CS2109S Tutorial 9

Attention Mechanism and Transformer

(AY 25/26 Semester 2)

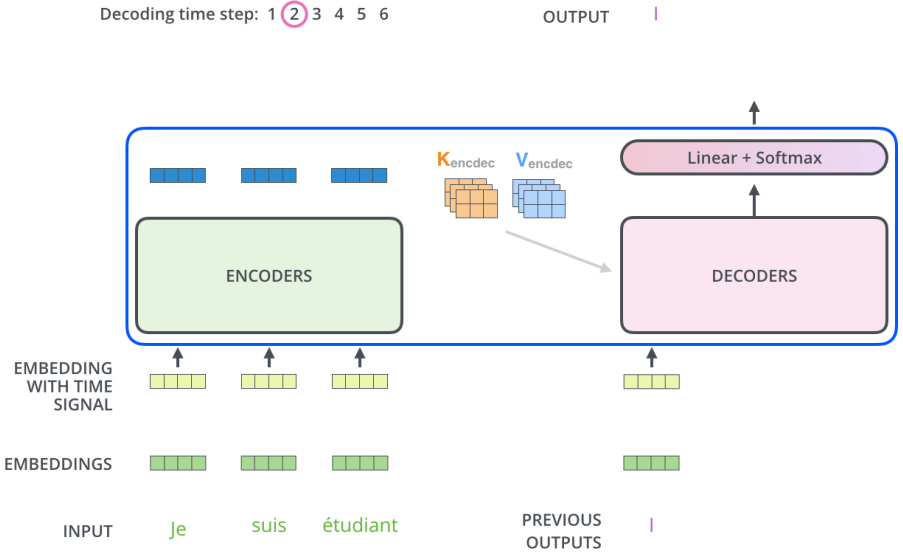
April 9, 2026

(Prepared by Benson)

Contents

Recap: Transformer Architecture

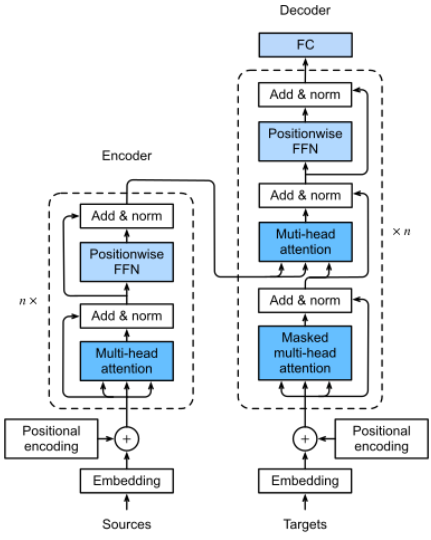
See HTML Slides



Recap: Transformer Architecture

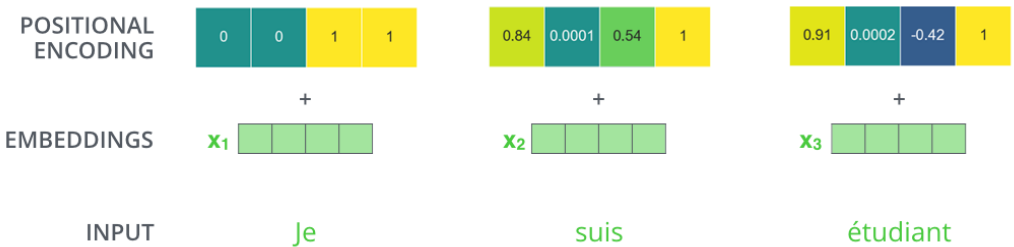
Encoder-Decoder Models:

- Transformer is used by GPT, BERT, etc.
- Note: GPT is a **decoder-only** model!



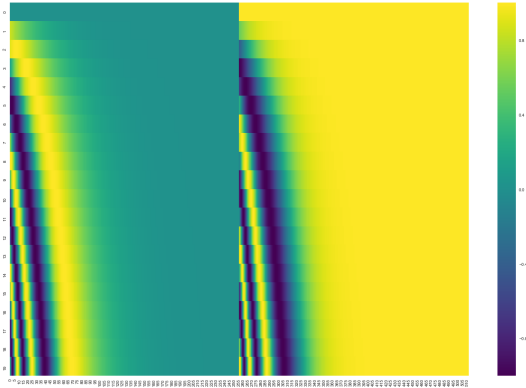
Recap: Transformer Architecture

Token and Positional Embedding

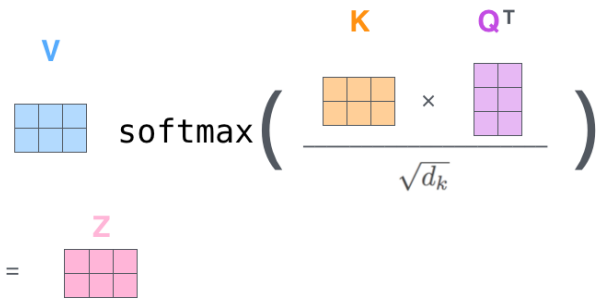


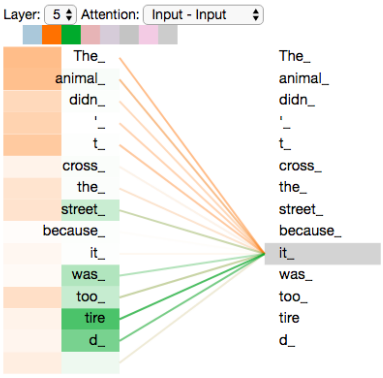
Recap: Transformer Architecture

$$PE(i, 2k) = \sin\left(\frac{i}{10000^{2k/d}}\right), PE(i, 2k+1) = \cos\left(\frac{i}{10000^{2k/d}}\right)$$



Recap: Transformer Architecture





Recap: Transformer Architecture

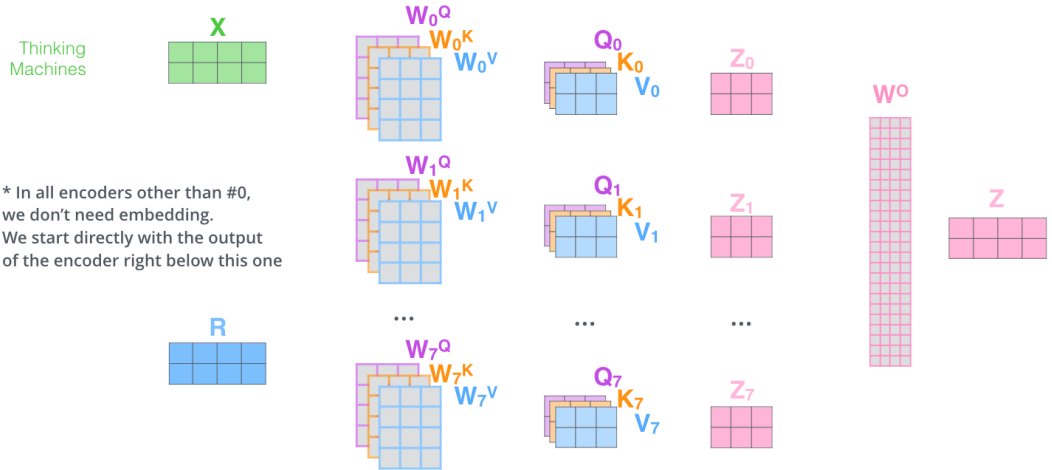
1) This is our input sentence*

2) We embed each word*

3) Split into 8 heads. We multiply X or R with weight matrices

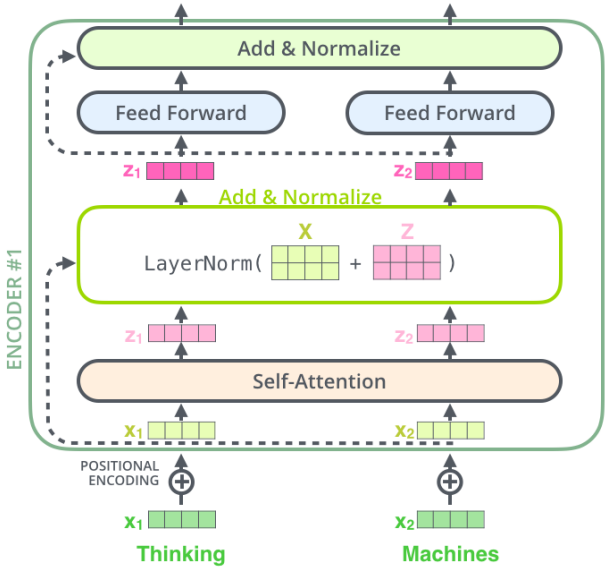
4) Calculate attention using the resulting Q/K/V matrices

5) Concatenate the resulting Z matrices, then multiply with weight matrix W^O to produce the output of the layer

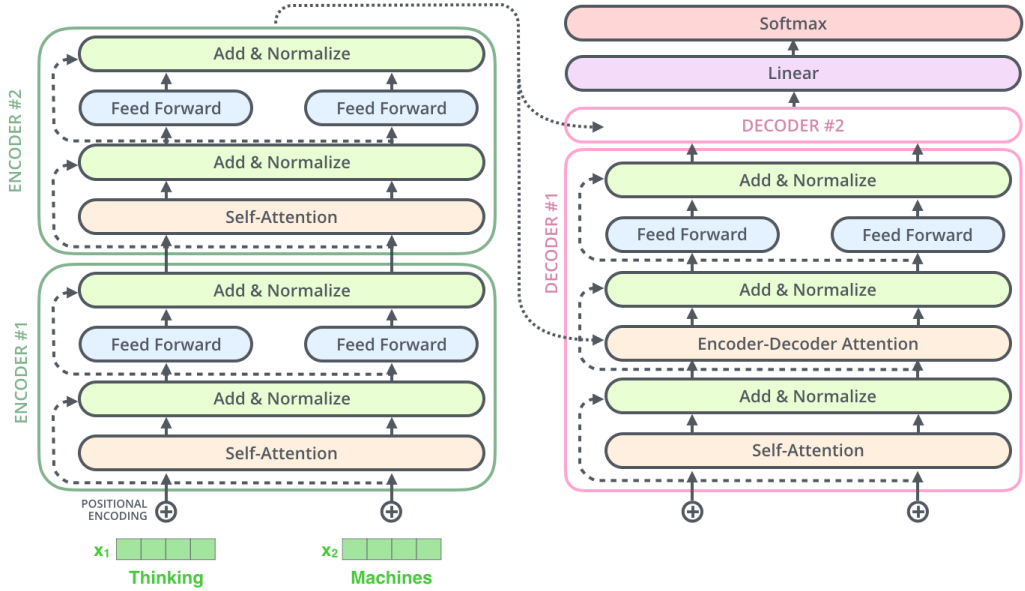


* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one

Recap: Transformer Architecture

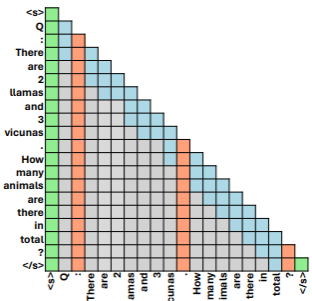


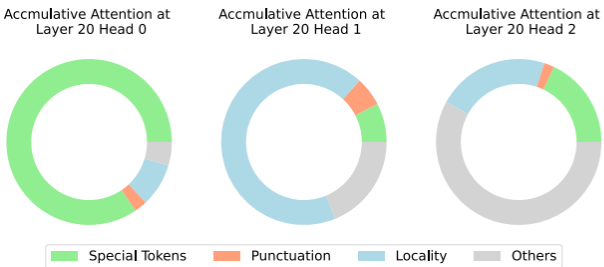
Recap: Transformer Architecture



Recap: Transformer Architecture

"Our findings show that standard attention modules do not provide meaningful explanations and should not be treated as though they do." (Read the Paper)





Q1. Self-Attention

Consider $\boldsymbol{X} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}, \boldsymbol{W}^q = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \boldsymbol{W}^k = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \boldsymbol{W}^v = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$.

(a) Compute the corresponding query, key and value matrices $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}$.

$$\boldsymbol{Q} = \boldsymbol{W}^q \boldsymbol{X} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 2 \end{bmatrix}$$

$$\boldsymbol{K} = \boldsymbol{W}^k \boldsymbol{X} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

$$\boldsymbol{V} = \boldsymbol{W}^v \boldsymbol{X} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 2 \\ 0 & 1 & 1 \end{bmatrix}$$

Q1. Self-Attention

From previous slide: $\boldsymbol{Q} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 2 \end{bmatrix}, \boldsymbol{K} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}, \boldsymbol{V} = \begin{bmatrix} 1 & 1 & 2 \\ 0 & 1 & 1 \end{bmatrix}$.

(b) Compute the attention scores. Identify which word receives the highest attention when the query comes from "cat".

$$\text{softmax}\left(\frac{\boldsymbol{K}^\top \boldsymbol{Q}}{\sqrt{d_k}}\right) = \text{softmax}\left(\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 2 \end{bmatrix}\right)$$

$$= \text{softmax}\left(\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 2 \\ 2 & 1 & 3 \end{bmatrix}\right)$$

$$= \begin{array}{c} \begin{matrix} \text{cat} & \text{monkey} & \text{dog} \end{matrix} \\ \begin{bmatrix} 0.248 & 0 & 0.140 \\ 0.248 & 0.5 & 0.284 \\ 0.503 & 0.5 & 0.576 \end{bmatrix} \end{array} \begin{matrix} \text{cat} \\ \text{monkey} \\ \text{dog} \end{matrix}$$

Q1. Self-Attention

From previous slides: $\boldsymbol{S} = \begin{bmatrix} 0.248 & 0 & 0.140 \\ 0.248 & 0.5 & 0.284 \\ 0.503 & 0.5 & 0.576 \end{bmatrix}$, $\boldsymbol{V} = \begin{bmatrix} 1 & 1 & 2 \\ 0 & 1 & 1 \end{bmatrix}$.

(c) Compute the output of the self-attention layer.

$$
\begin{aligned}
\boldsymbol{H} &= \boldsymbol{V} \cdot \boldsymbol{S} \\
&= \begin{bmatrix} 1 & 1 & 2 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0.248 & 0 & 0.140 \\ 0.248 & 0.5 & 0.284 \\ 0.503 & 0.5 & 0.576 \end{bmatrix} \\
&= \begin{bmatrix} 1.502 & 1.5 & 1.576 \\ 0.751 & 1 & 0.860 \end{bmatrix}
\end{aligned}
$$

Q1. Self-Attention

(d) Assume the input dimension to a self-attention layer is $d_{in} = 512$, and the Query, Key, and Value projections each have dimension $d_q = d_k = d_v = 64$. Compute the total number of trainable parameters in $\boldsymbol{W}^q$, $\boldsymbol{W}^k$, and $\boldsymbol{W}^v$.

- $512 \times 64 \times 3 = \boxed{98304}$.

Q2. Cross-Attention

Consider a English $\rightarrow$ French translation model using the Transformer architecture.

(a) Why must the decoder include self-attention layer in addition to cross-attention to the English encoder states?

- Looks at the words the decoder has already generated, makes use of that context in the generation of the next word, and helps ensure the output is grammatically correct and logically coherent.

Why is this self-attention masked?

- Preventing the model from using information from "future" predicted tokens.

(b) In the cross-attention layer, why must the Query come from Decoder while the Key and Value come from Encoder?

- Query: To ask for information (decoder).
- Key and Value: The "database" of information (encoder).

Q2. Cross-Attention

Consider a English $\rightarrow$ French translation model using the Transformer architecture.

(c) Suppose the implementer mistakenly replaced the cross-attention layer in the decoder with self-attention, what will likely happen to the translation output of this model?

- No connection to the source sentence. Likely: Unrelated French sentences.

Q3. Positional Embedding

Three guidelines for positional embedding:

- **Unique**: Each position should have a distinct encoding.
- **Consistent**: The encoding for a specific position should be the same regardless of the total length of the sequence.
- **Bounded**: The values generated by the positional encoding function stay within a reasonable range.

Consider:

$$\mathbf{PE}_1^{[t]} = \begin{bmatrix} t \\ t \\ \vdots \\ t \end{bmatrix}, \mathbf{PE}_2^{[t]} = \begin{bmatrix} t/T \\ t/T \\ \vdots \\ t/T \end{bmatrix}, \mathbf{PE}_3^{[t]} = \left[\cos\left(\frac{t \times (k+1)}{T_{\max} \times d} \right) \right]_{k=0}^{d-1}$$

Q4. Encoder-Decoder

Determine the most appropriate transformer architecture for the following tasks.

(a) Spam Email Detection
(Input: Email content, Output: Classification of email being a spam email)
- Encoder-Only.

(b) Text Summarisation
(Input: Long text document, Output: Summary of the text)
- Encoder-Decoder.

(c) Story Generation
(Input: A single theme keyword, Output: A generated short story)
- Decoder-Only.

Extra: More Developments

What we haven't covered...

Extra Slide

Types of Innovations in AI (Soumith Chintala, 2018):

- Better neural network structure
- Better optimization scheme
- Define a more clever objective function
  - Semi-supervised or Unsupervised objective function
- Injecting better priors into pre-processing and post-processing
- More Data
- Collecting better data, better labels, cleverer labels

ML Scientist

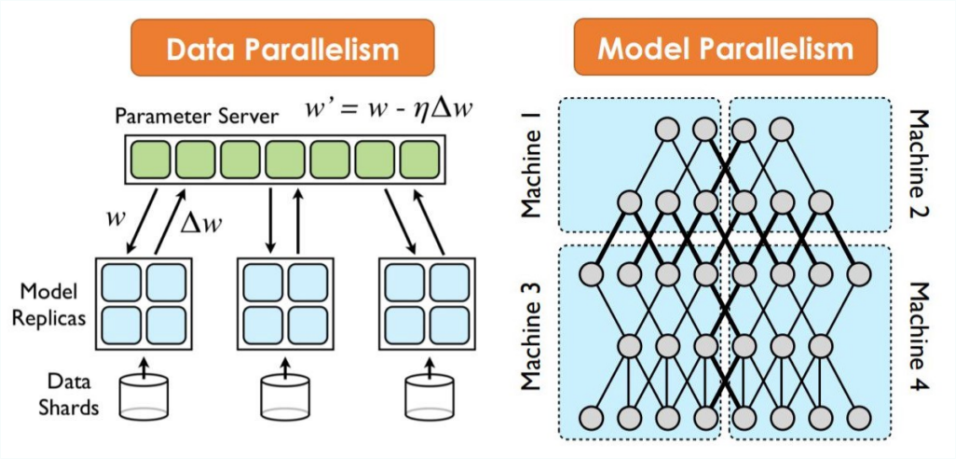
- Scaling to larger neural networks
- Using the hardware more efficiently, or designing more hardware

HPC Scientist

Distributed Training

Extra Slide

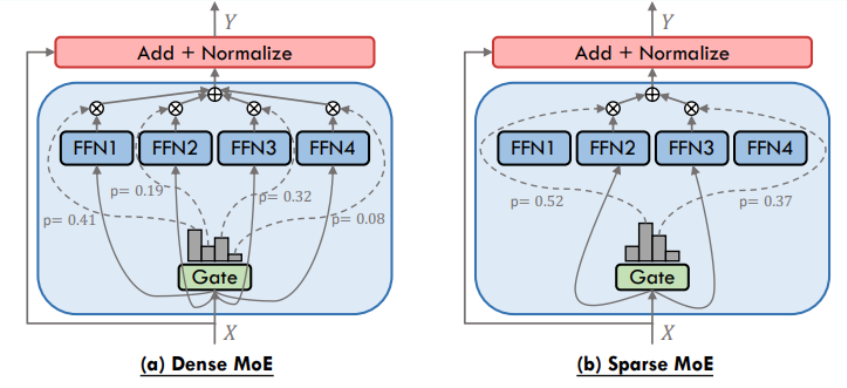
Data Parallelism vs. **Model Parallelism**



Mixture of Experts (MoE)

Extra Slide

Rely on **many small models** instead of **one large model** (Survey Paper)



Distributed Inference / Serving

Extra Slide

Serving Model (example): Notice the similarities with process scheduling (CS2106)!

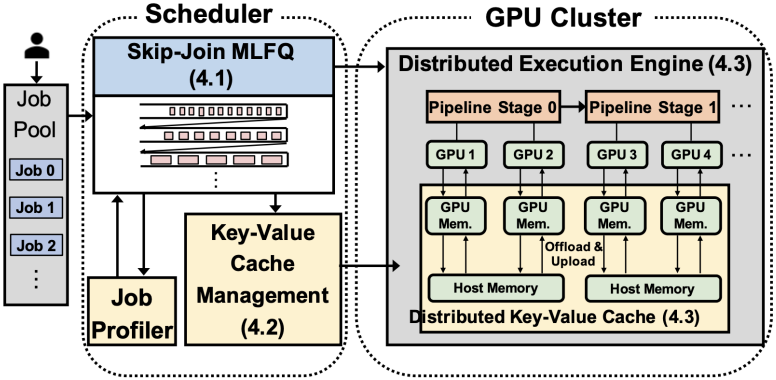
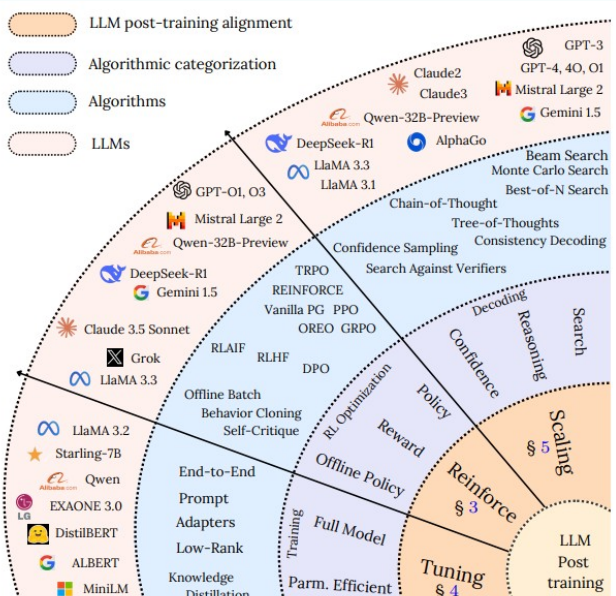


Figure 3. FastServe architecture.

Post-Training Recipes

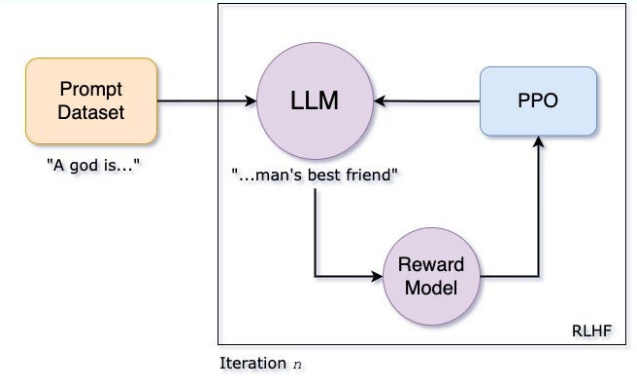
Extra Slide

Survey Paper



Post-Training Recipes

Extra Slide



Summary

If you want to try ML...

Which aspects of ML fascinates you the most?