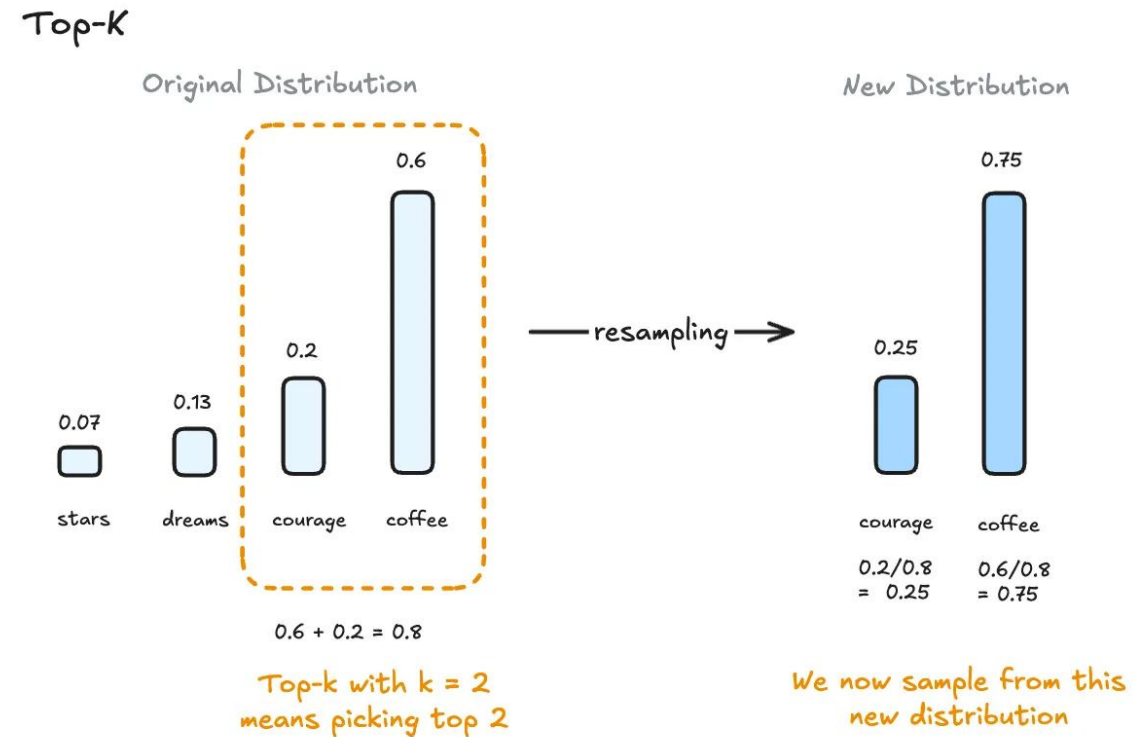
<https://huggingface.co/blog/how-to-generate>

Top-K Sampling

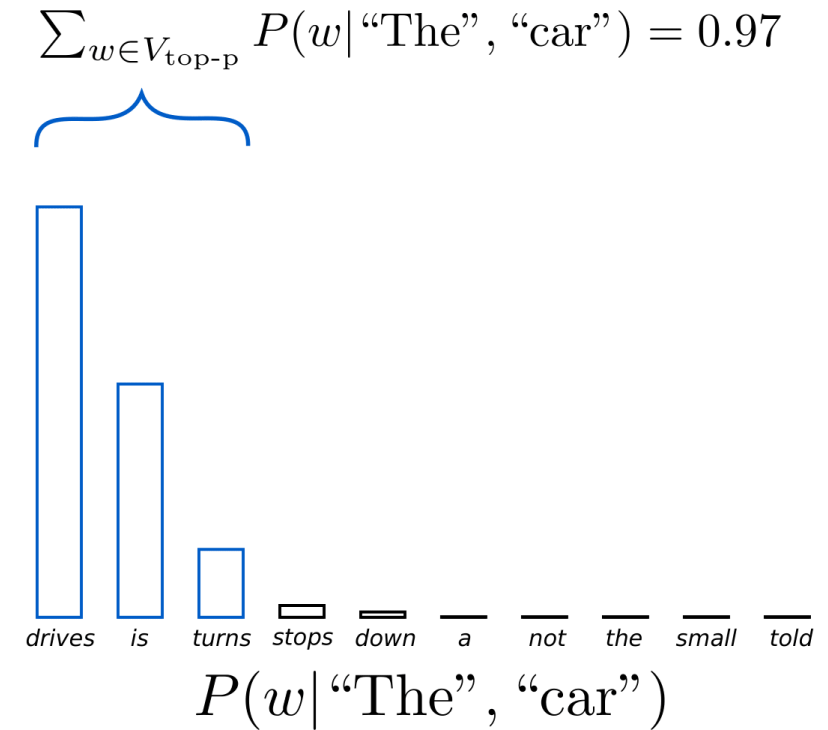
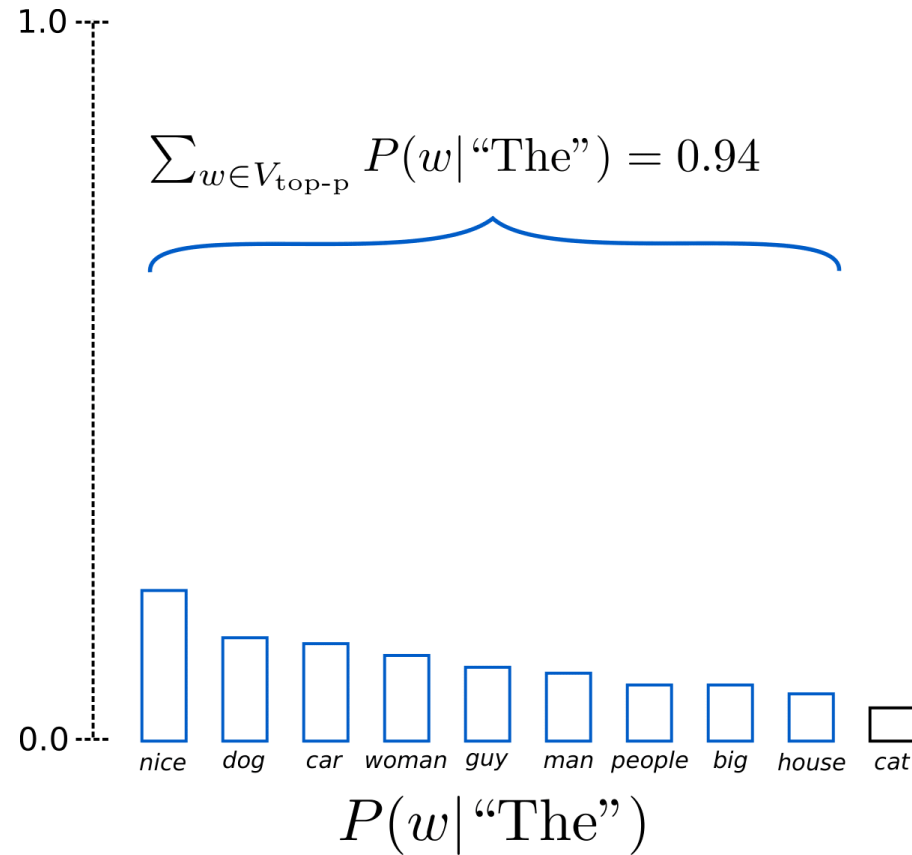


<https://huggingface.co/blog/how-to-generate>

Resampling

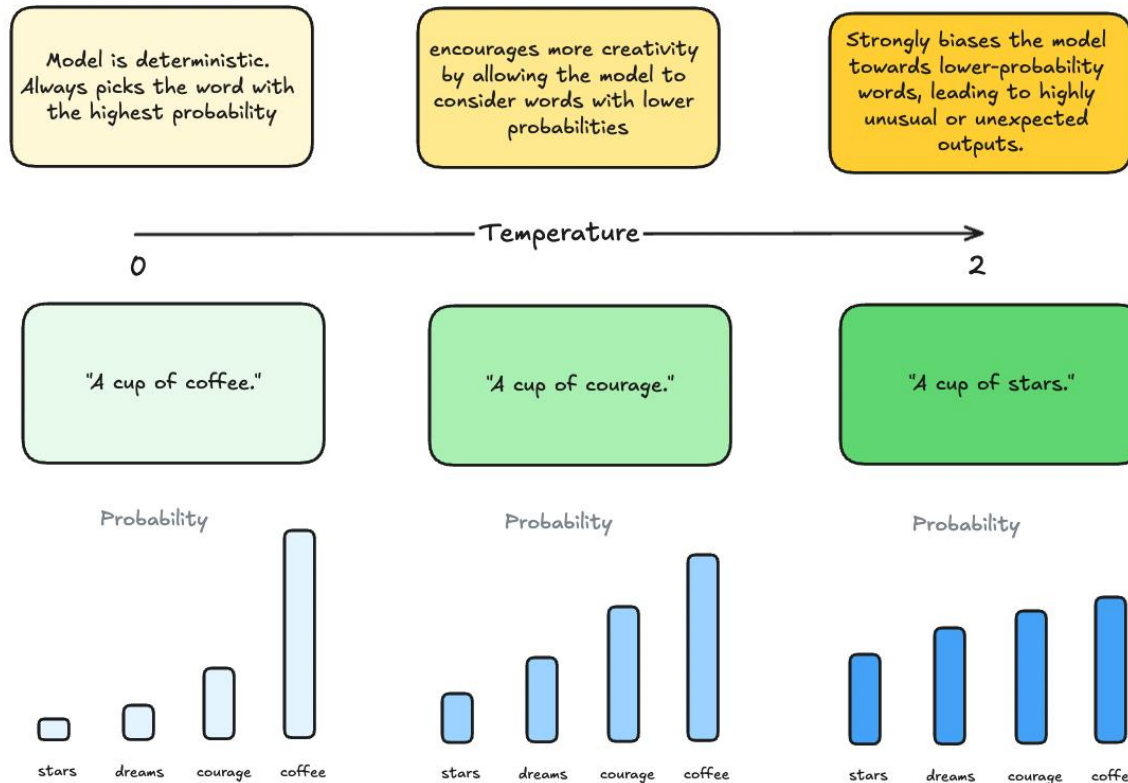


Top-P / Nucleus Sampling



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

“Temperature”



Temperature in Action

Playground

Save

View code

Share

...

Does it always rain on Tuesdays?



No, it does not always rain on Tuesdays.

Mode



Model

text-curie-001



Temperature 0.35



Does it always rain on Tuesdays?



No, Wednesday is the normal precipitation day. However, Tuesday can occasionally experience light rain or even a thunderstorm.

Mode



Model

text-curie-001



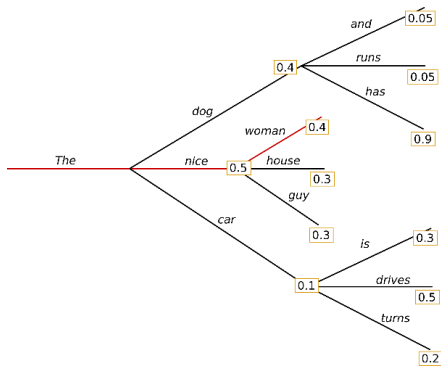
Temperature 1



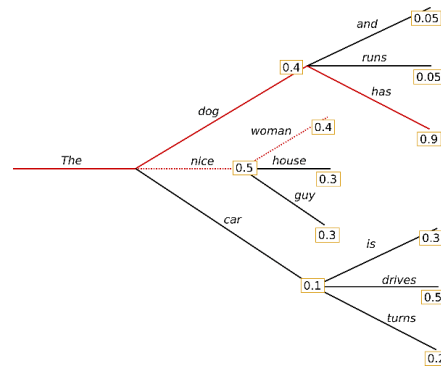
18

Think-Pair-Share

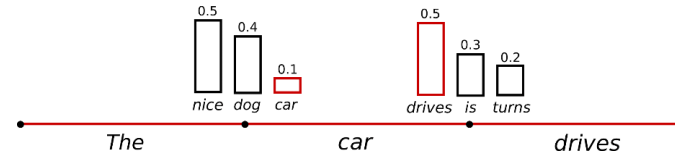
When might you want to use one sampling algorithm over the other?



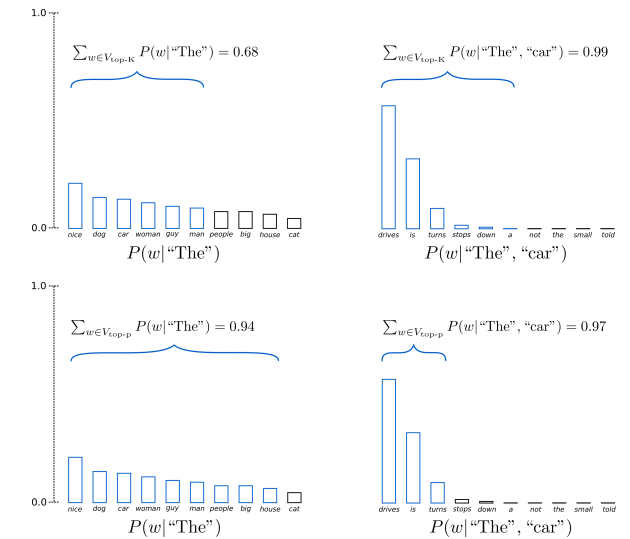
Greedy



Beam Search



Random Sampling



Top-K/P

Fine-tuning

Start with pre-trained model

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

- Sometimes you can adjust the weights of the whole model instead of just the last layer

Start with generalized “foundation” 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 Child¹ David Luan¹ Dario Amodei^{**1} Ilya Sutskever^{**1}

Abstract

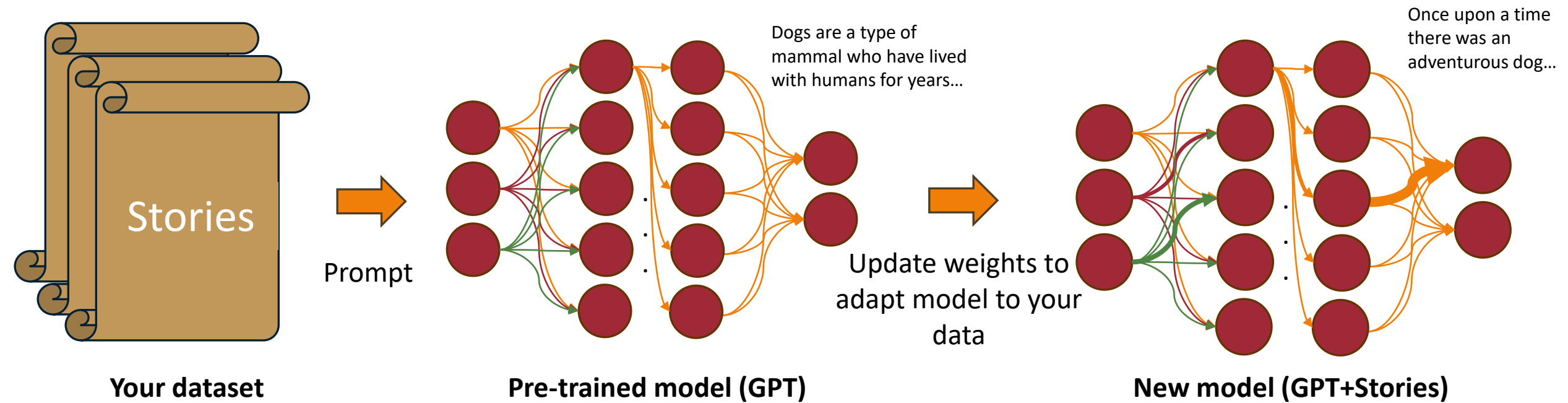
Natural language processing tasks, such as question answering, machine translation, reading comprehension, and summarization, are typically approached with supervised learning on task-specific datasets. We demonstrate that language models begin to learn these tasks without any explicit supervision when 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

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 (if finetuning over all weights) or fits too well/memorizes training data



Most finetuning jobs should do 1-3 epochs through your data

Pre-trained models

Most LLMs people use today are pre-trained models

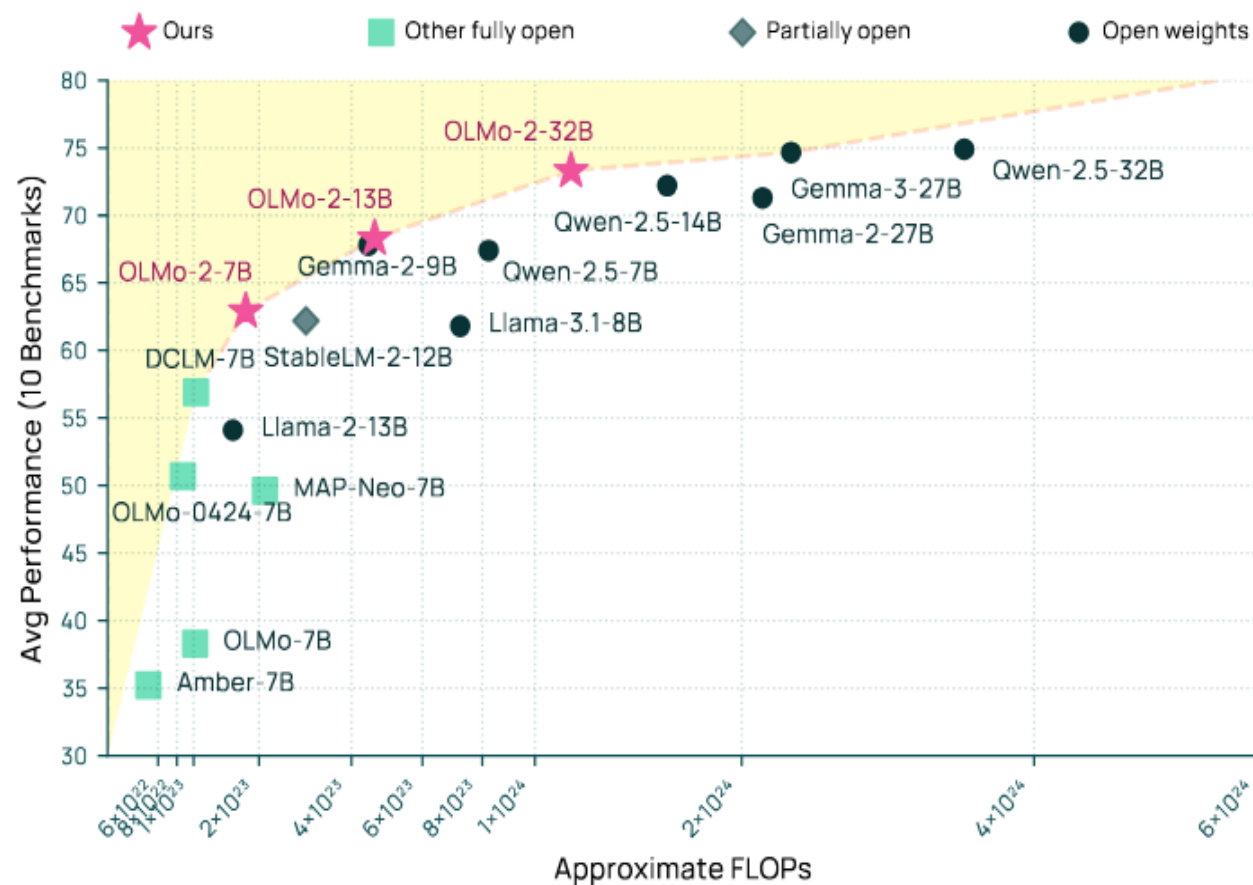
Trained on “the Internet” → Impossible to know all of what it’s train on

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

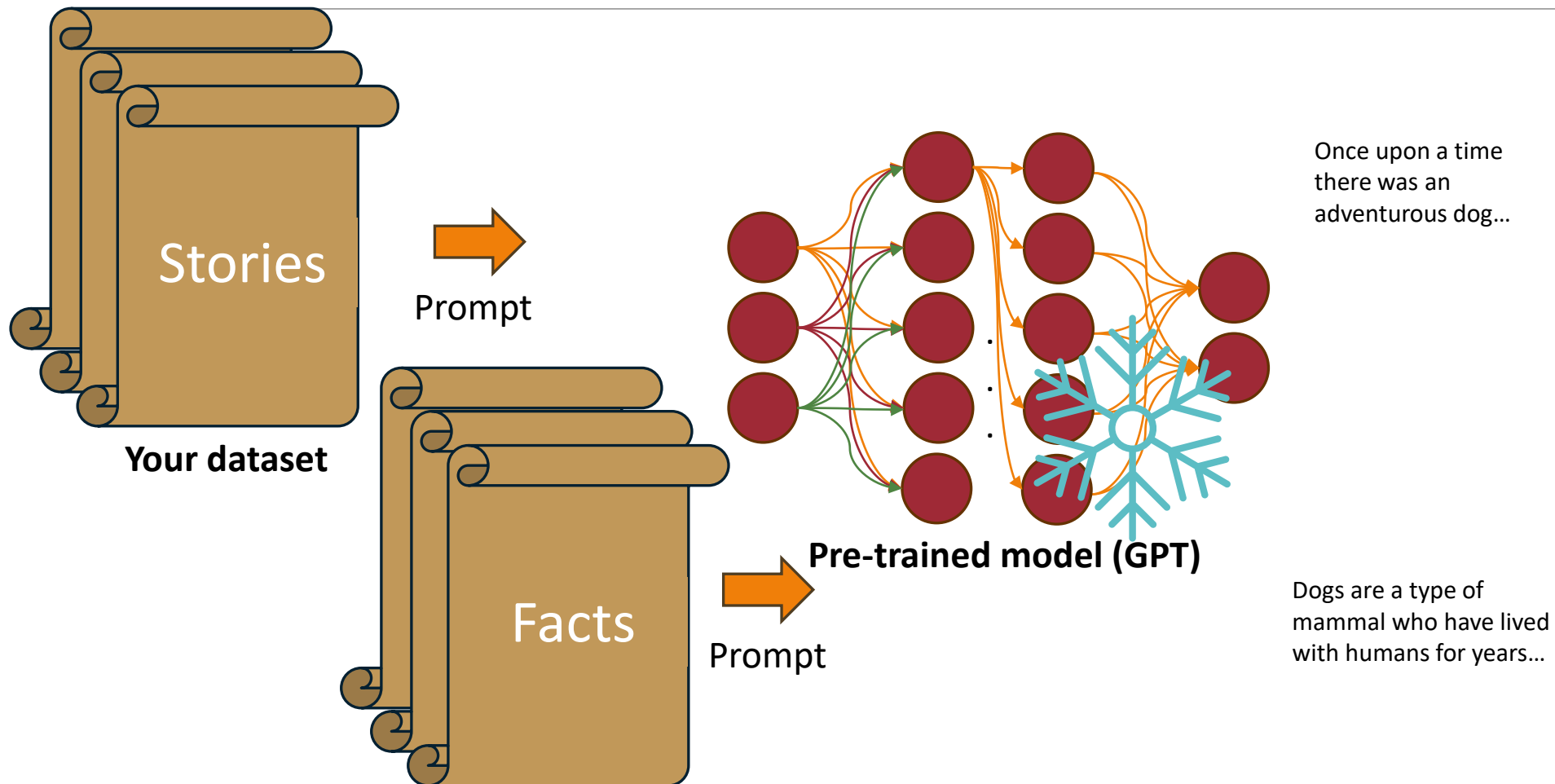
Can then be finetuned on specific data

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

Open-Sourced Models



Prompting

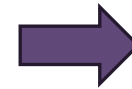
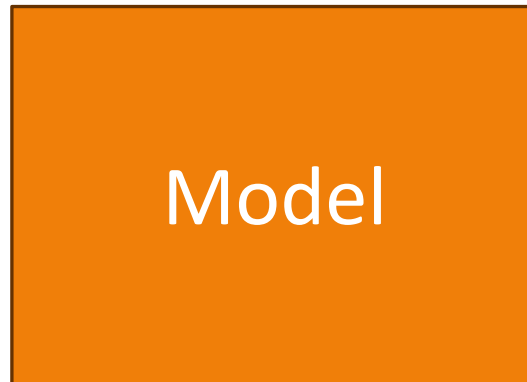
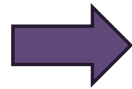


Zero-shot Prompting

You are a helpful assistant.
You will be tagging the parts
of speech in sentences.

Instructions

Task



Output

Sentence:
The dog ate the giant fish.

Few-shot Prompting

Instructions

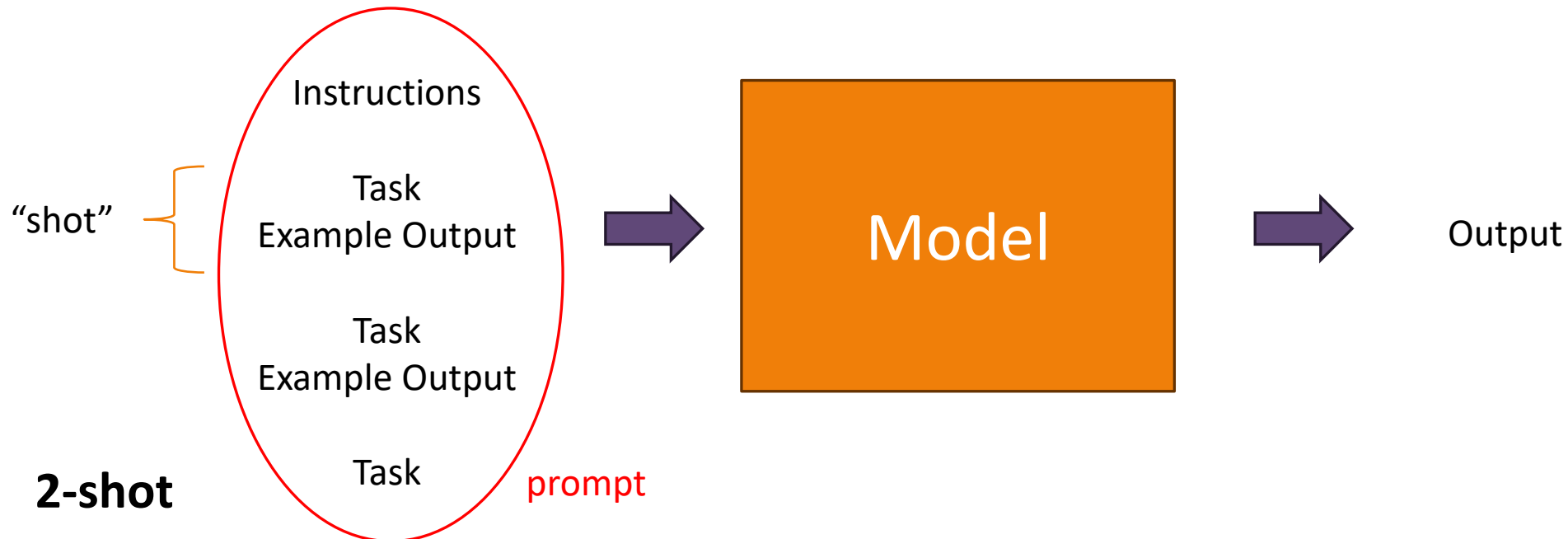
You are a helpful assistant.
You will be tagging the parts
of speech in sentences.

Task

Sentence:
The dog ate the giant fish.

Example Output

The dog ate the giant fish.
D N V D Adj N



Prompting



"A child playing on a sunny happy beach, their laughter as they build a simple sandcastle, emulate Nikon D6 high shutter speed action shot, soft yellow lighting."
Generated with Midjourney.

via <https://zapier.com/blog/ai-art-prompts/>

Need to be really specific
(also match the training data)

Chain-of-Thought Prompting

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

Standard Prompting

Model Output

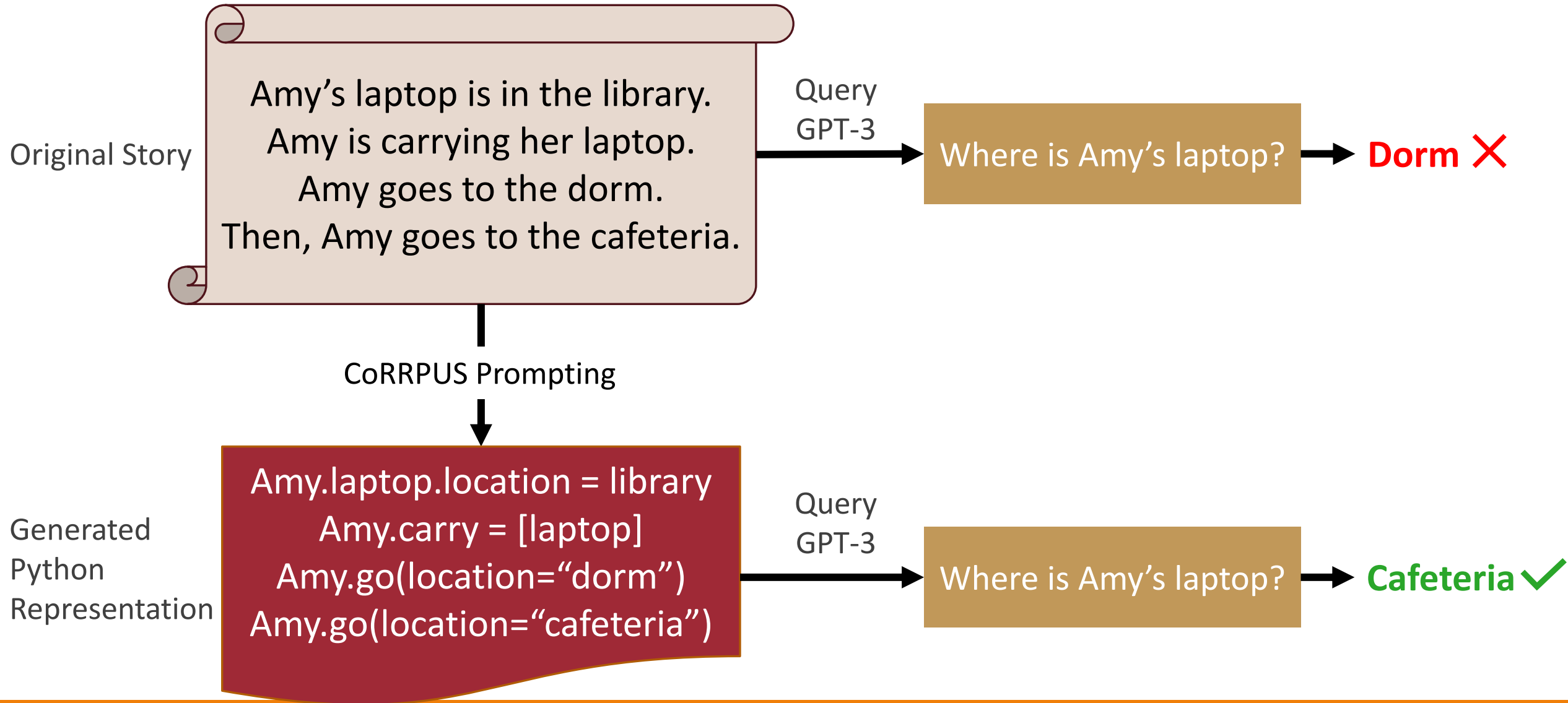
A: The answer is 27. ❌

Chain-of-Thought Prompting

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. ✅

CoRRPUS (Code Representations to Reason & Prompt over for Understanding in Stories)



CoRRPUS Chain-of-Thought Prompting

Three versions that are initialized the same:

Comment

```
def story(self):
```

```
## Mary moved to the bathroom.  
self.Mary.location = "bathroom"  
## Mary got the football there.  
self.Mary.inventory.append("football")  
...
```

Specific Functions

```
self.Mary_moved_to_the_bathroom()  
self.Mary_got_the_football_there()  
self.John_went_to_the_kitchen()  
self.Mary_went_back_to_the_garden()
```

```
def Mary_moved_to_the_bathroom()  
    self.Mary.location="bathroom"  
def Mary_got_the_football_there():  
    ...
```

Abstract Functions

```
def go(self, character, location):  
    character.location = location  
    for item in character.inventory:  
        item.location = location  
def pick_up(): ...
```

```
def story(self):  
    ## Mary moved to the bathroom.  
    self.go(character=self.Mary,  
            location = "bathroom")  
    ...
```

Tested On 2 Tasks

bAbI (Weston et al. 2015)

- Task 2: Stories tracking objects that characters carry

Re³ (Yang et al. 2022)

- Identifying inconsistencies in stories (e.g., descriptions of characters' appearances, relationships)
- Stories were generated from a list of facts (the premise). They also generated premises with a contradiction.

bAbI (Weston et al. 2015)

Method	# Shot	Accuracy ↑
Random	-	25%
GPT-3	1	56.5%
Chain of Thought (Creswell et al. 2022)	1	46.4%
Selection-Inference (Creswell et al. 2022)	1	29.3%
Dual-System (Nye et al. 2021)	10	100%
CoRRPUS (comment)	1	67.0%
CoRRPUS (specific)	1	78.7%
CoRRPUS (abstract)	1	99.1%

Re³

The task is to see what stories match what premises based on the facts extracted from both.

Joan Westfall premise

Attribute	Value
Gender	Female
Occupation	Teacher
Brother	Brent Westfall
Appearance	Blue eyes

entails

entails

contradicts

Joan Westfall in story

Attribute	Value
Gender	Female
Father	Jason Westfall
Brother	Brent Westfall
Appearance	Brown eyes

Takeaway: structured representations help!

Re³ (Yang et al. 2022)

Method	ROC-AUC ↑
Random	0.5
GPT-3	0.52
Entailment (Yang et al. 2022)	0.528
Entailment with Dense Passage Retrieval (Yang et al. 2022)	0.610
Attribute Dictionary → Sentence (Yang et al. 2022)	0.684
CoRRPUS (comment)	0.751
CoRRPUS (specific)	0.794
CoRRPUS (abstract)	0.704

→ Probably because functions like `set_age(self, character, age)` complicate more than they help.

Prompt Composition

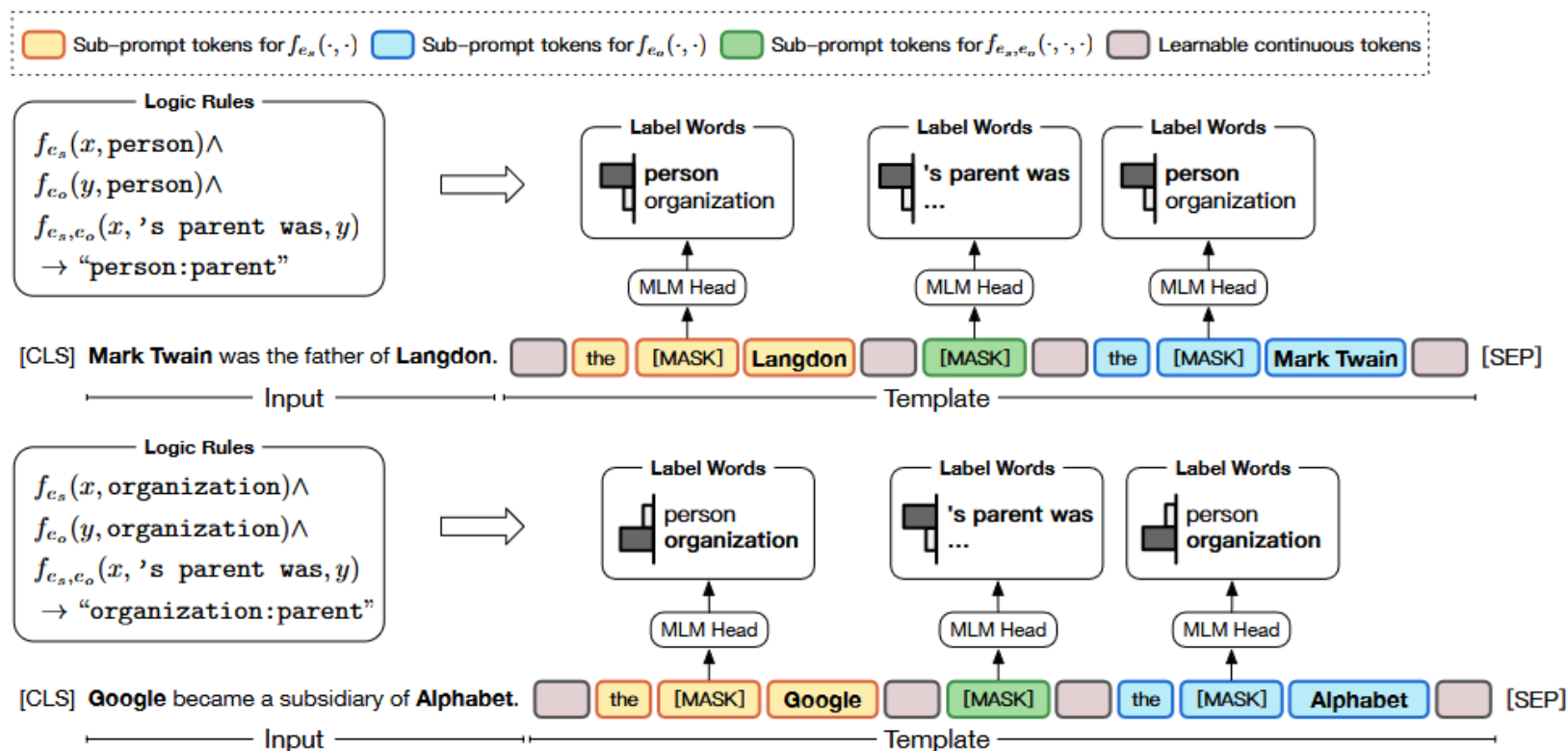
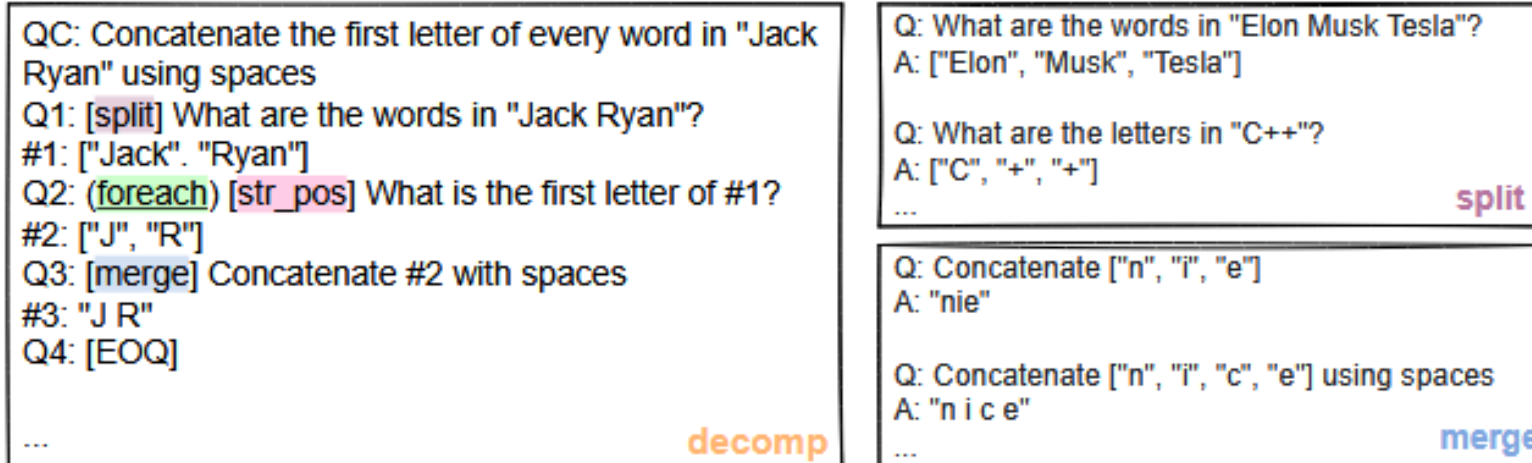


Figure 2: Illustration of PTR. By composing the sub-prompts of the conditional functions $f_{e_s}(\cdot, \cdot)$, $f_{e_o}(\cdot, \cdot)$ and $f_{e_s, e_o}(\cdot, \cdot, \cdot)$, we could easily get an effective prompt to distinguish “person:parent” and “organization:parent”.

Decomposition

Breaking down complicated problems into components that feed into each other

Each instance of a model does a different step or function



Self-Criticism

LLM “ruminates” on its output to try to come up with better output

Precursor to reasoning models that are finetuned to do this automatically

```
Question: Who was the third president of the United States?  
Here are some brainstormed ideas: James Monroe  
Thomas Jefferson  
John Adams  
Thomas Jefferson  
George Washington  
Possible Answer: James Monroe  
Is the possible answer:  
  (A) True  
  (B) False  
The possible answer is:
```

Meta-Prompting

Prompting to generate prompts

Meta-prompts...

- “encode what the LLM should “do” given different descriptions of the same task” [1]
- “will return the relevant outputs for an arbitrary task, provided that the task description is provided as an input” [1]

Meta-Prompting Example (WHAT-IF)

```
{{
  "branching_event_number": branching_event,
  "original_decision" : storyline[f"node_{branching_node}"]['decision'],
  "alternate_decision" :
    storyline[f"node_{branching_node}"]['alternate_decision'],
  "new_story_length": (len(storyline) - branching_node) * 3,
  "major_plot_points" : mpp,
  "prompt": f"TODO: <a prompt for ChatGPT for every branching point above with
    following requirements:
      1. Ask to use the original storyline as a reference to write an alternate
        storyline that branches out at event {branching_event} if {char_name}
        {storyline[f'node_{branching_node}'] ['alternate_decision']} instead
        of {storyline[f'node_{branching_node}'] ['decision']}.
      2. Provide 5 thought-provoking concrete guiding questions as potential
        directions to explore that expand the following:\n
          a. How would alternate decision change or replace {mpp}?
          b. How would {char_name} make key decisions that overcome new
            challenges and propel the story forward?
      3. Describe what an ideal alternate storyline should look like.
      4. Ask to output the alternate storyline as a list of
        {(len(storyline) - branching_node + 1) * 3} events that has
        {storyline[f'node_{branching_node}'] ['alternate_decision']} as the
        first event>"
}}
```

Other Tricks of the Trade

Instruction-tuned models like GPT-3.5 and Mistral-7B-Instruct like to be given a “role” first (e.g., “You are a helpful writing assistant.”)

The more defined the task, the better

- More details
- One thing to do at a time

LLMs are overly confident (like people on the internet)

- To “objectively” have the model evaluate something, you should create a new instance and ask it

Chain-of-thought prompting helps models come up with better answers

They will “Yes and...” your prompt

Back to this example: Leading questions

Playground

Save

View code

Share

...

Does it always rain on Tuesdays?



No, it does not always rain on Tuesdays.

Mode



Model

text-curie-001



Temperature 0.35



Does it always rain on Tuesdays?



No, Wednesday is the normal precipitation day. However, Tuesday can occasionally experience light rain or even a thunderstorm.

Mode



Model

text-curie-001



Temperature 1



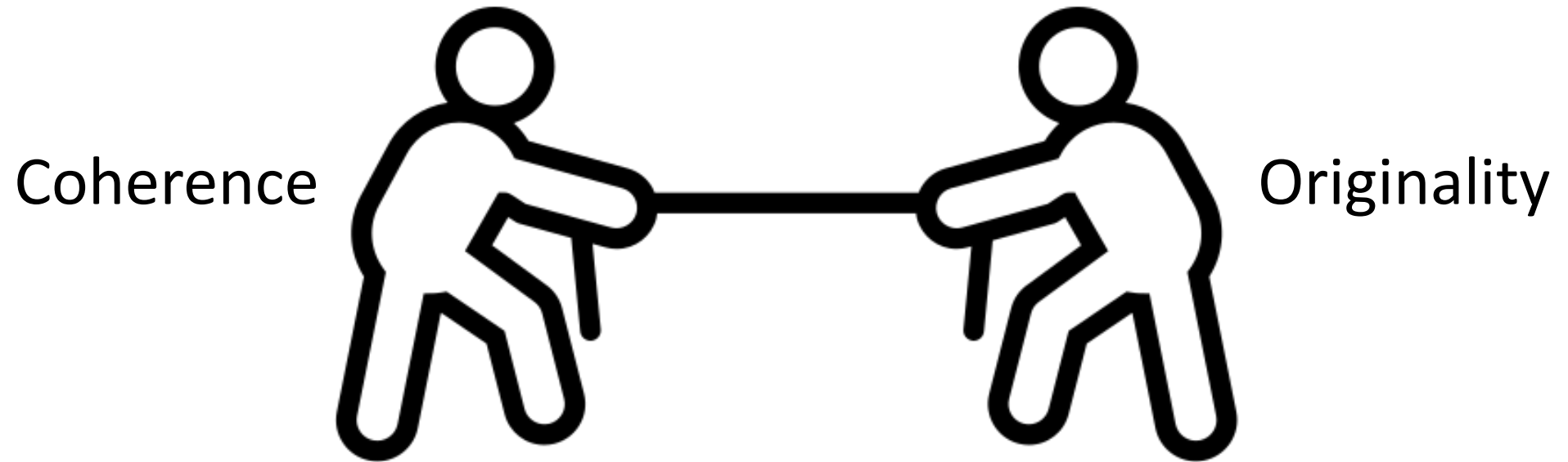
42

Dealing with any language model

Likelihoods → Not cause & effect

What is probable might not be possible.

Lara's Language Model Tradeoff



<https://thenounproject.com/icon/tug-of-war-1016981/>