Large Language Models (GPT, ChatGPT, Llama, ...)

Tim Metzler

Department of Computer Science

HBRS / ST.A.

Large Language Models

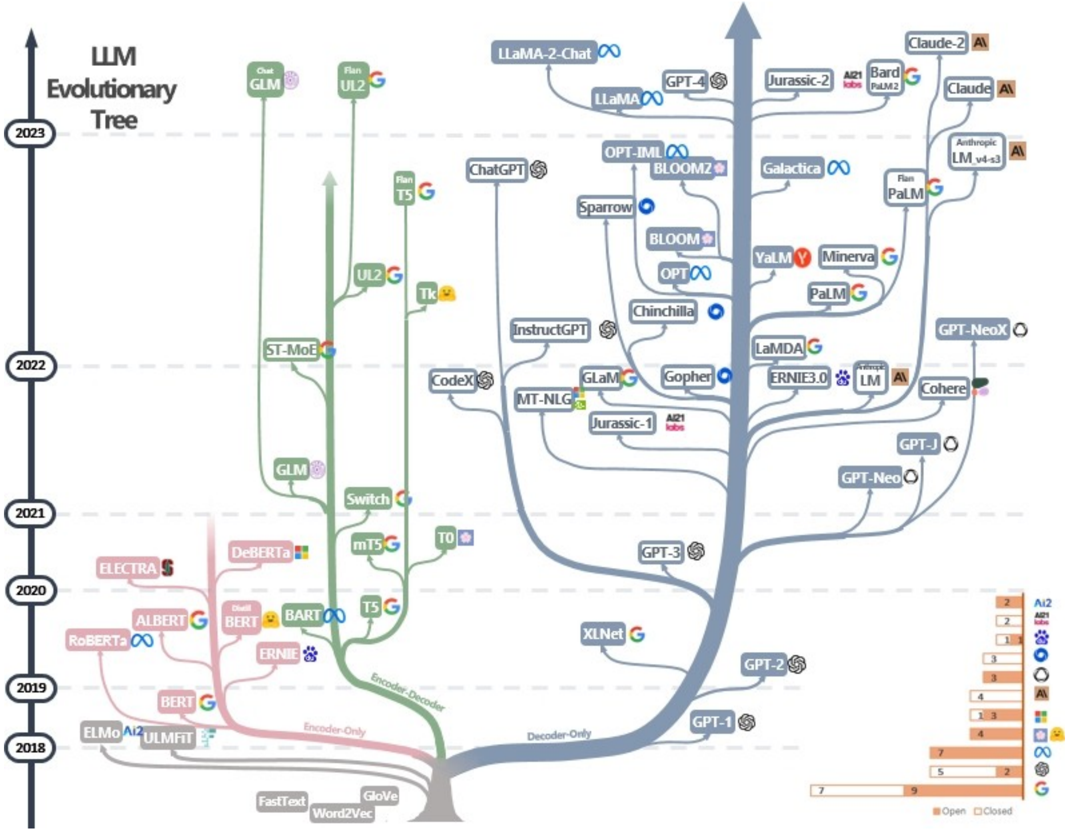


Fig. 1: LLM Evolutionary Tree (https://github.com/Mooler0410/LLMsPracticalGuide)

Large Language Models

Encoder Models (previous lecture on Transformers)

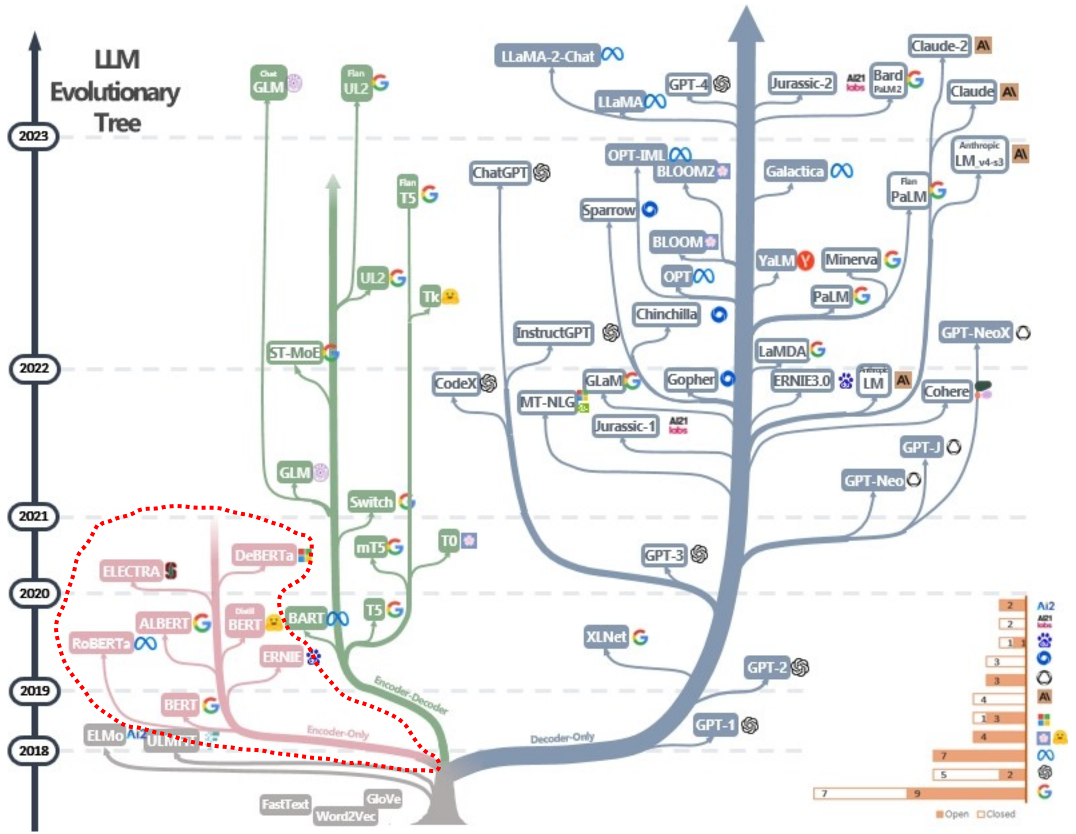
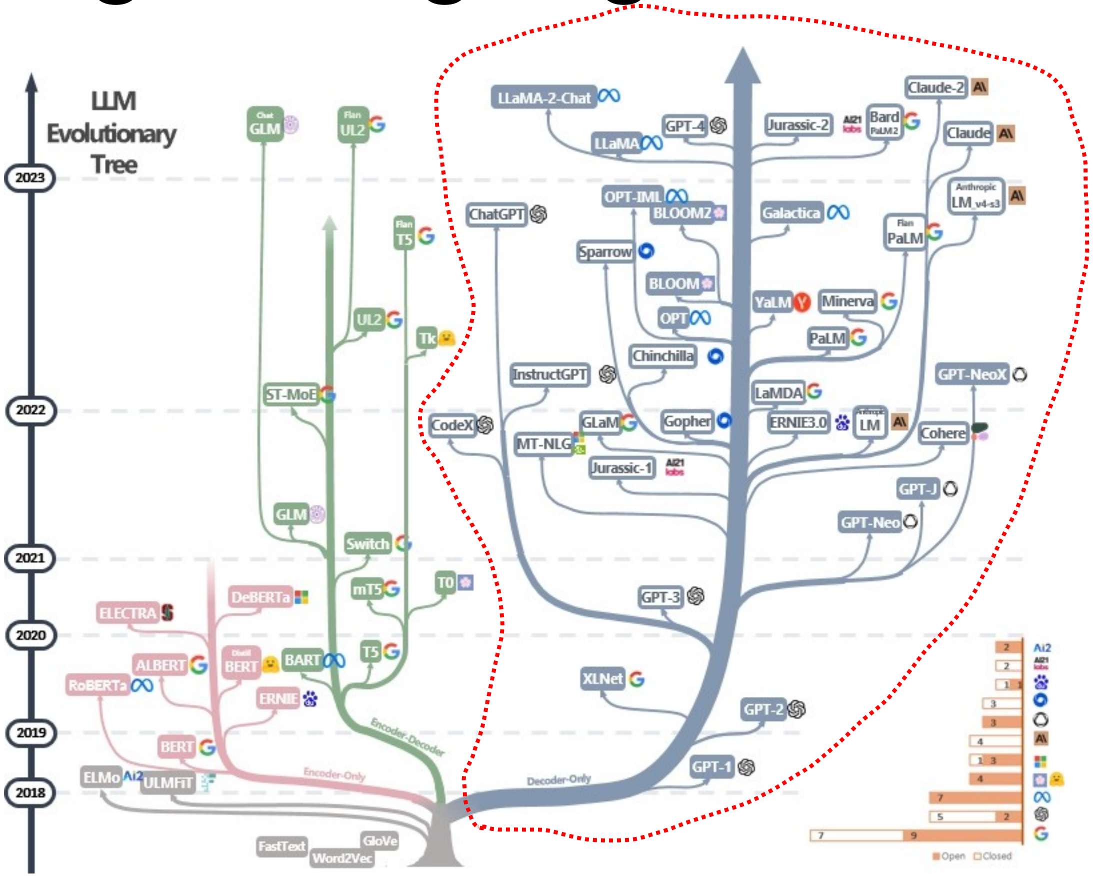


Fig. 1: LLM Evolutionary Tree (https://github.com/Mooler0410/LLMsPracticalGuide)

Large Language Models



Decoder Models
(this lecture)

Fig. 1: LLM Evolutionary Tree (https://github.com/Mooler0410/LLMsPracticalGuide)

Transformer Architecture

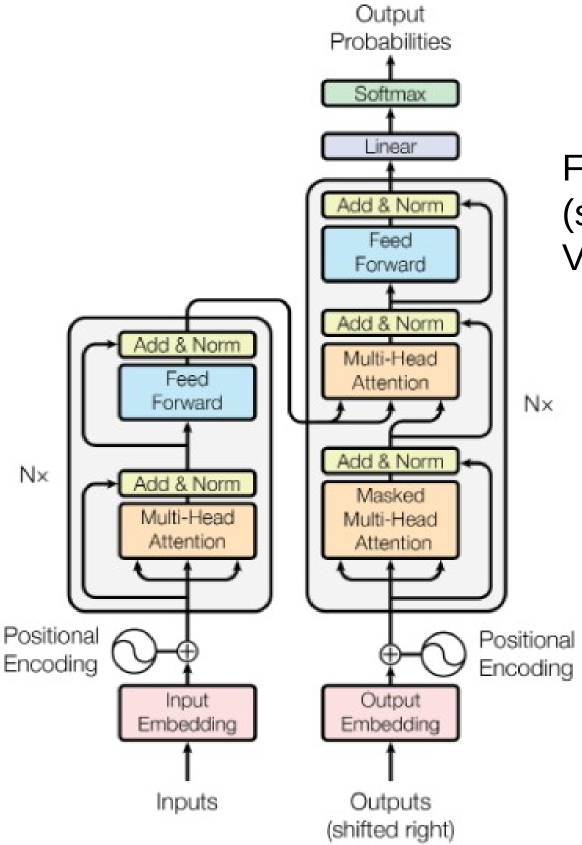


Fig. 2: Transformer Encoder Decoder (source: Attention is all you need. Vaswani et al. 2017)

Transformer Architecture

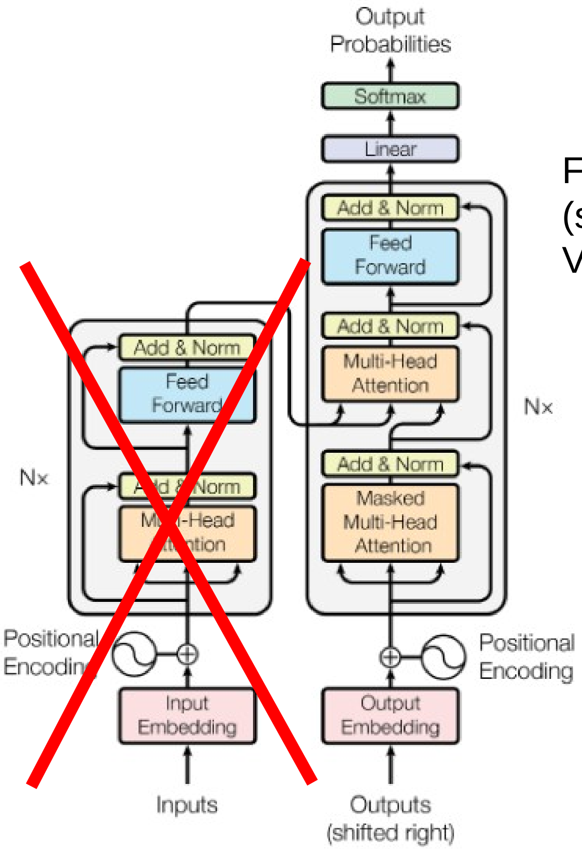
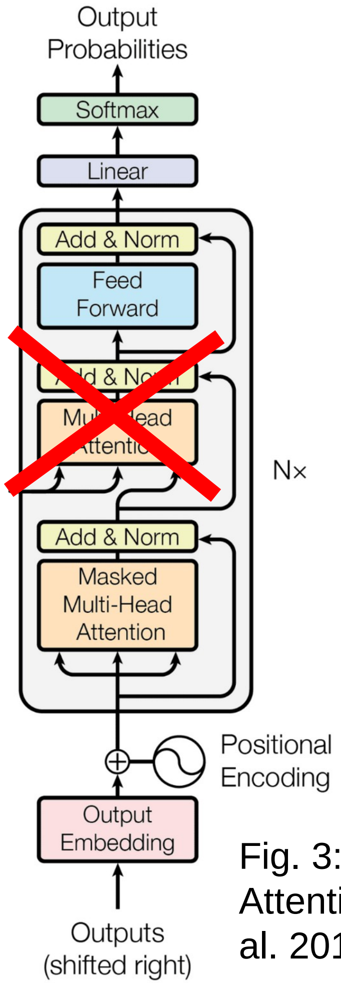


Fig. 2: Transformer Encoder Decoder (source: Attention is all you need. Vaswani et al. 2017)

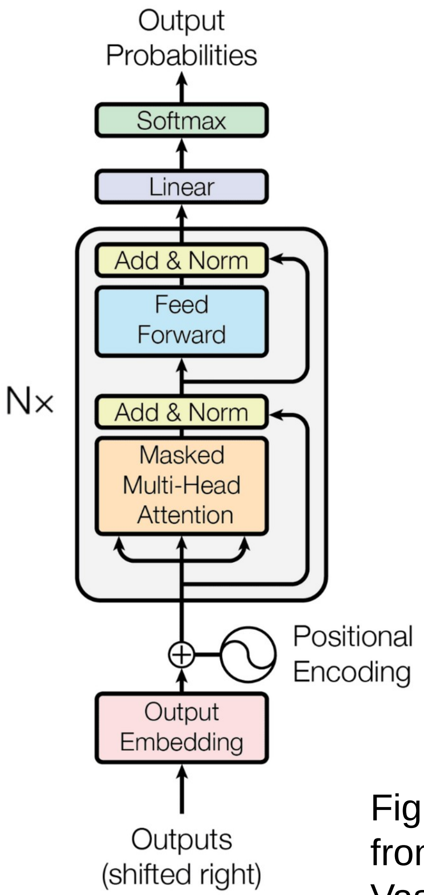
Decoder



- We do not have an encoder so we throw away the Multi-Head Attention block

Fig. 3: Transformer Decoder (source: Attention is all you need. Vaswani et al. 2017)

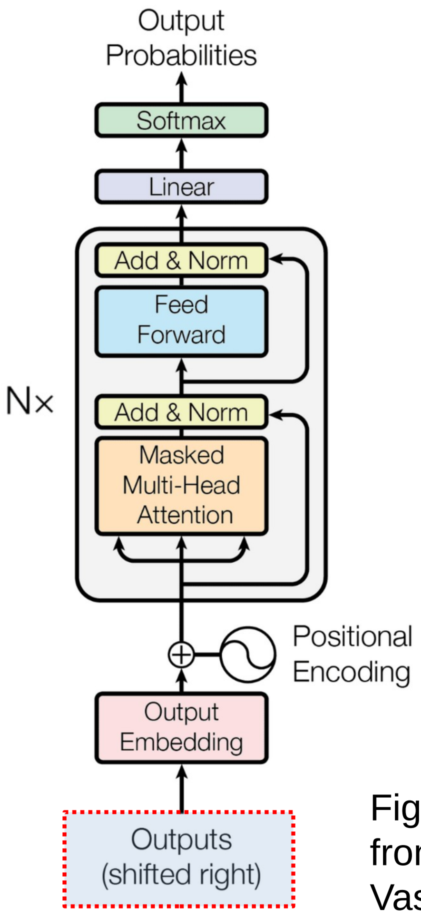
Decoder



- We do not have an encoder so we throw away the Multi-Head Attention block
- This is the basic GPT architecture (generative pretrained transformer)
- This decoder can generate text by predicting the next token

Fig. 4: Transformer Decoder (adapted from: Attention is all you need. Vaswani et al. 2017)

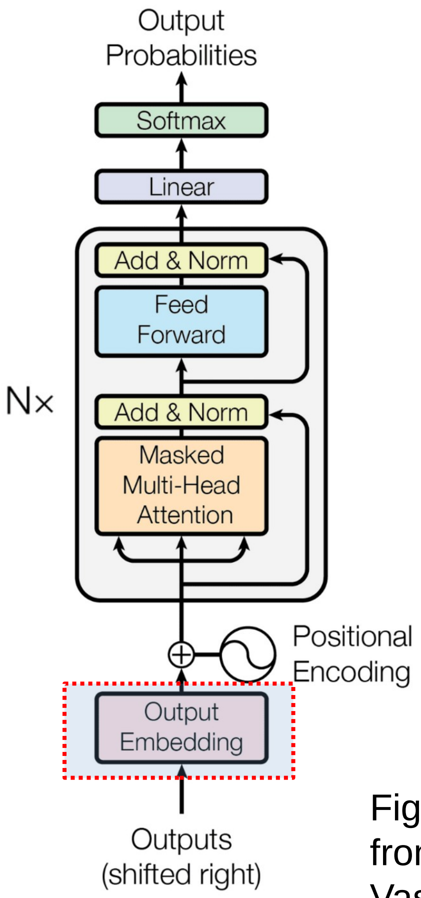
Decoder - Inputs



- We start with a partial sequence.
  Example: <s> Hello
- We tokenize this: [‘<s>’, ‘Hello’]
- We lookup the token ids: [0, 245]

Fig. 4: Transformer Decoder (adapted from: Attention is all you need. Vaswani et al. 2017)

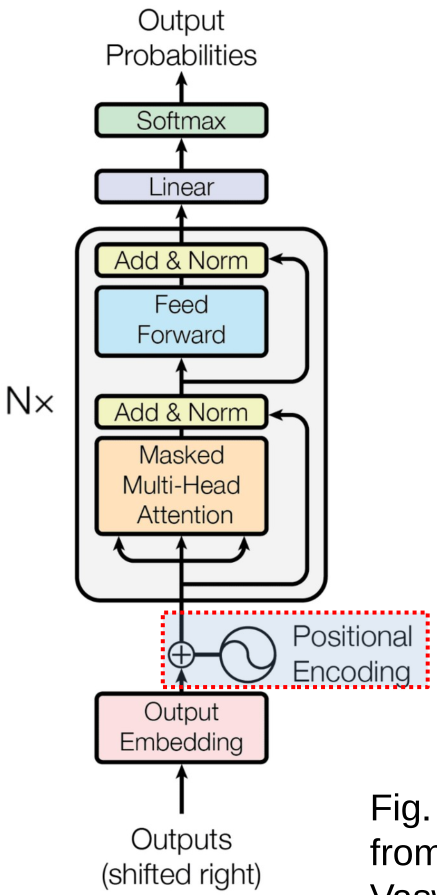
Decoder - Inputs



- We start with a partial sequence. Example: <s> Hello
- We tokenize this: [‘<s>’, ‘Hello’]
- We lookup the token ids: [0, 245]
- We start with initial embeddings: $[E_{<s>}, E_{Hello}]$

Fig. 4: Transformer Decoder (adapted from: Attention is all you need. Vaswani et al. 2017)

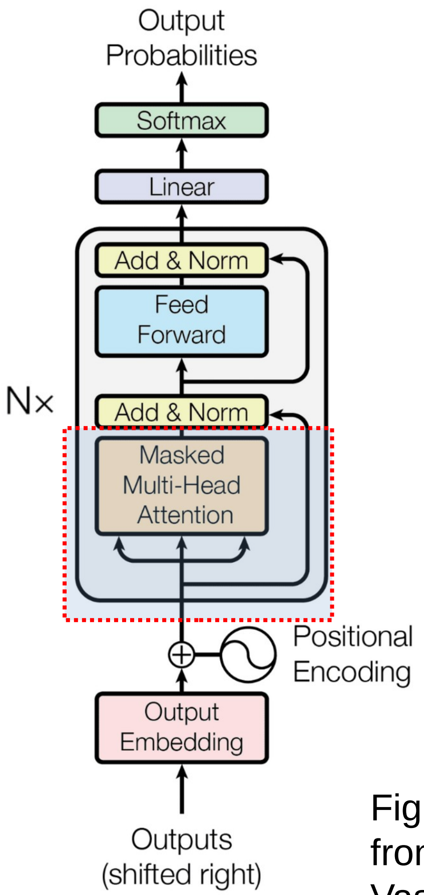
Decoder - Inputs



- We start with a partial sequence. Example: <s> Hello
- We tokenize this: [‘<s>’, ‘Hello’]
- We lookup the token ids: [0, 245]
- We start with initial embeddings: $[y_{<s>}, y_{Hello}]$
- We create the positional embeddings: $[P_0, P_1]$
- We add these together: $[y_{<s>}+P_0,\ y_{Hello}+P_1] = [x_{<s>},\ x_{Hello}]$

Fig. 4: Transformer Decoder (adapted from: Attention is all you need. Vaswani et al. 2017)

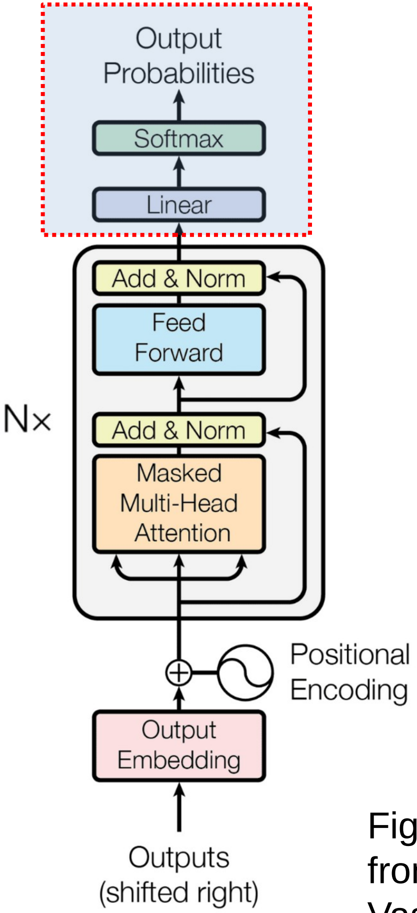
Decoder - Attention



- Our embeddings: [$x_{<s>}$, x_{Hello}]
- Perform Multi-Head Attention:
  Attention weight matrix = QK^T
  These tell us how much information about the other tokens we should take into account for each token.
- We **mask** the attention weights, such that a token can not attend to any subsequent token. In this example the embedding for <s> should not get any information about what comes after it.
  This is called a **causal model**.

Fig. 4: Transformer Decoder (adapted from: Attention is all you need. Vaswani et al. 2017)

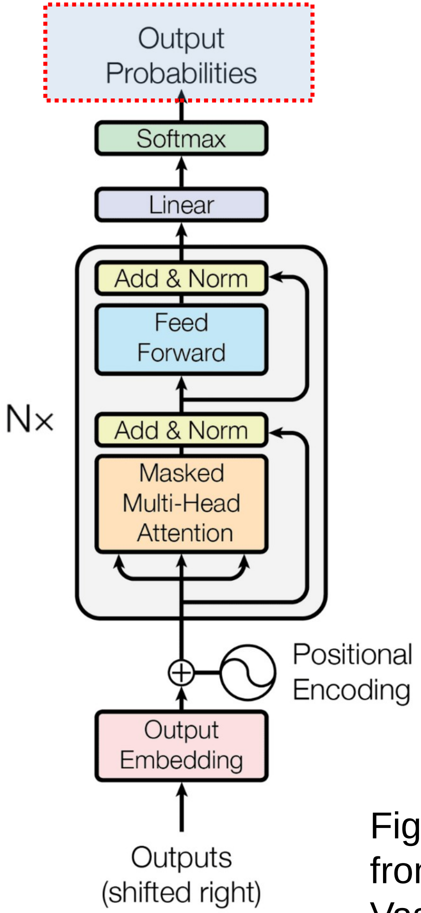
Decoder - Outputs



- After feeding the embeddings through N transformer layers, we get our output embeddings: [$o_{<s>}$, o_{Hello}]. This is a matrix of sequence length times embedding size
- Now we feed this into a linear layer to map this to a vector of vocabulary size.
- Finally we perform a softmax.
- We now have a probability distribitution over all tokens in the vocabulary

Fig. 4: Transformer Decoder (adapted from: Attention is all you need. Vaswani et al. 2017)

Decoder – Next Token

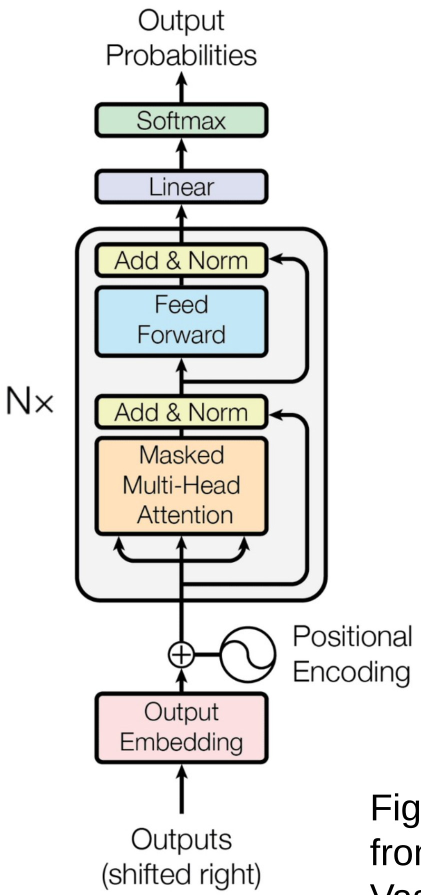


- We now have a probability vector that tells us how likely the next word is token 0, 1, 2, 3, …
- We can either take the argmax or sample from this distribution (roulette wheel sampling)

Next Token ID	**Next Token**	**Probability**
75	World	0.75
5	there	0.12
109	,	0.08
20343	everyone	0.03
3	folks	0.02

Fig. 4: Transformer Decoder (adapted from: Attention is all you need. Vaswani et al. 2017)

Decoder – Next Token

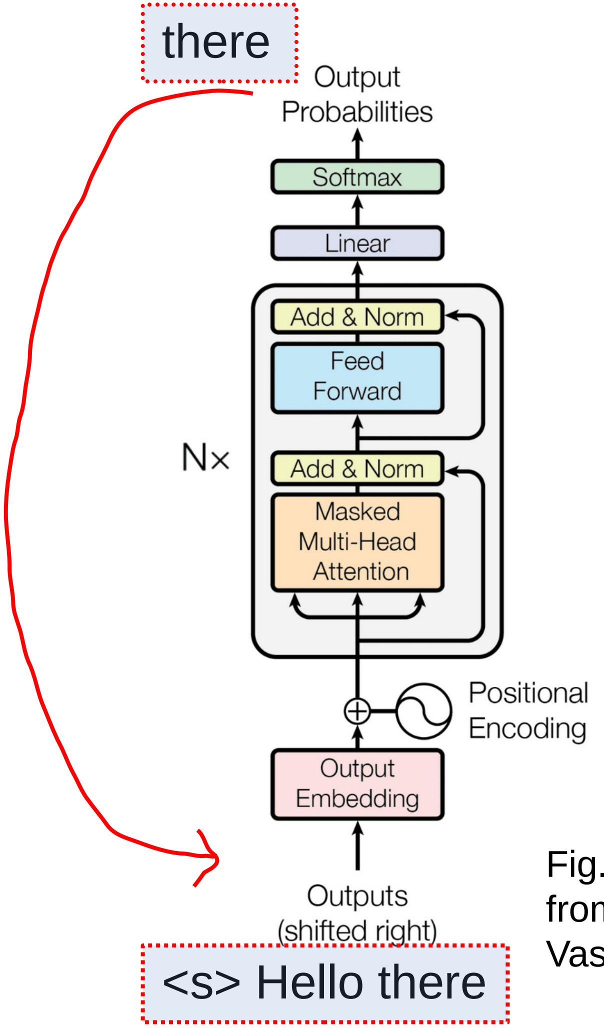


- We now have a probability vector that tells us how likely the next word is token 0, 1, 2, 3, …
- We can either take the argmax or sample from this distribution (roulette wheel sampling)

Next Token ID	Next Token	Probability
75	World	0.75
5	there	0.12
109	,	0.08
20343	everyone	0.03
3	folks	0.02

Fig. 4: Transformer Decoder (adapted from: Attention is all you need. Vaswani et al. 2017)

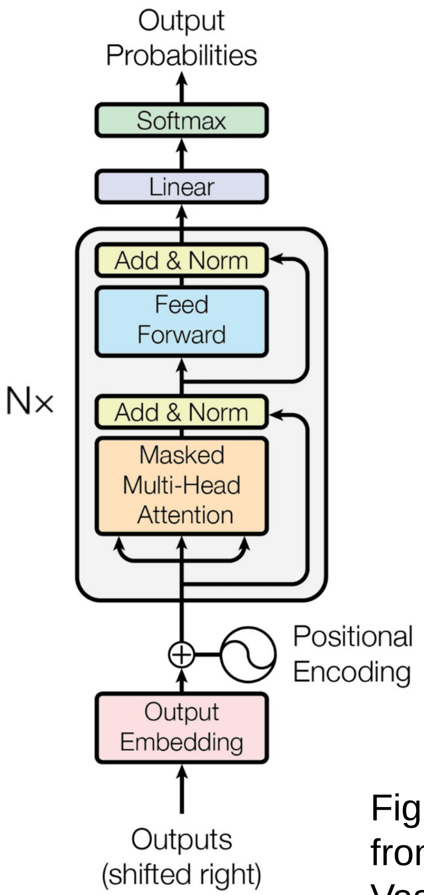
Decoder – Next Iteration



- Assume we picked the token “there” with the id 5 as the next token.
- We append this to the previous input (shifted right)
- We repeat this process until we hit a maximum sequence length or produce the end of sentence token </s>

Fig. 4: Transformer Decoder (adapted from: Attention is all you need. Vaswani et al. 2017)

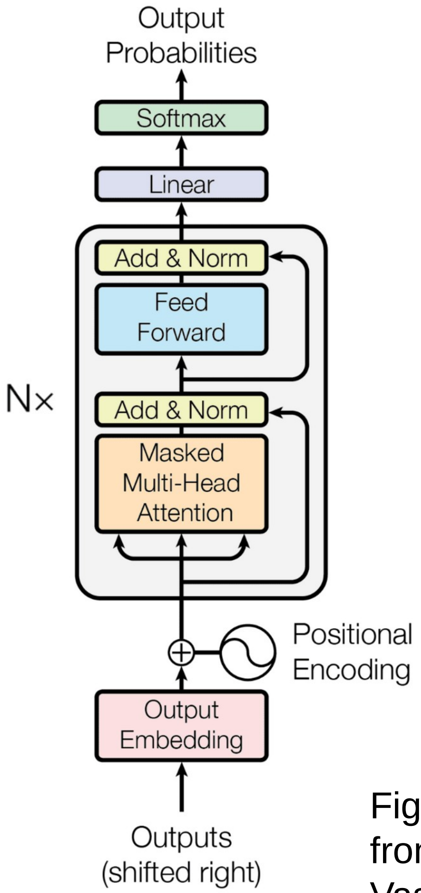
Decoder – Training



- Take a training sequence:
  ”<s> I like to eat pizza </s>”
- Predict the next token from “<s>”
  BCE Loss between the output probabilities and the correct token “I”.
- Feed the sequence “<s> I” into the model.
- Predict the next token
  BCE Loss
- Feed the correct next token to the input:
  ”<s> I like”
- Repeat for all training sequences

Fig. 4: Transformer Decoder (adapted from: Attention is all you need. Vaswani et al. 2017)

Generative Pre-trained Transformer



- GPT-1 (06/2018): 117M parameters
  Trained on BookCorpus (7000 unpublished books)
- GPT-2 (02/2019, 11/2019): 1.5B parameters
  Trained on Web, Reddit, BookCorpus
- GPT-3 (05/2020): 175B parameters
  Trained on filtered Common Crawl, Web, Books, Wikipedia

Fig. 4: Transformer Decoder (adapted from: Attention is all you need. Vaswani et al. 2017)

Generative Pre-trained Transformer

What can these models do?

- Predict the next token
- Translate
- Summarize
- Code generation
- ...

Generative Pre-trained Transformer

What can these models do?

- Predict the next token
- Translate
- Summarize
- Code generation
- ...

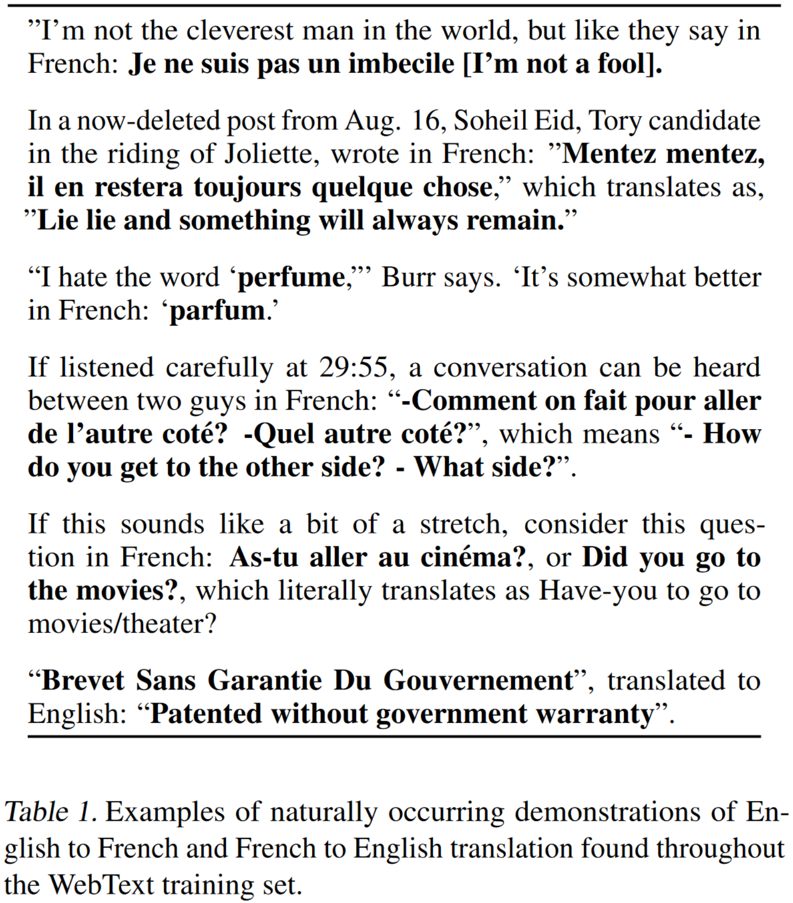
Why can these models do that?

“Language Models are Unsupervised Multitask Learners”
Radford et al (OpenAI), 2019

From the 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. “

“Language Models are Unsupervised Multitask Learners”
Radford et al (OpenAI), 2019

Closer look into the training data:



”I’m not the cleverest man in the world, but like they say in French: **Je ne suis pas un imbecile [I’m not a fool].**

In a now-deleted post from Aug. 16, Soheil Eid, Tory candidate in the riding of Joliette, wrote in French: ”**Mentez mentez, il en restera toujours quelque chose**,” which translates as, ”**Lie lie and something will always remain.**”

“I hate the word ‘**perfume**,”’ Burr says. ‘It’s somewhat better in French: ‘**parfum**.’

If listened carefully at 29:55, a conversation can be heard between two guys in French: “**-Comment on fait pour aller de l’autre coté? -Quel autre coté?**”, which means “**- How do you get to the other side? - What side?**”.

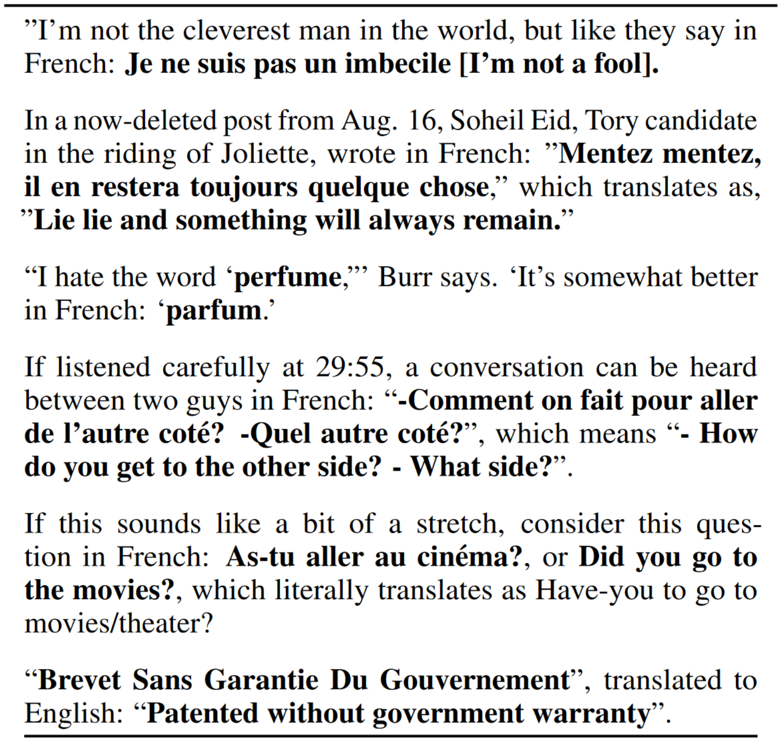
If this sounds like a bit of a stretch, consider this question in French: **As-tu aller au cinéma?**, or **Did you go to the movies?**, which literally translates as Have-you to go to movies/theater?

“**Brevet Sans Garantie Du Gouvernement**”, translated to English: “**Patented without government warranty**”.

Table 1. Examples of naturally occurring demonstrations of English to French and French to English translation found throughout the WebText training set.

“Language Models are Unsupervised Multitask Learners”
Radford et al (OpenAI), 2019

Closer look into the training data:



”I’m not the cleverest man in the world, but like they say in French: **Je ne suis pas un imbecile [I’m not a fool].**

In a now-deleted post from Aug. 16, Soheil Eid, Tory candidate in the riding of Joliette, wrote in French: ”**Mentez mentez, il en restera toujours quelque chose**,” which translates as, ”**Lie lie and something will always remain.**”

“I hate the word ‘**perfume**,”’ Burr says. ‘It’s somewhat better in French: ‘**parfum**.’

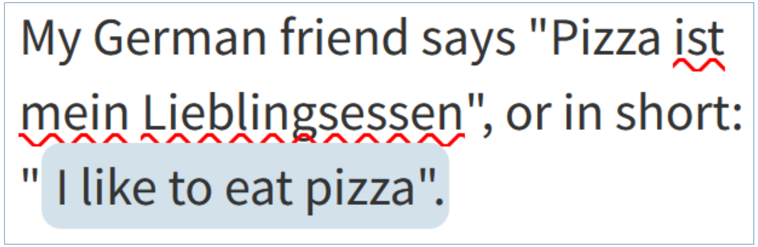
If listened carefully at 29:55, a conversation can be heard between two guys in French: “**-Comment on fait pour aller de l’autre coté? -Quel autre coté?**”, which means “**- How do you get to the other side? - What side?**”.

If this sounds like a bit of a stretch, consider this question in French: **As-tu aller au cinéma?**, or **Did you go to the movies?**, which literally translates as Have-you to go to movies/theater?

“**Brevet Sans Garantie Du Gouvernement**”, translated to English: “**Patented without government warranty**”.

Table 1. Examples of naturally occurring demonstrations of English to French and French to English translation found throughout the WebText training set.

Using this knowledge to translate sentences with GPT2 (https://transformer.huggingface.co/doc/gpt2-large)



My German friend says "Pizza ist mein Lieblingsessen", or in short: " I like to eat pizza".

“Language Models are Unsupervised Multitask Learners”
Radford et al (OpenAI), 2019

Exercise:
Go to this website and try to get the model to perform one of the following tasks:
- Translation
- Sentiment Analysis
- Summarization
- Question Answering

(https://transformer.huggingface.co/doc/gpt2-large)

Instruct GPT

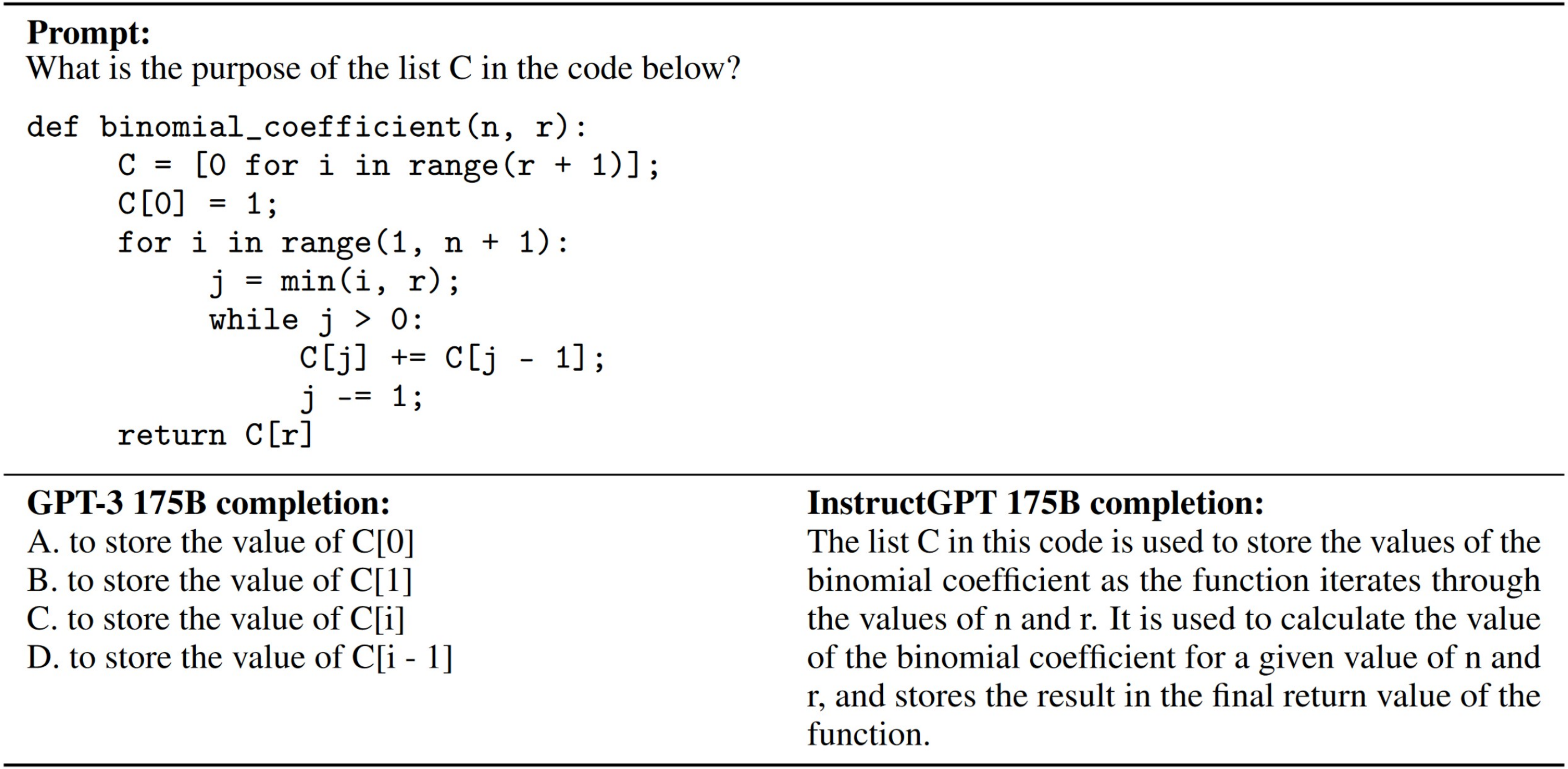
GPT has the ability to perform a variety of tasks but it is hard to engineer the correct prompt.

→ We need to train the model to respond correctly to our prompts:

- Current Prompt:
  My German friend says “Pizza ist mein Lieblingsessen” or in short “

- Desired Prompt:
  Translate “Pizza ist mein Lieblingsessen” to German.

Instruct GPT



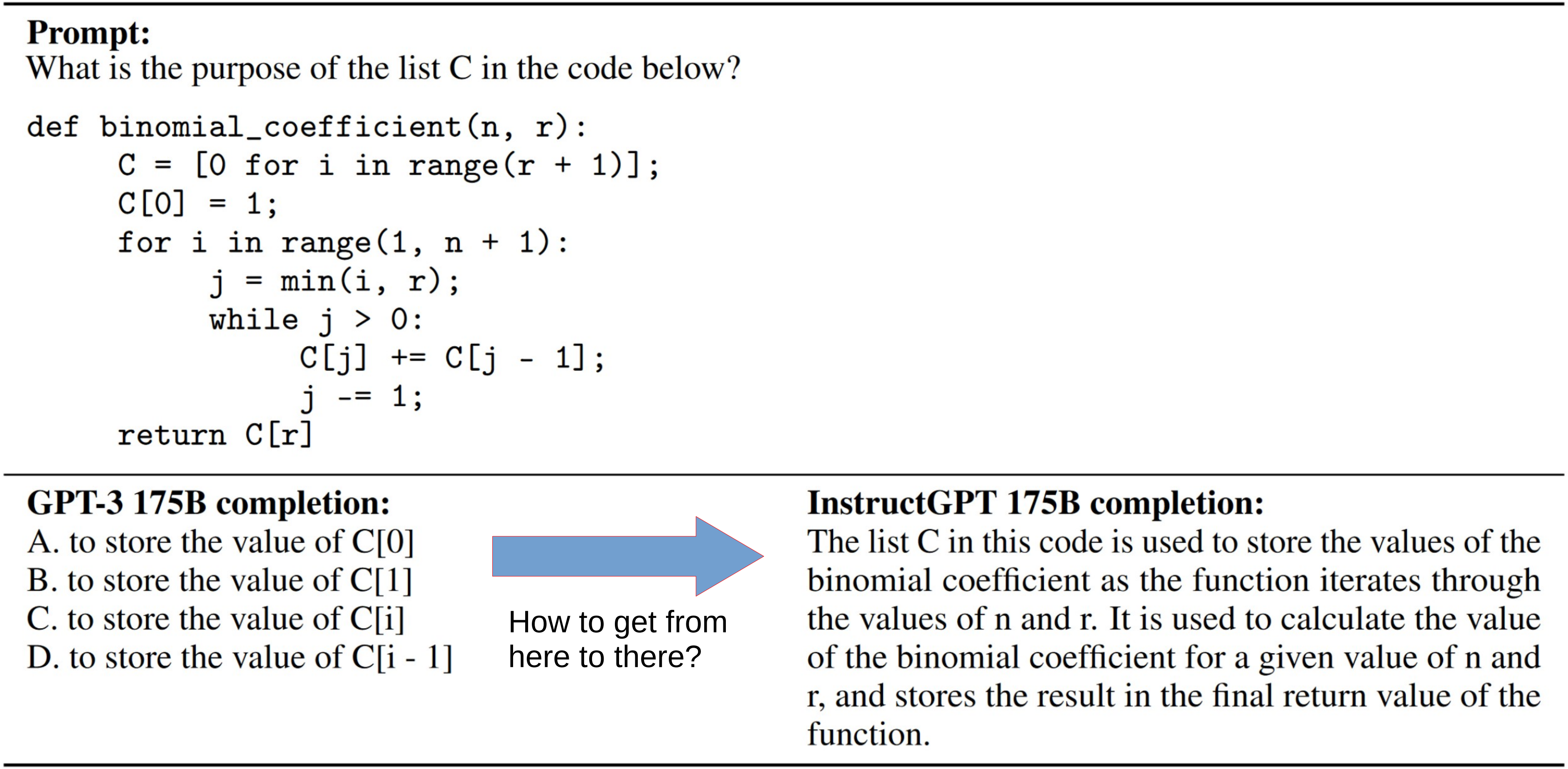
Prompt:
What is the purpose of the list C in the code below?

```
def binomial_coefficient(n, r):
    C = [0 for i in range(r + 1)];
    C[0] = 1;
    for i in range(1, n + 1):
        j = min(i, r);
        while j > 0:
            C[j] += C[j - 1];
            j -= 1;
    return C[r]
```

GPT-3 175B completion:
A. to store the value of C[0]
B. to store the value of C[1]
C. to store the value of C[i]
D. to store the value of C[i - 1]

InstructGPT 175B completion:
The list C in this code is used to store the values of the binomial coefficient as the function iterates through the values of n and r. It is used to calculate the value of the binomial coefficient for a given value of n and r, and stores the result in the final return value of the function.

Instruct GPT



Prompt:
What is the purpose of the list C in the code below?

```
def binomial_coefficient(n, r):
     C = [0 for i in range(r + 1)];
     C[0] = 1;
     for i in range(1, n + 1):
          j = min(i, r);
          while j > 0:
               C[j] += C[j - 1];
               j -= 1;
     return C[r]
```

GPT-3 175B completion:
A. to store the value of C[0]
B. to store the value of C[1]
C. to store the value of C[i]
D. to store the value of C[i - 1]

How to get from here to there?

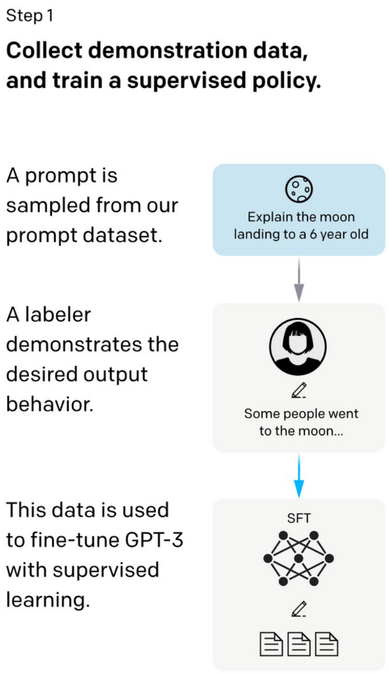
InstructGPT 175B completion:
The list C in this code is used to store the values of the binomial coefficient as the function iterates through the values of n and r. It is used to calculate the value of the binomial coefficient for a given value of n and r, and stores the result in the final return value of the function.

“Training language models to follow instructions with human feedback” Ouyang et al (OpenAI), 2021

“These models are not *aligned* with their user.”

→ The model might produce content that is not helpful for users, toxic or untruthful. So they need to be aligned to produce the desired output.

“Training language models to follow instructions with human feedback”
Ouyang et al (OpenAI), 2021



Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

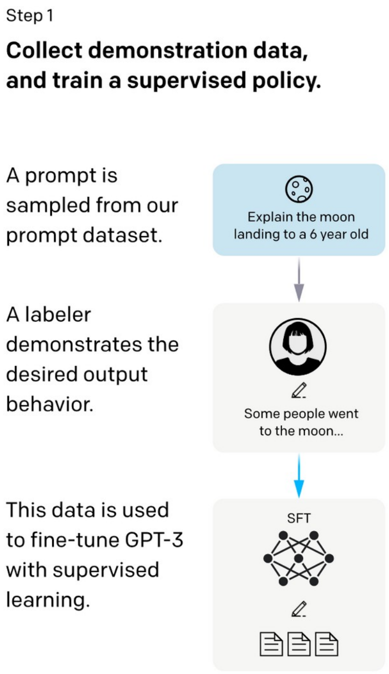
A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.

Explain the moon landing to a 6 year old
Some people went to the moon...
SFT

Figure 2: A diagram illustrating the three steps of our method: (1) supervised fine-tuning (SFT), (2)

“Training language models to follow instructions with human feedback” Ouyang et al (OpenAI), 2021



Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

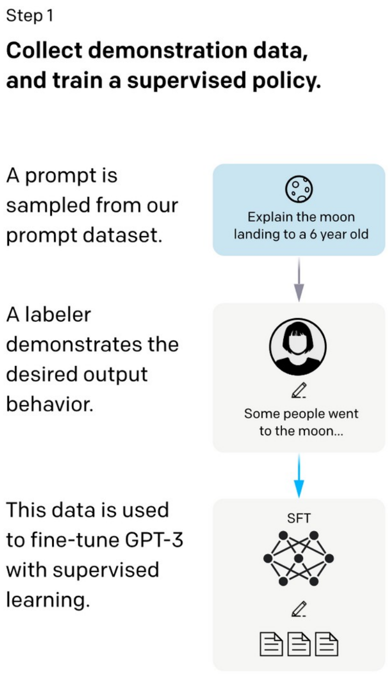
This data is used to fine-tune GPT-3 with supervised learning.

Explain the moon landing to a 6 year old
Some people went to the moon...
SFT

OpenAI hired 40 contractors to label data. They were selected based on a screening test.

Figure 2: A diagram illustrating the three steps of our method: (1) supervised fine-tuning (SFT), (2)

“Training language models to follow instructions with human feedback” Ouyang et al (OpenAI), 2021



Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.

Explain the moon landing to a 6 year old
Some people went to the moon...
SFT

OpenAI hired 40 contractors to label data. They were selected based on a screening test.

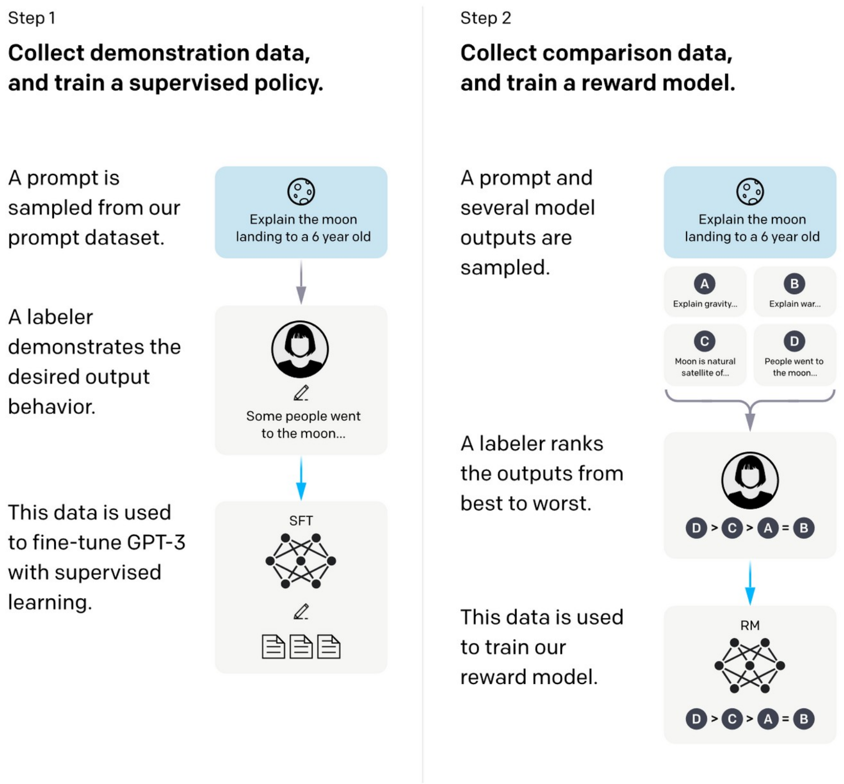
<u>Problems:</u>

This is an expensive manual process.

There are no good large prompt datasets.

Figure 2: A diagram illustrating the three steps of our method: (1) supervised fine-tuning (SFT), (2)

“Training language models to follow instructions with human feedback” Ouyang et al (OpenAI), 2021



Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.

Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

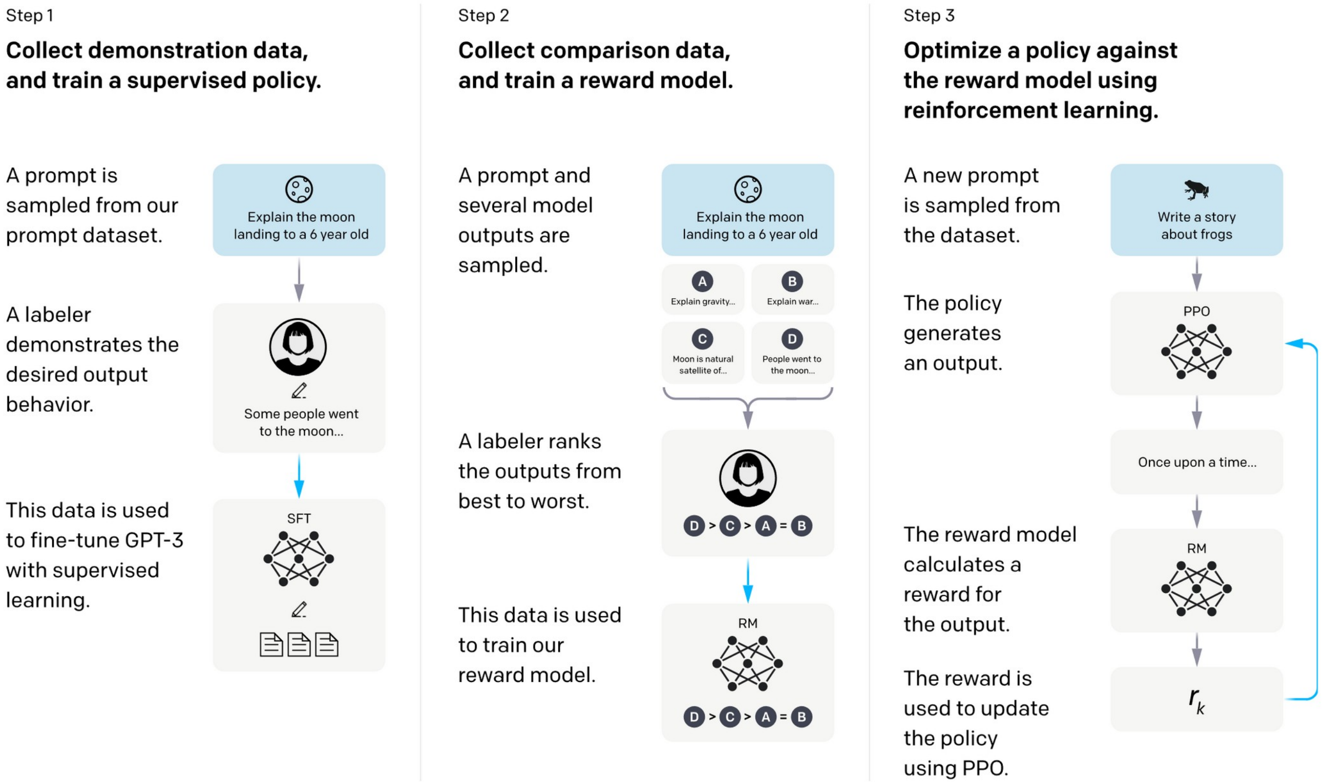
This data is used to train our reward model.

Explain the moon landing to a 6 year old
Some people went to the moon...
SFT
Explain the moon landing to a 6 year old
A
Explain gravity...
B
Explain war...
C
Moon is natural satellite of...
D
People went to the moon...
D > C > A = B
RM
D > C > A = B

Training a reward model to predict the ranking allows for large scale training.

Figure 2: A diagram illustrating the three steps of our method: (1) supervised fine-tuning (SFT), (2) reward model (RM) training,

“Training language models to follow instructions with human feedback” Ouyang et al (OpenAI), 2021



Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.

Explain the moon landing to a 6 year old
Some people went to the moon...
SFT

Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.

Explain the moon landing to a 6 year old
A
Explain gravity...
B
Explain war...
C
Moon is natural satellite of...
D
People went to the moon...
D > C > A = B
RM
D > C > A = B

Step 3

Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.

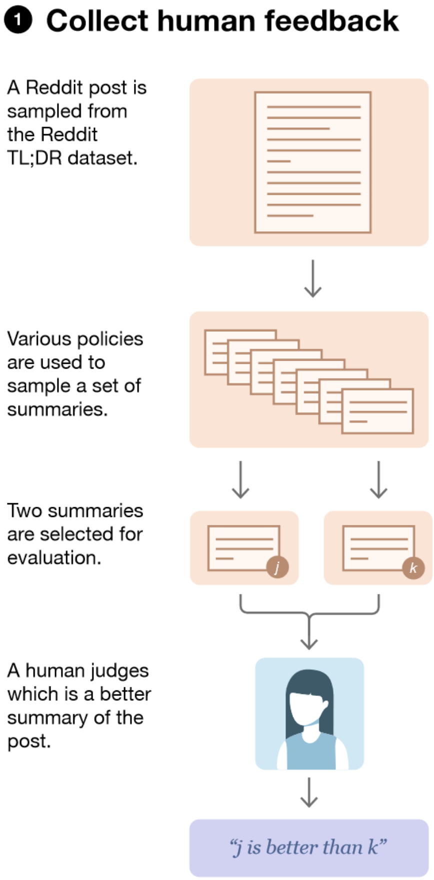
Figure 2: A diagram illustrating the three steps of our method: (1) supervised fine-tuning (SFT), (2) reward model (RM) training, and (3) reinforcement learning via proximal policy optimization (PPO) on this reward model. Blue arrows indicate that this data is used to train one of our models. In Step 2, boxes A-D are samples from our models that get ranked by labelers. See Section 3 for more details on our method.

“Learning to summarize from human feedback”
Stiennon et al (OpenAI), 2022

Train several policies using the Reddit TL;DR (too long, didn’t read) dataset.

Task: Produce a summary from a text

“Learning to summarize from human feedback”
Stiennon et al (OpenAI), 2022



❶ **Collect human feedback**

A Reddit post is sampled from the Reddit TL;DR dataset.

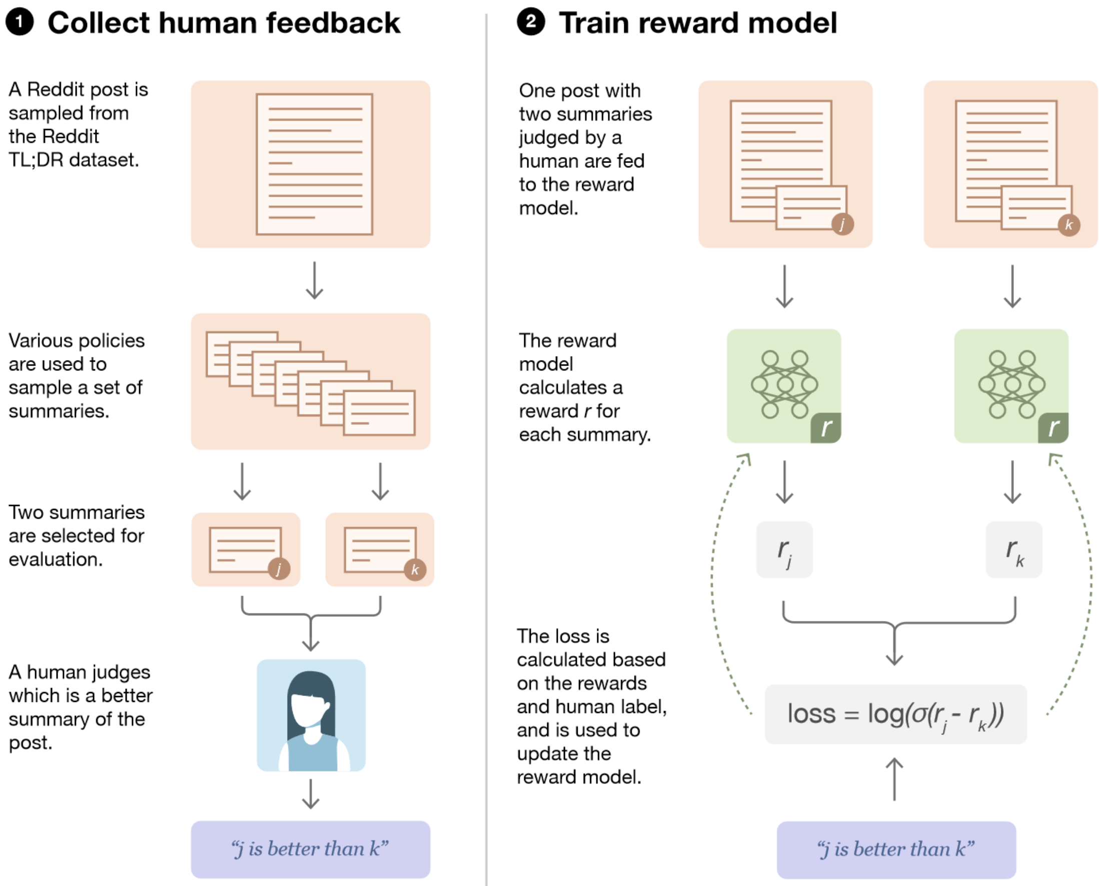
Various policies are used to sample a set of summaries.

Two summaries are selected for evaluation.

A human judges which is a better summary of the post.

Figure 2: Diagram of our human feedback, reward model training, and policy training procedure.

“Learning to summarize from human feedback” Stiennon et al (OpenAI), 2022



❶ **Collect human feedback**

A Reddit post is sampled from the Reddit TL;DR dataset.

Various policies are used to sample a set of summaries.

Two summaries are selected for evaluation.

A human judges which is a better summary of the post.

❷ **Train reward model**

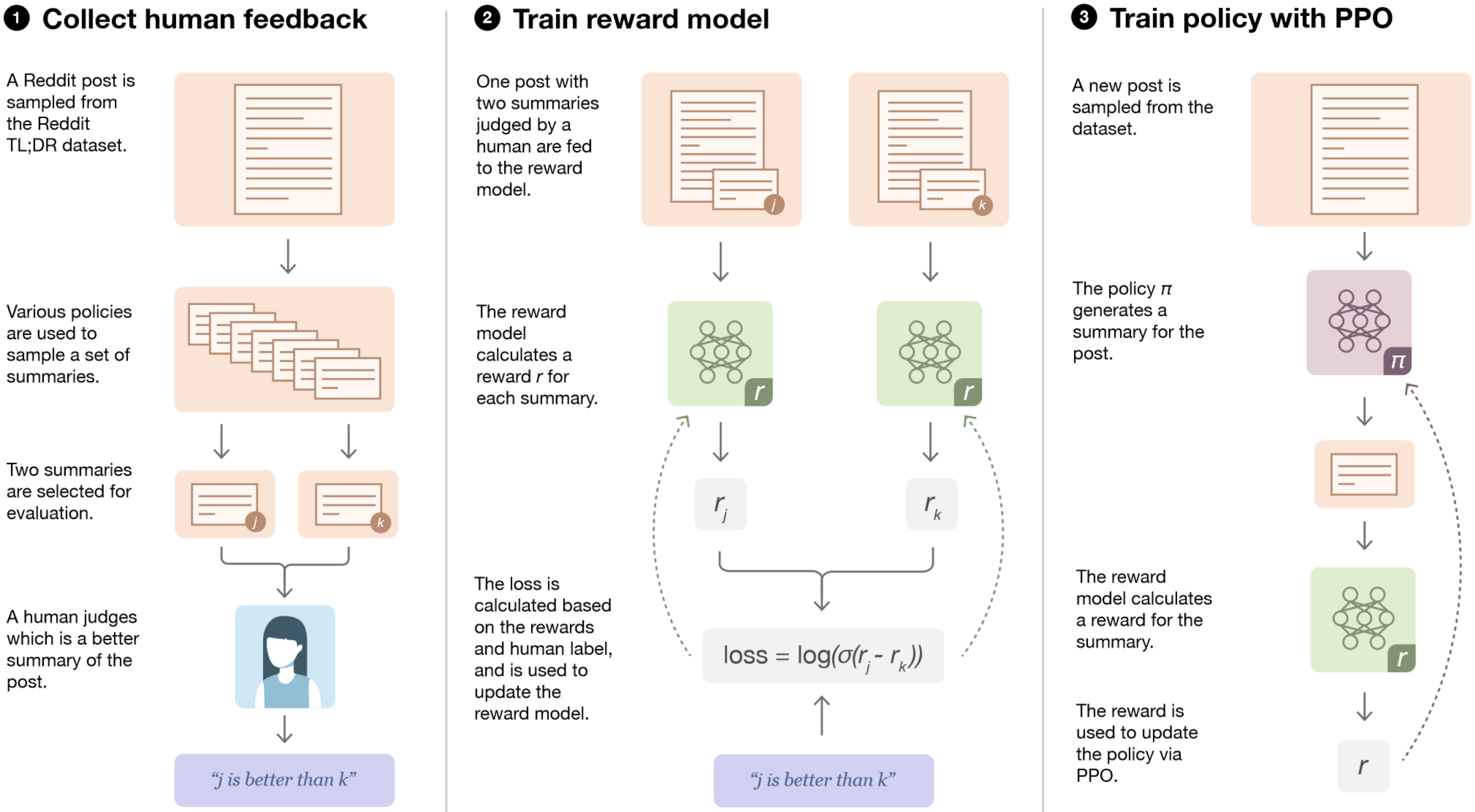
One post with two summaries judged by a human are fed to the reward model.

The reward model calculates a reward *r* for each summary.

The loss is calculated based on the rewards and human label, and is used to update the reward model.

Figure 2: Diagram of our human feedback, reward model training, and policy training procedure.

“Learning to summarize from human feedback”
Stiennon et al (OpenAI), 2022



1 Collect human feedback

A Reddit post is sampled from the Reddit TL;DR dataset.

Various policies are used to sample a set of summaries.

Two summaries are selected for evaluation.

A human judges which is a better summary of the post.

2 Train reward model

One post with two summaries judged by a human are fed to the reward model.

The reward model calculates a reward *r* for each summary.

The loss is calculated based on the rewards and human label, and is used to update the reward model.

3 Train policy with PPO

A new post is sampled from the dataset.

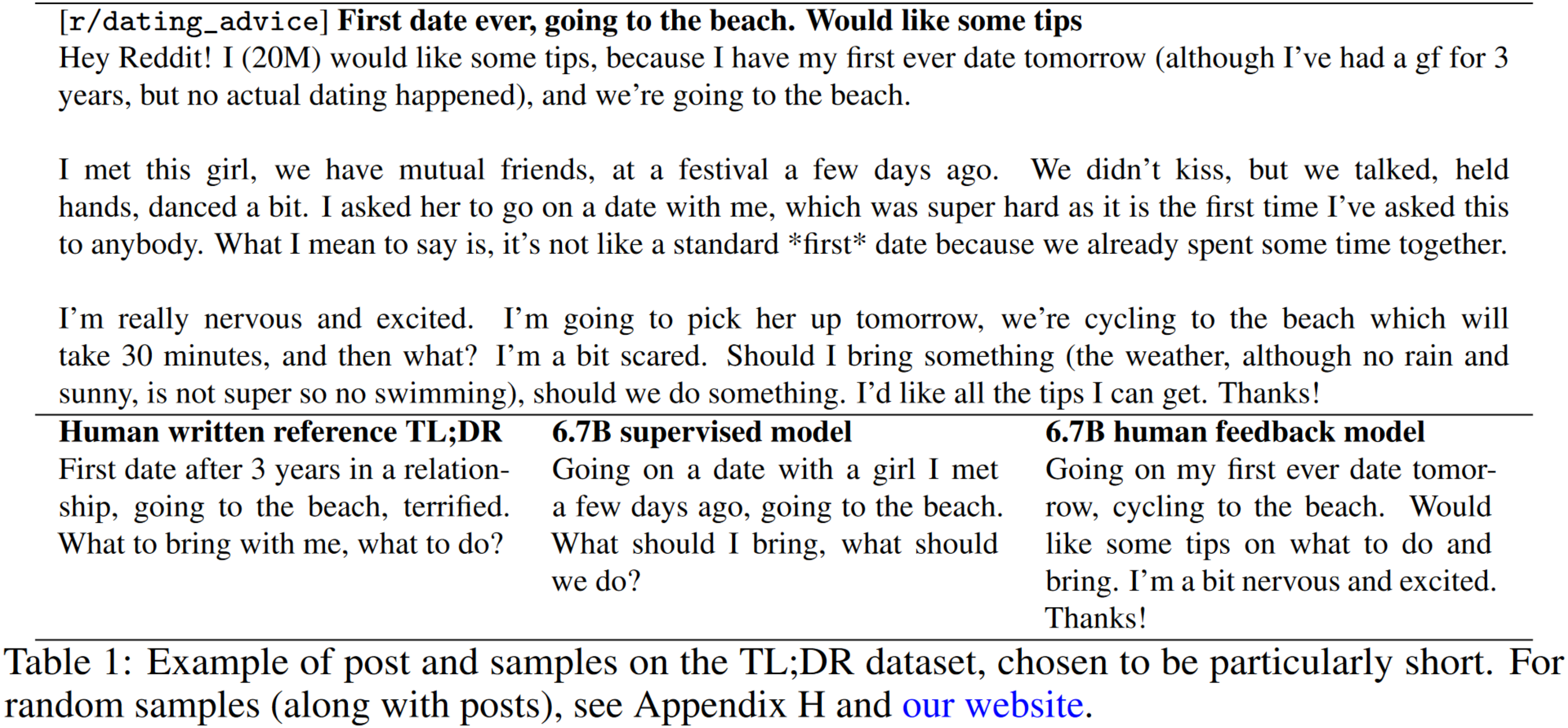
The policy π generates a summary for the post.

The reward model calculates a reward for the summary.

The reward is used to update the policy via PPO.

Figure 2: Diagram of our human feedback, reward model training, and policy training procedure.

“Learning to summarize from human feedback” Stiennon et al (OpenAI), 2022



[r/dating_advice] **First date ever, going to the beach. Would like some tips**

Hey Reddit! I (20M) would like some tips, because I have my first ever date tomorrow (although I’ve had a gf for 3 years, but no actual dating happened), and we’re going to the beach.

I met this girl, we have mutual friends, at a festival a few days ago. We didn’t kiss, but we talked, held hands, danced a bit. I asked her to go on a date with me, which was super hard as it is the first time I’ve asked this to anybody. What I mean to say is, it’s not like a standard *first* date because we already spent some time together.

I’m really nervous and excited. I’m going to pick her up tomorrow, we’re cycling to the beach which will take 30 minutes, and then what? I’m a bit scared. Should I bring something (the weather, although no rain and sunny, is not super so no swimming), should we do something. I’d like all the tips I can get. Thanks!

Human written reference TL;DR	**6.7B supervised model**	**6.7B human feedback model**
First date after 3 years in a relationship, going to the beach, terrified. What to bring with me, what to do?	Going on a date with a girl I met a few days ago, going to the beach. What should I bring, what should we do?	Going on my first ever date tomorrow, cycling to the beach. Would like some tips on what to do and bring. I’m a bit nervous and excited. Thanks!

Table 1: Example of post and samples on the TL;DR dataset, chosen to be particularly short. For random samples (along with posts), see Appendix H and our website.

Cost of training these models

Llama 2:
7B model: 184,320 GPU hours, 400W / GPU hour, 31 tons of CO_2 emitted.
13B model: 368,640 GPU hours, 400W / GPU hour, 62 tons of CO_2 emitted.
70B model: 1,720,320 GPU hours, 400W / GPU hour, 291 tons of CO_2 emitted.

Power consumption 70B:
688,128 kWh
This is the power consumption of ~344 two person house holds in Germany for a year. And you haven’t even paid for the GPUs yet!