How LLMs Work & Why Prompt Engineering Matters

ByteMonk
204K subscribers

Join | Subscribe | 88 | Share | Thanks | Clip | ...

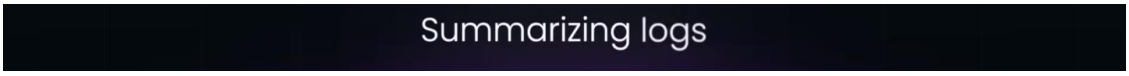
839 views Jun 21, 2025 Members first on June 19, 2025 #GPT4 #PromptEngineering #agenticai

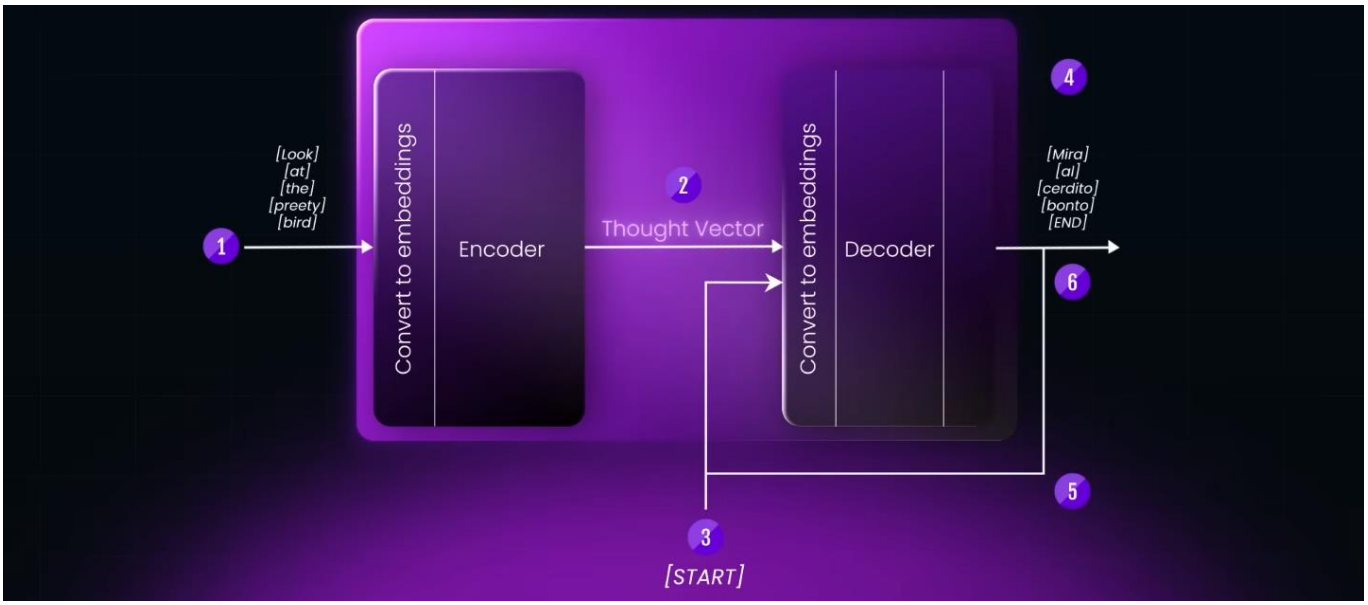
Prompt engineering isn't about tricks — it's about understanding how large language models (LLMs) actually work.

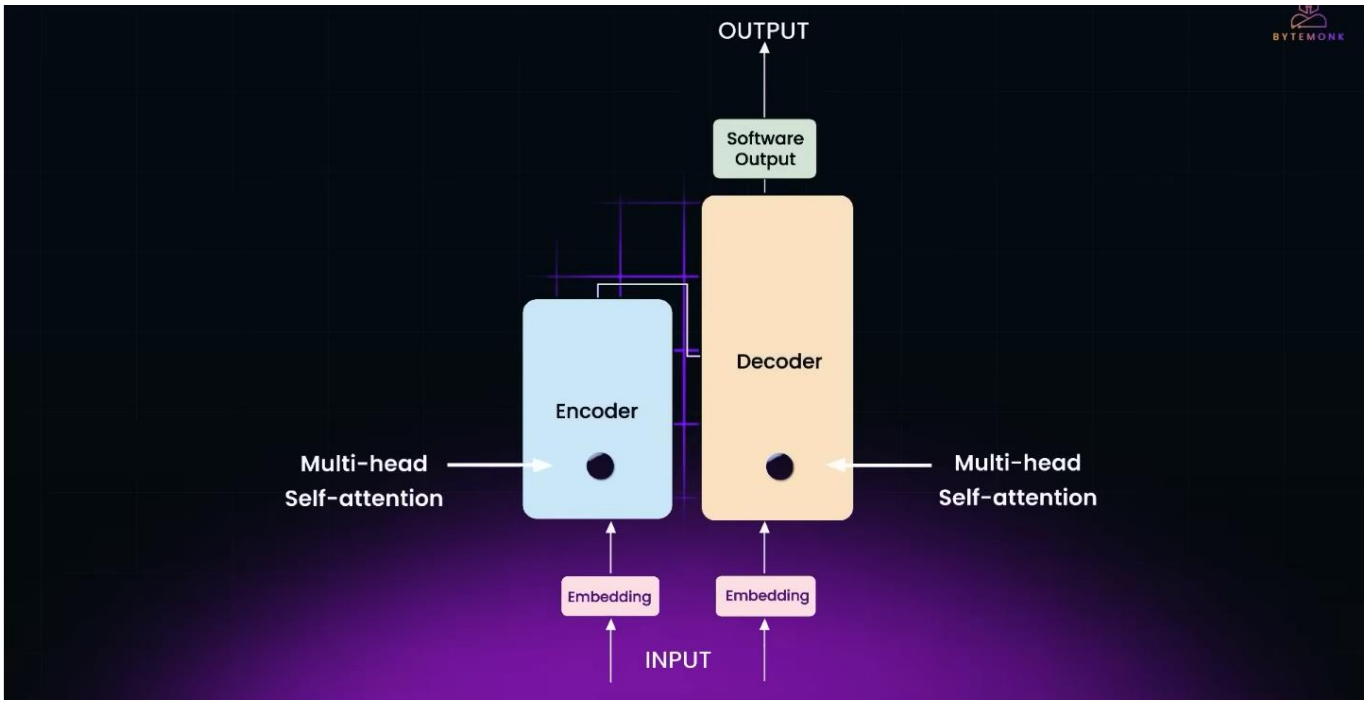
In this first part of the series, we unpack the evolution of LLMs — from Seq2Seq to Transformers — and explain how models generate responses, why prompting replaced fine-tuning, and what it means to design prompts that shape output reliably. We also touch on the system design behind prompt-driven workflows, memory injection, and multi-modal prompts — setting the stage for building real-world AI applications.This is where solid architecture meets practical AI.

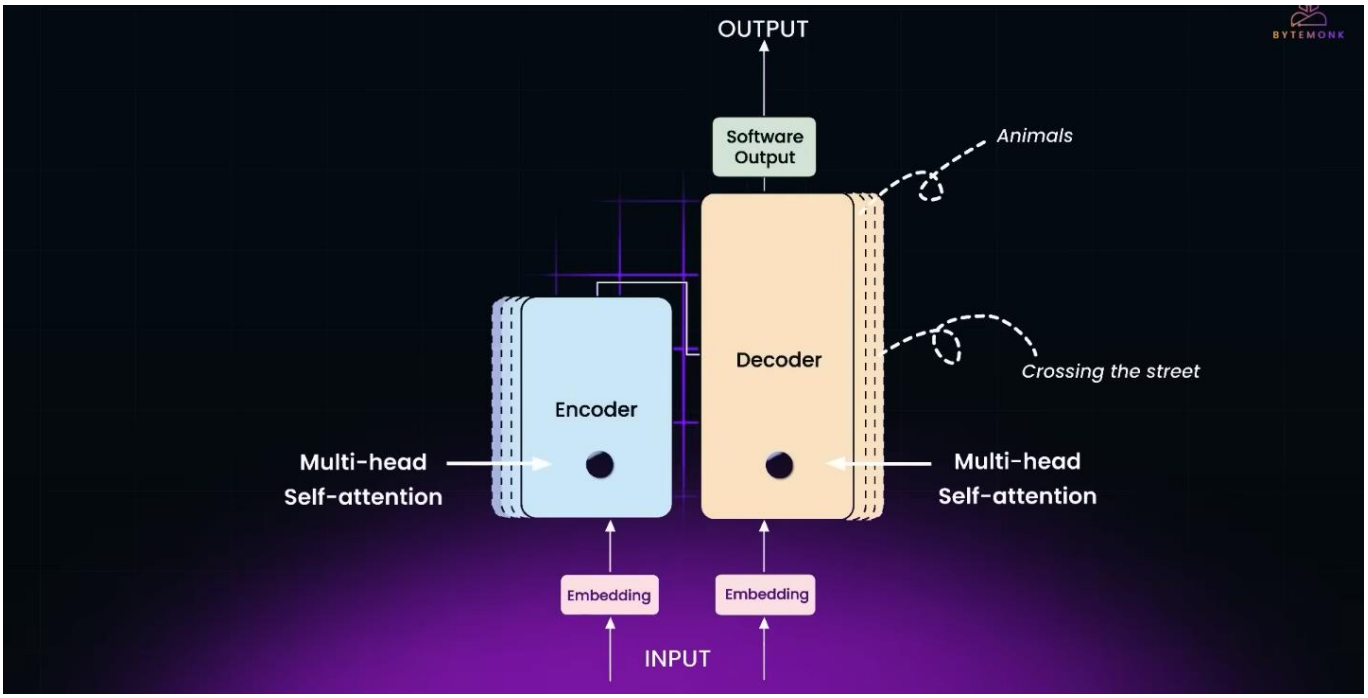
My Linkedin Profile: / bytemonk

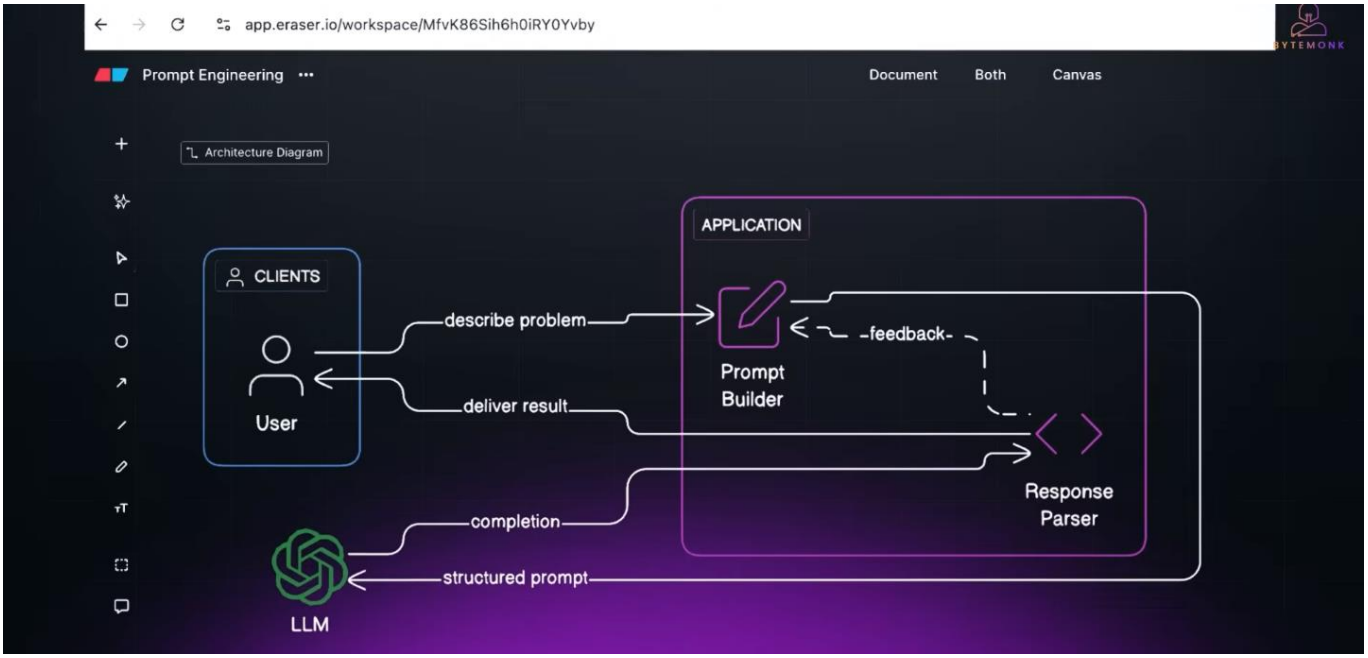
📌 Timestamps

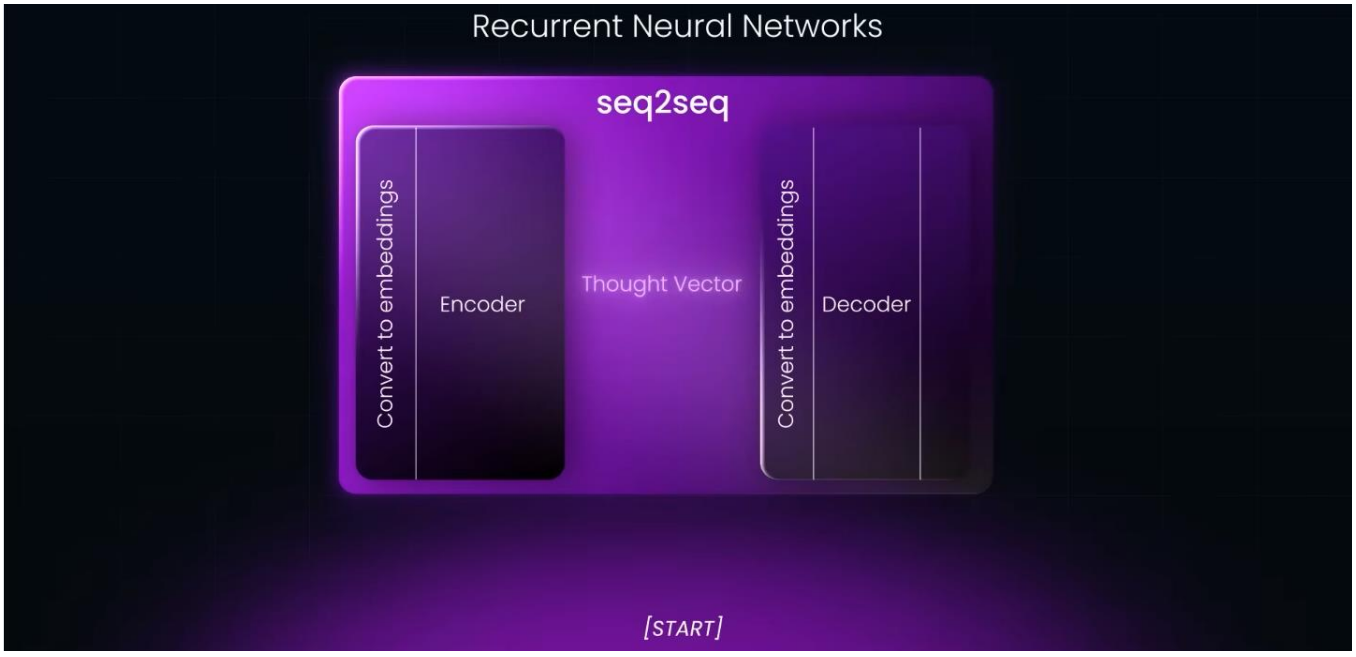



[Look]
[at]
[the]
[preety]
[bird]
1
Convert to embeddings
Encoder
2
Thought Vector
Convert to embeddings
Decoder
4
[Mira]
[a]
[cerdito]
[bonto]
[END]
6
5
3
[START]

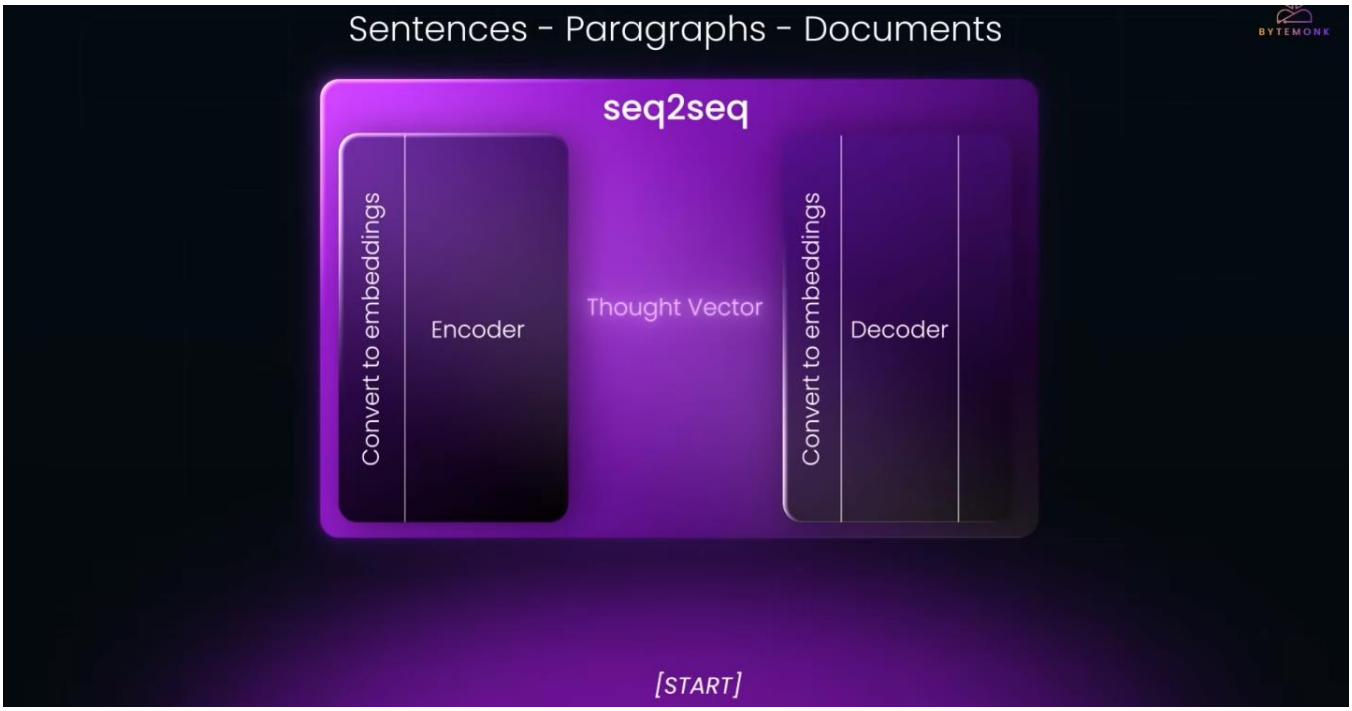

OUTPUT
Software Output
Decoder
Encoder
Multi-head Self-attention
Multi-head Self-attention
Embedding
Embedding
INPUT

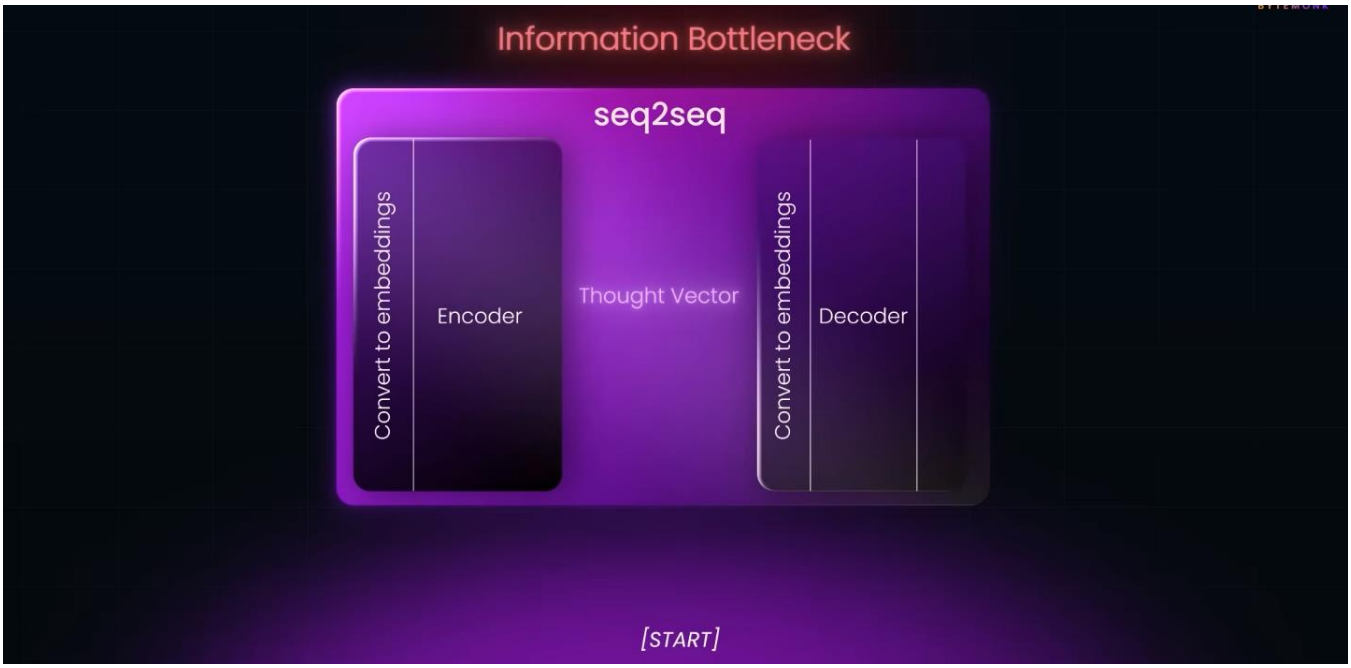

OUTPUT
Software Output
Animals
Decoder
Crossing the street
Encoder
Multi-head Self-attention
Multi-head Self-attention
Embedding
Embedding
INPUT

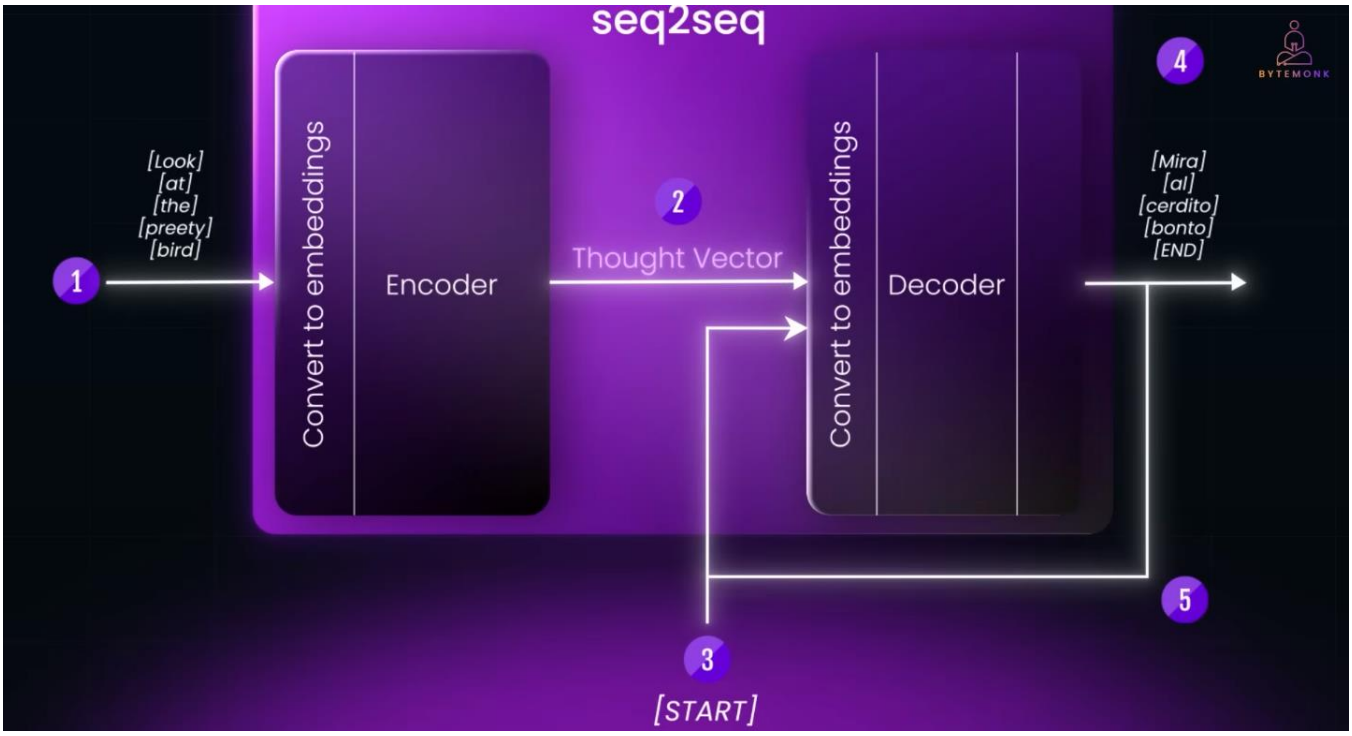

app.eraser.io/workspace/MfvK86Sih6h0iRY0Yvby
Prompt Engineering
Document
Both
Canvas
Architecture Diagram
APPLICATION
CLIENTS
describe problem
-feedback-
Prompt Builder
User
deliver result
Response Parser
completion
structured prompt
LLM

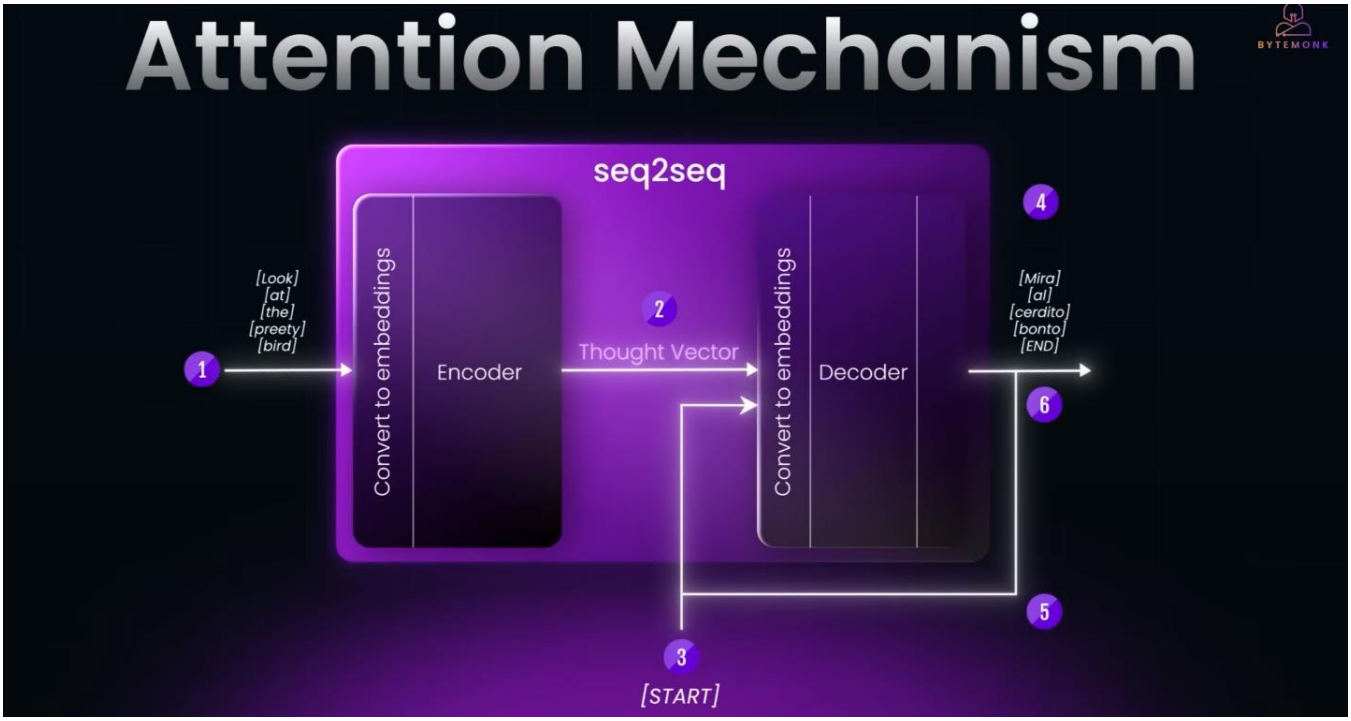


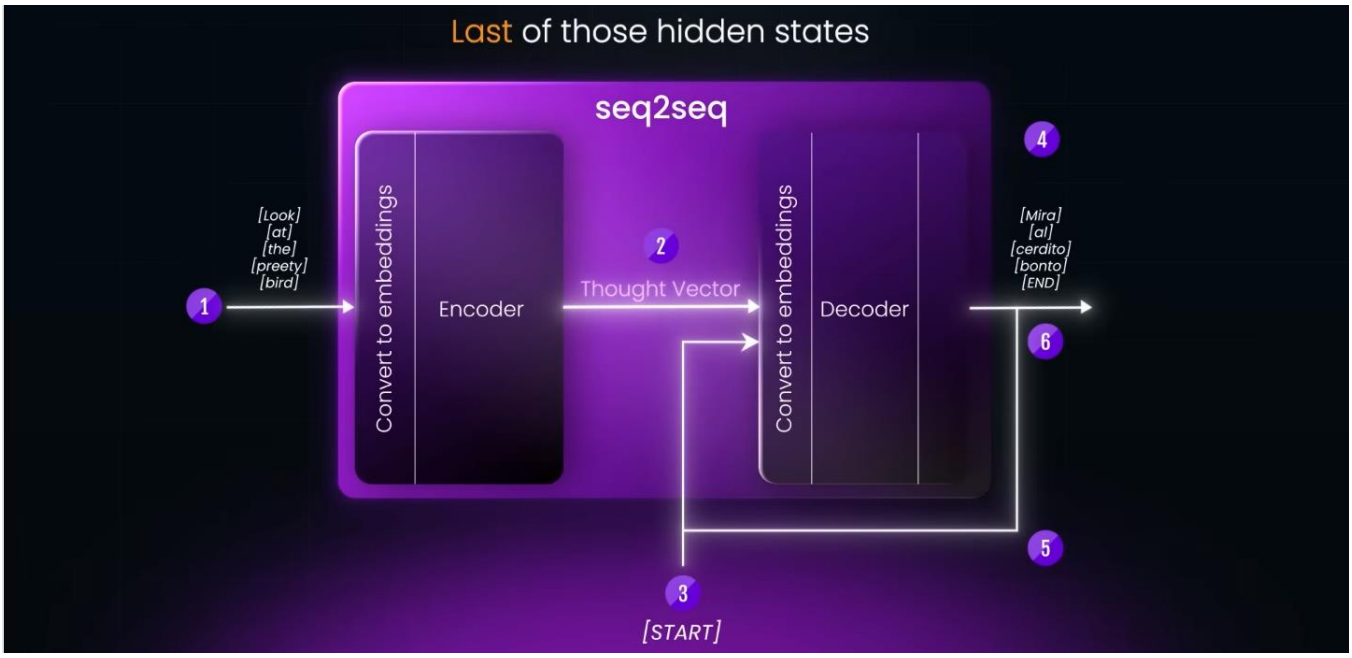
These models processed 1 token at a time and updated an internal memory after each step.

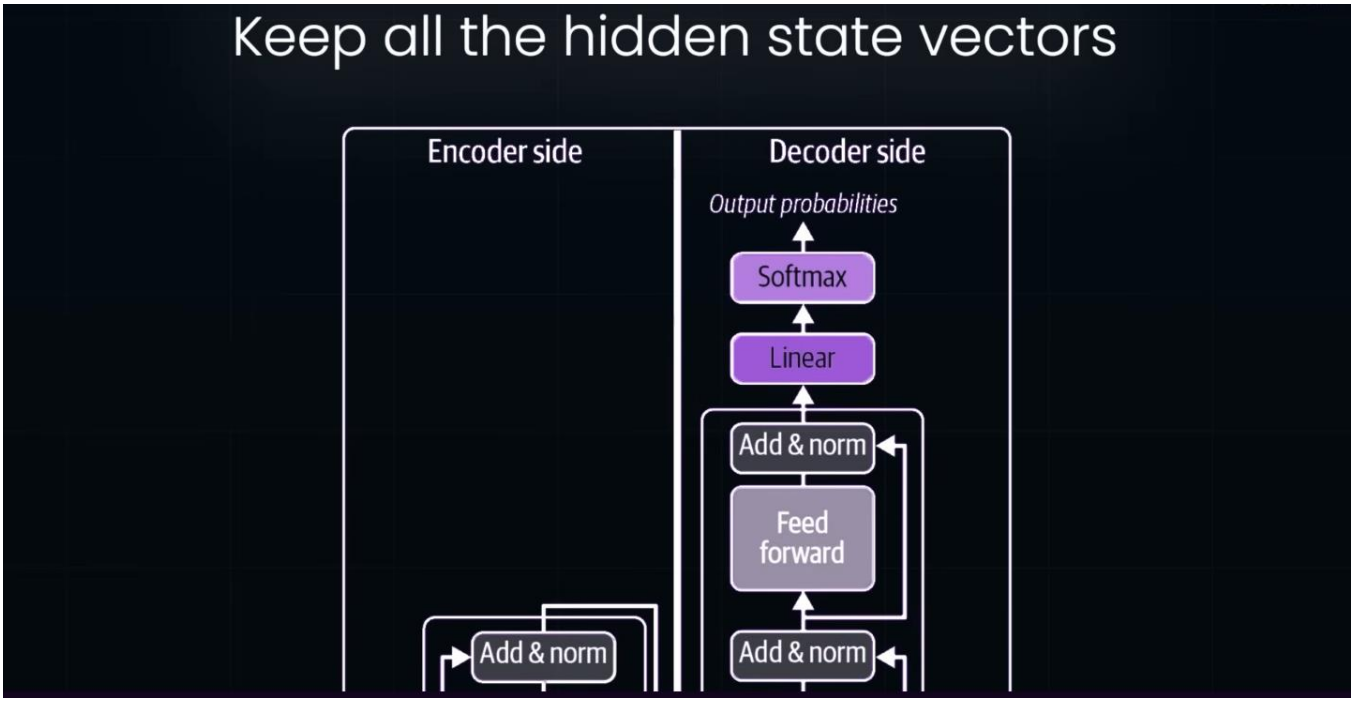

Sentences - Paragraphs - Documents
seq2seq
Convert to embeddings
Encoder
Thought Vector
Convert to embeddings
Decoder
[START]

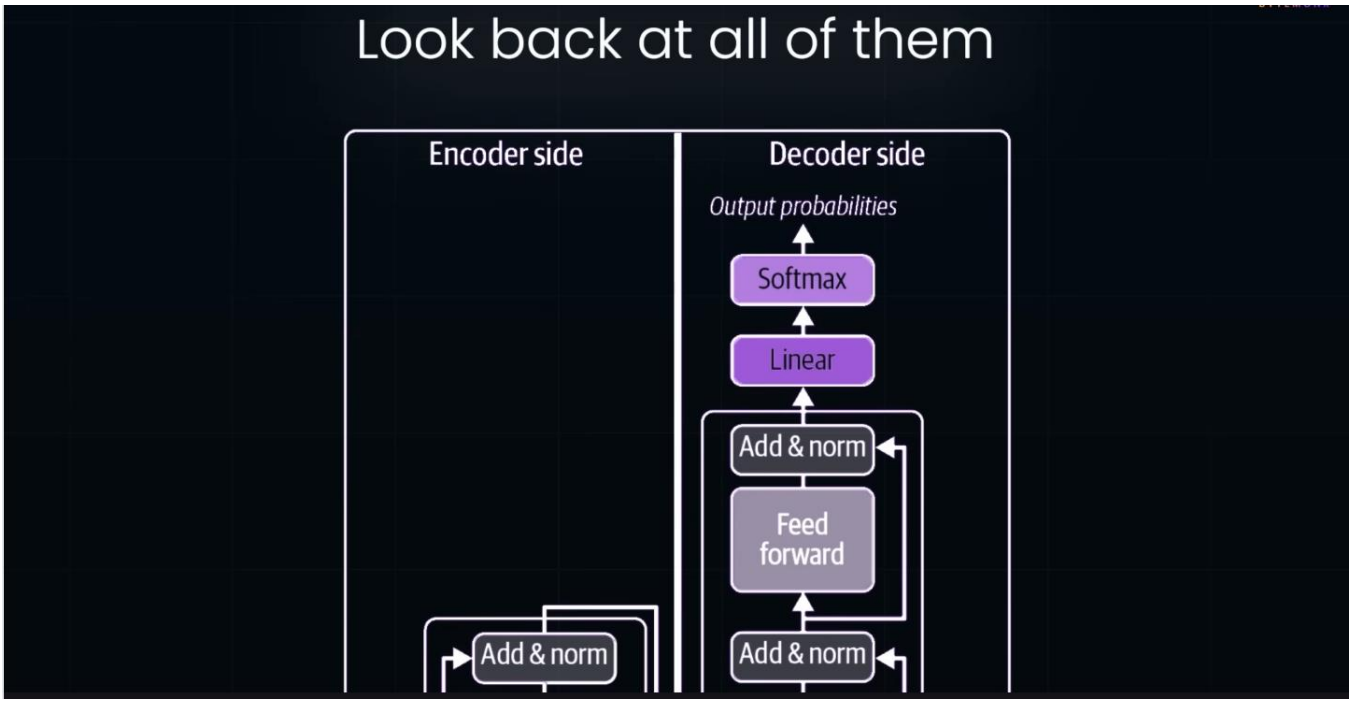

Information Bottleneck
seq2seq
Convert to embeddings
Encoder
Thought Vector
Convert to embeddings
Decoder
[START]

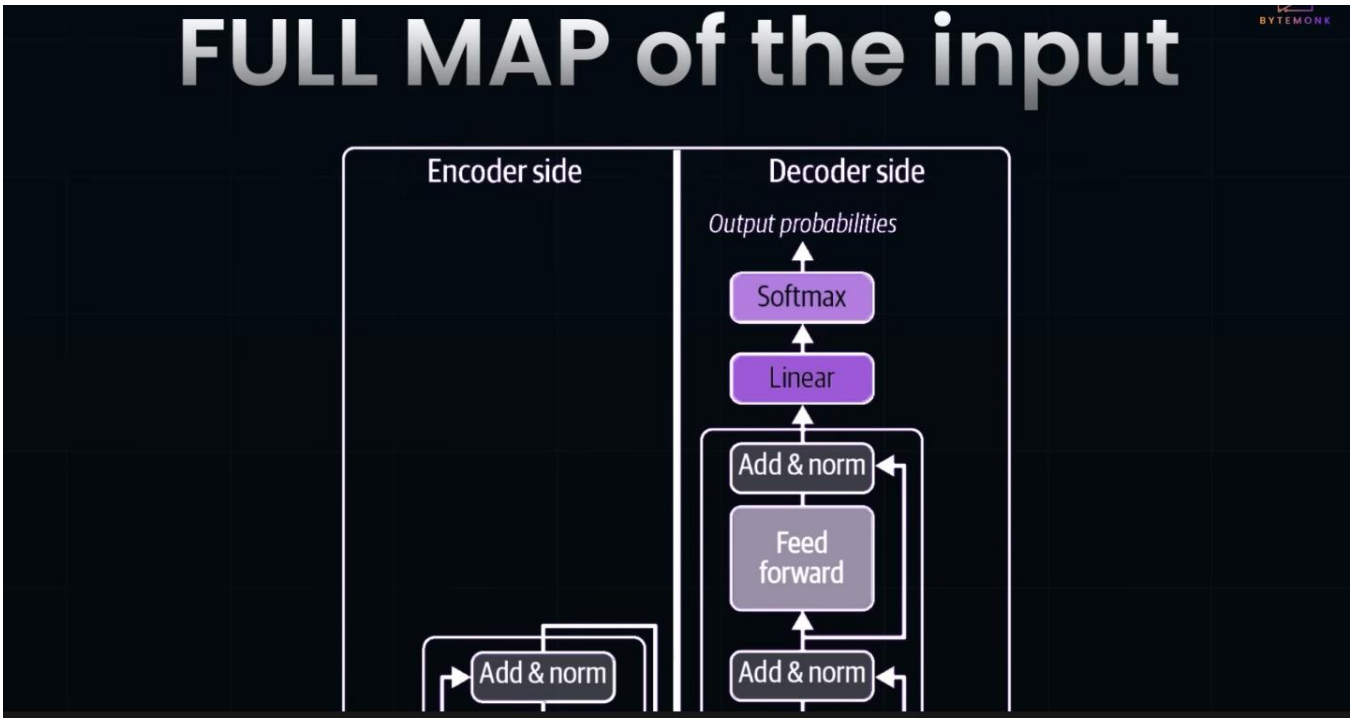

seq2seq
BYTEMONK
[Look]
[at]
[the]
[preety]
[bird]
1
Convert to embeddings
Encoder
2
Thought Vector
Convert to embeddings
Decoder
4
[Mira]
[ai]
[cerdito]
[bonto]
[END]
5
3
[START]

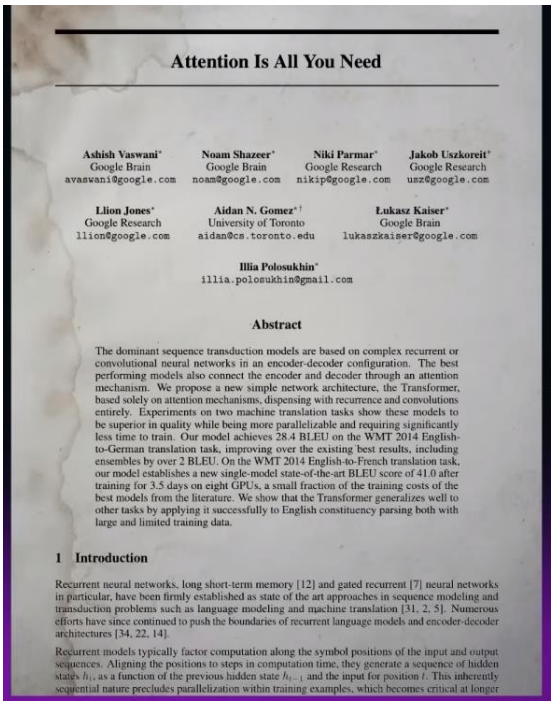

Attention Mechanism
BYTEMONK
seq2seq
[Look]
[at]
[the]
[preety]
[bird]
1
Convert to embeddings
Encoder
2
Thought Vector
Convert to embeddings
Decoder
4
[Mira]
[ai]
[cerdito]
[bonto]
[END]
6
5
3
[START]


Last of those hidden states
seq2seq
[Look]
[at]
[the]
[preety]
[bird]
1
Convert to embeddings
Encoder
2
Thought Vector
Convert to embeddings
Decoder
4
[Mira]
[al]
[cerdito]
[bonto]
[END]
6
5
3
[START]


Keep all the hidden state vectors
Encoder side
Decoder side
Output probabilities
Softmax
Linear
Add & norm
Feed forward
Add & norm
Add & norm


Look back at all of them
Encoder side
Decoder side
Output probabilities
Softmax
Linear
Add & norm
Feed forward
Add & norm
Add & norm


FULL MAP of the input
BYTEMONK
Encoder side
Decoder side
Output probabilities
Softmax
Linear
Add & norm
Feed forward
Add & norm
Add & norm



Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez*† University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin* illia.polosukhin@gmail.com

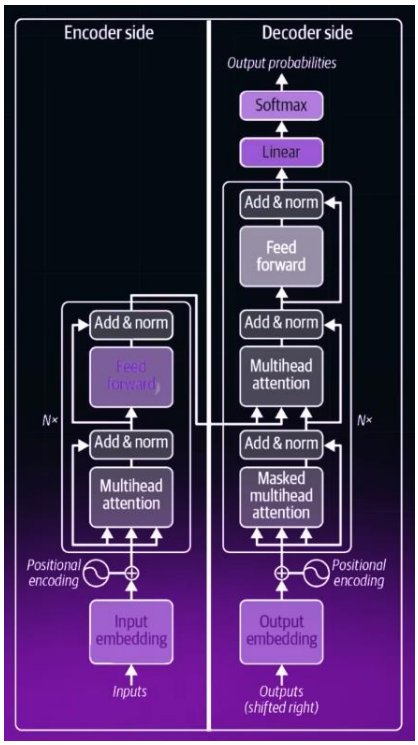
Abstract

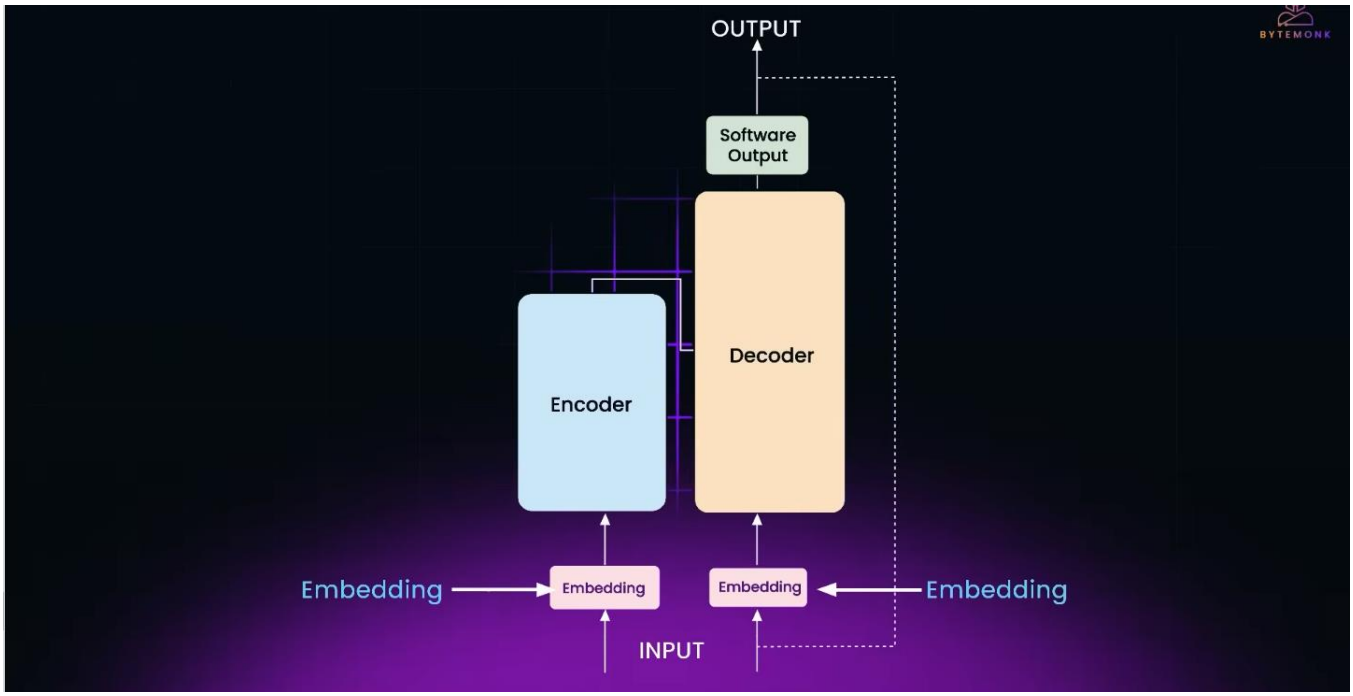
The dominant sequence transduction models are based on complex recurrent or convolutional neural networks in an encoder-decoder configuration. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.0 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

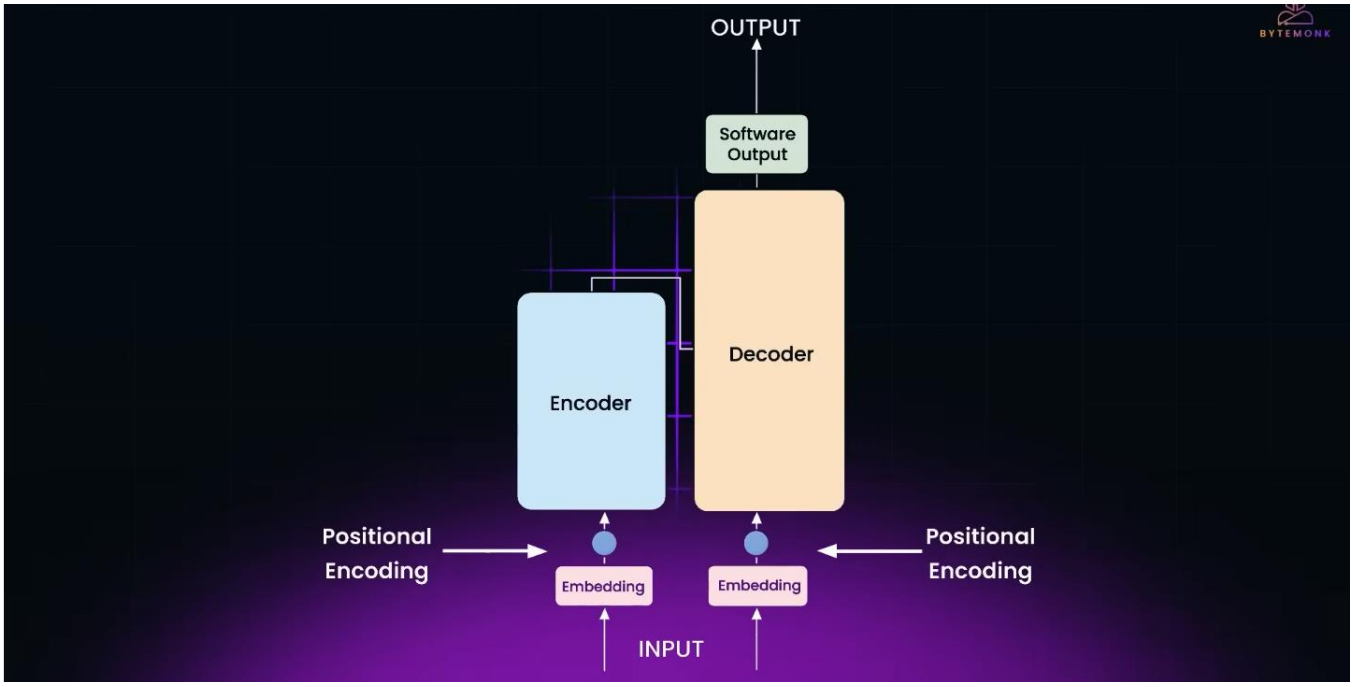
1 Introduction

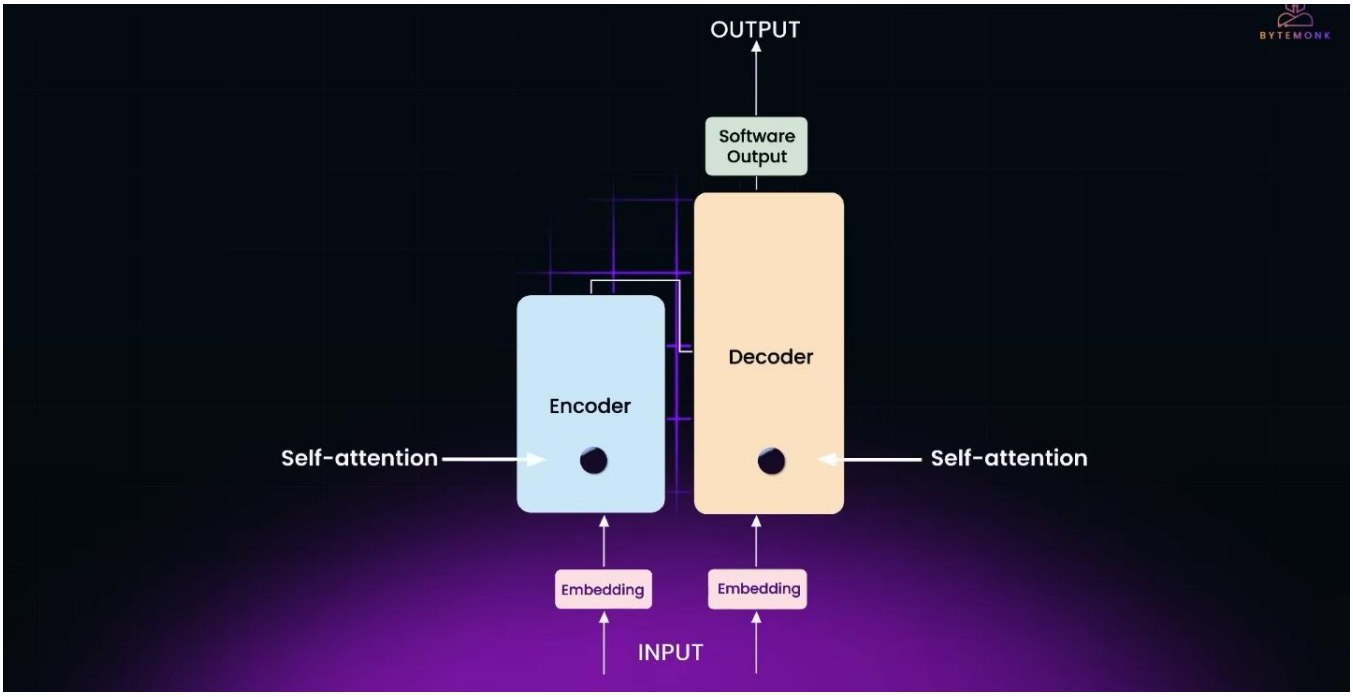
Recurrent neural networks, long short-term memory [12] and gated recurrent [7] neural networks in particular, have been firmly established as state of the art approaches in sequence modeling and transduction problems such as language modeling and machine translation [31, 2, 5]. Numerous efforts have since continued to push the boundaries of recurrent language models and encoder-decoder architectures [34, 22, 14].

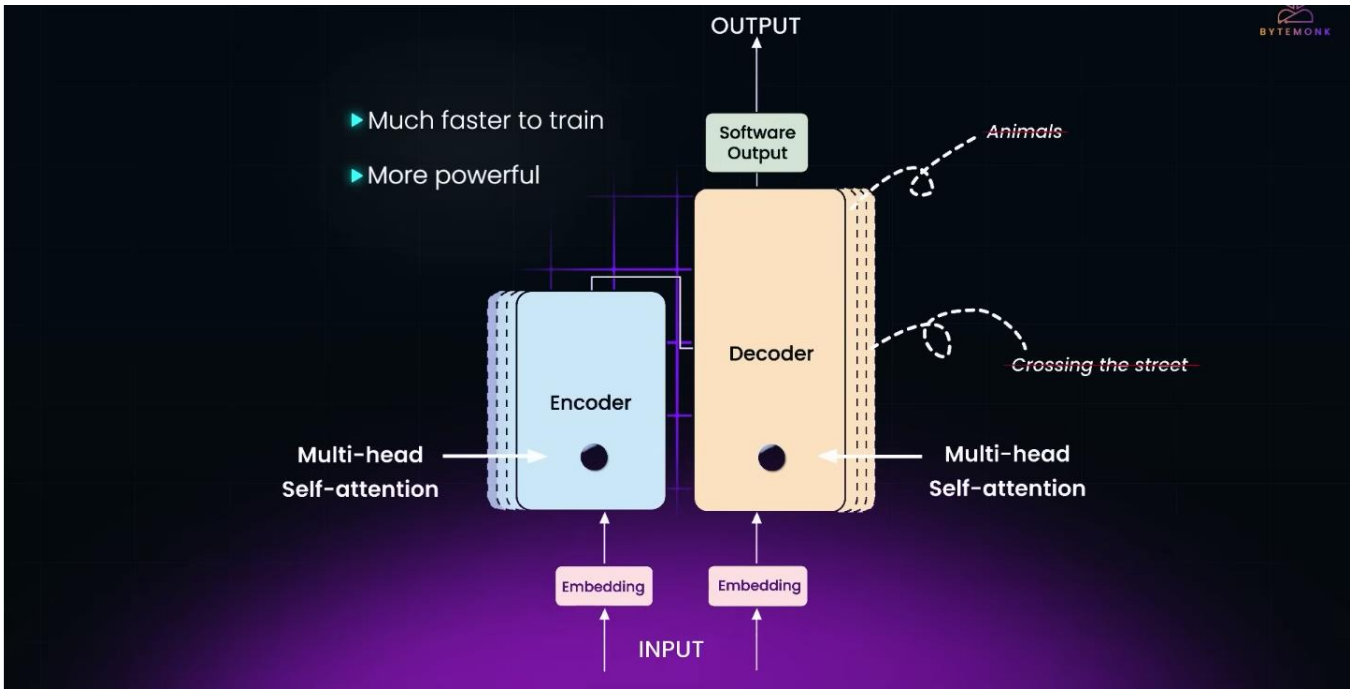
Recurrent models typically factor computation along the symbol positions of the input and output sequences. Aligning the positions to steps in computation time, they generate a sequence of hidden states h_t, as a function of the previous hidden state h_{t-1} and the input for position t. This inherently sequential nature precludes parallelization within training examples, which becomes critical at longer



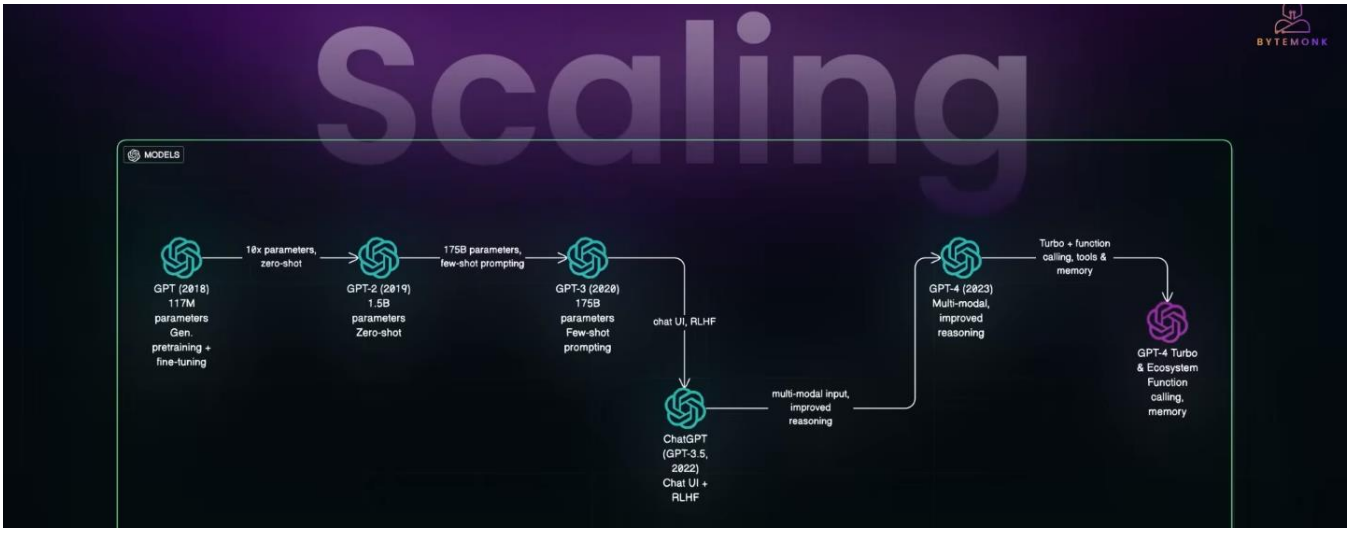


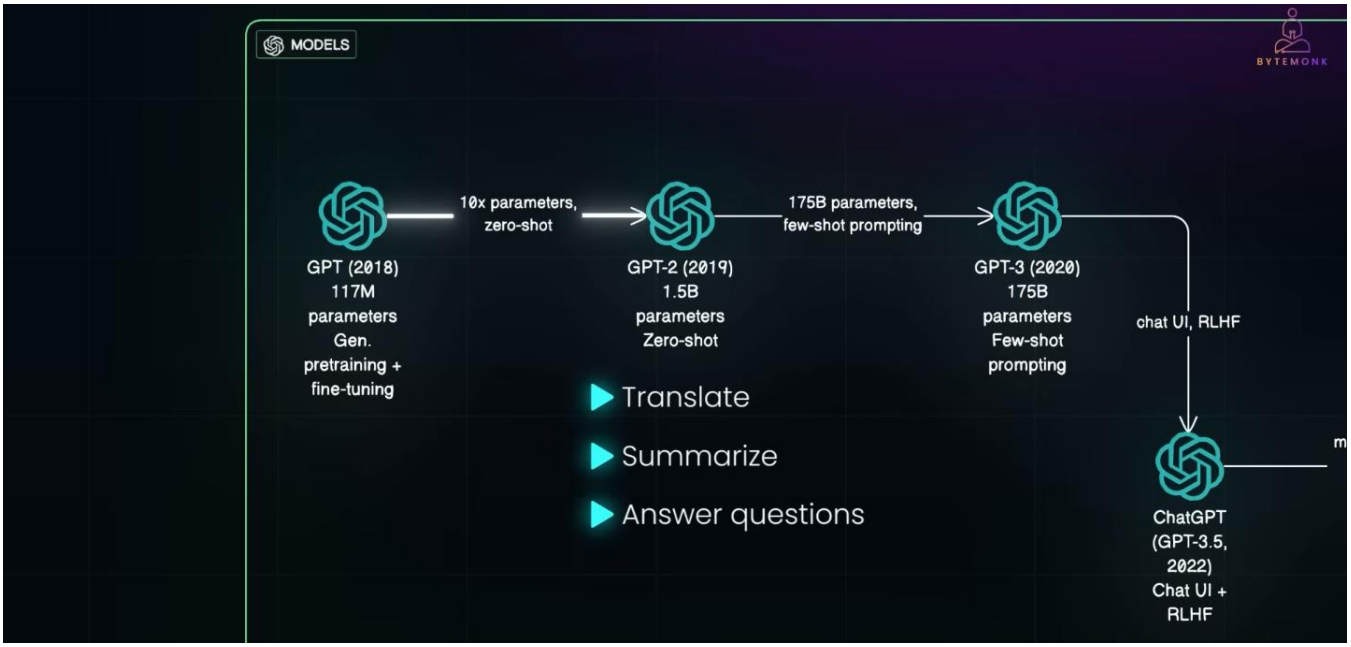

BYTEMONK
OUTPUT
Software Output
Decoder
Encoder
Positional Encoding
Positional Encoding
Embedding
Embedding
INPUT


BYTEMONK
OUTPUT
Software Output
Decoder
Encoder
Self-attention
Self-attention
Embedding
Embedding
INPUT

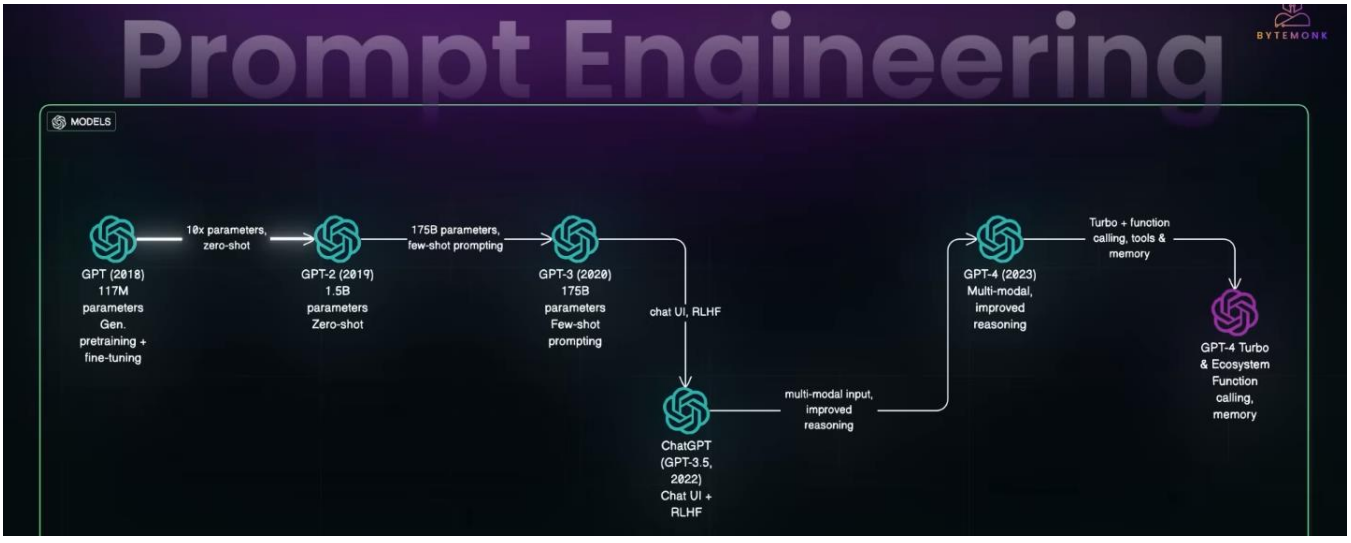

BYTEMONK
OUTPUT
Much faster to train
More powerful
Software Output
Animals
Decoder
Crossing the street
Encoder
Multi-head Self-attention
Multi-head Self-attention
Embedding
Embedding
INPUT

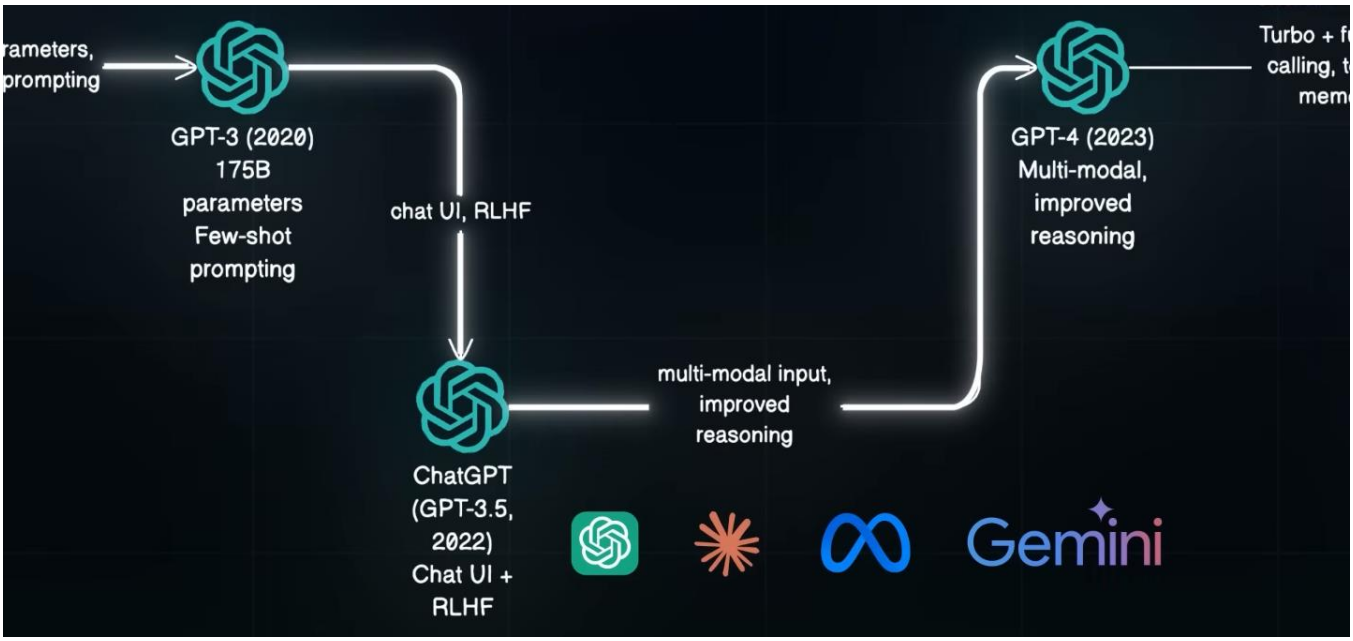

Tradeoff
Transformers can't handle unlimited input
Fixed context window


BYTEMONK
Scaling
MODELS
GPT (2018) 117M parameters Gen. pretraining + fine-tuning
10x parameters, zero-shot
GPT-2 (2019) 1.5B parameters Zero-shot
175B parameters, few-shot prompting
GPT-3 (2020) 175B parameters Few-shot prompting
chat UI, RLHF
ChatGPT (GPT-3.5, 2022) Chat UI + RLHF
multi-modal input, improved reasoning
GPT-4 (2023) Multi-modal, improved reasoning
Turbo + function calling, tools & memory
GPT-4 Turbo & Ecosystem Function calling, memory

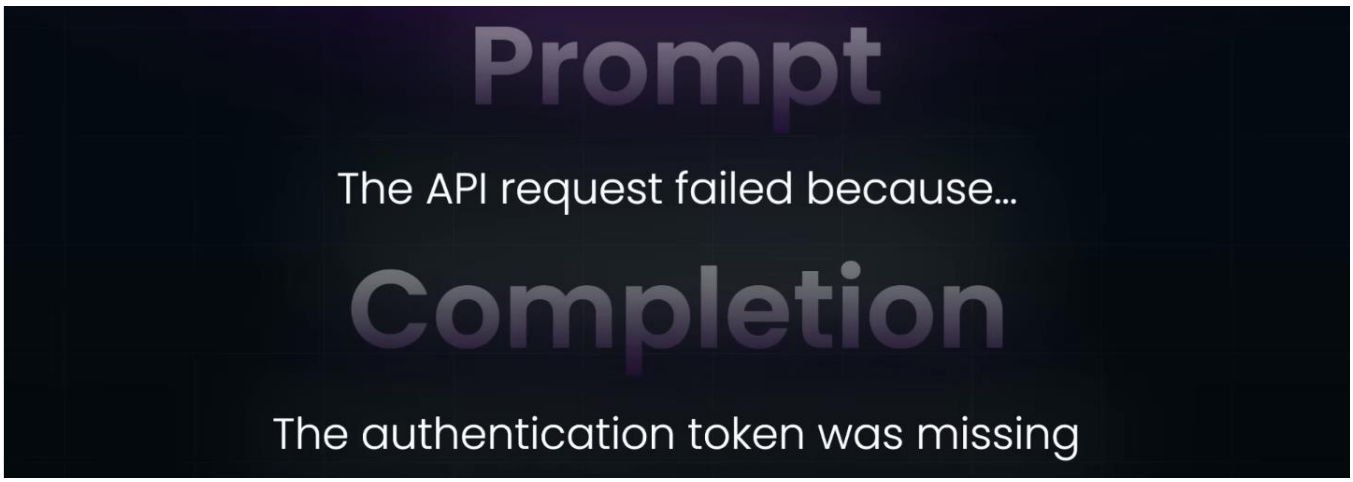

MODELS
BYTEMONK
10x parameters, zero-shot
175B parameters, few-shot prompting
GPT (2018) 117M parameters Gen. pretraining + fine-tuning
GPT-2 (2019) 1.5B parameters Zero-shot
GPT-3 (2020) 175B parameters Few-shot prompting
chat UI, RLHF
Translate
Summarize
Answer questions
ChatGPT (GPT-3.5, 2022) Chat UI + RLHF


10x parameters, zero-shot
175B parameters, few-shot prompting
GPT (2018) 117M parameters Gen. pretraining + fine-tuning
GPT-2 (2019) 1.5B parameters Zero-shot
GPT-3 (2020) 175B parameters Few-shot prompting
Don't need to fine-tune for every task.
You just need to prompt it right.


Prompt Engineering
BYTEMONK
MODELS
10x parameters, zero-shot
175B parameters, few-shot prompting
GPT (2018) 117M parameters Gen. pretraining + fine-tuning
GPT-2 (2019) 1.5B parameters Zero-shot
GPT-3 (2020) 175B parameters Few-shot prompting
chat UI, RLHF
ChatGPT (GPT-3.5, 2022) Chat UI + RLHF
multi-modal input, improved reasoning
GPT-4 (2023) Multi-modal, improved reasoning
Turbo + function calling, tools & memory
GPT-4 Turbo & Ecosystem Function calling, memory


rameters,
prompting
GPT-3 (2020)
175B
parameters
Few-shot
prompting
chat UI, RLHF
ChatGPT
(GPT-3.5,
2022)
Chat UI +
RLHF
multi-modal input,
improved
reasoning
GPT-4 (2023)
Multi-modal,
improved
reasoning
Turbo + f
calling, t
mem
Gemini


String of text
Returns a continuation
Prompt
Completion


Prompt
The API request failed because...
Completion
The authentication token was missing

The deployment failed during staging.
Now the app won't start on...

g9dl93

Monday

Production

The deployment failed during staging.
Now the app won't start on production...

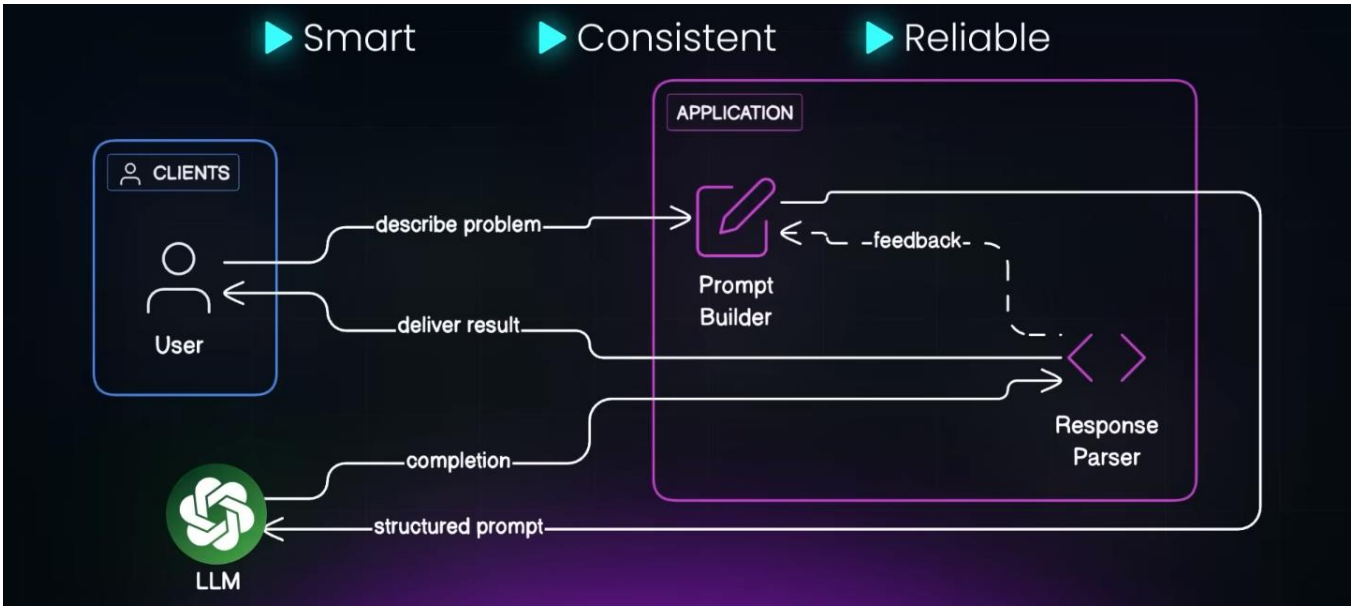
Let's refactor the pipeline before EOD.

Can you check the logs on your end?

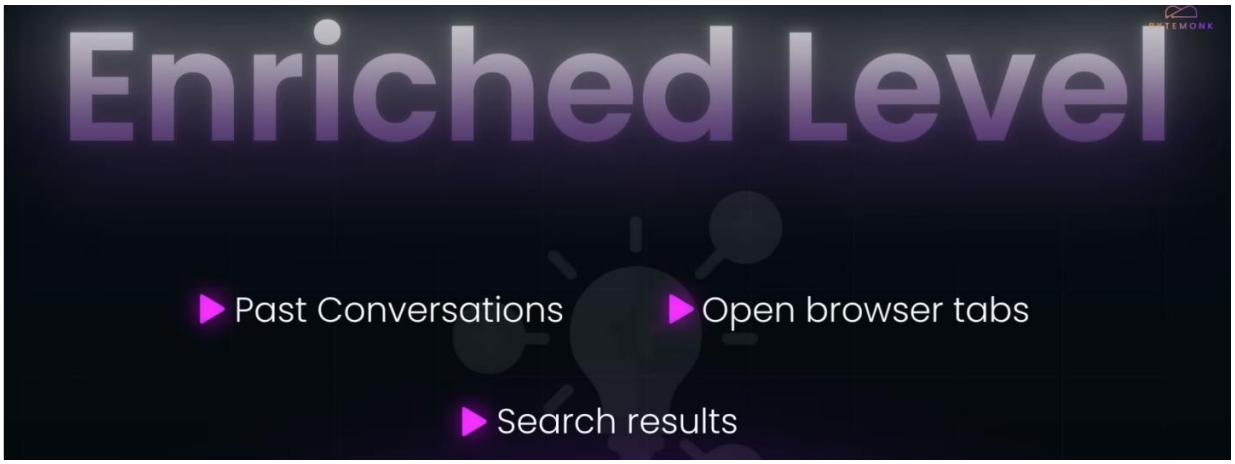
Try rolling back to the last known stable build.

Data Patterns

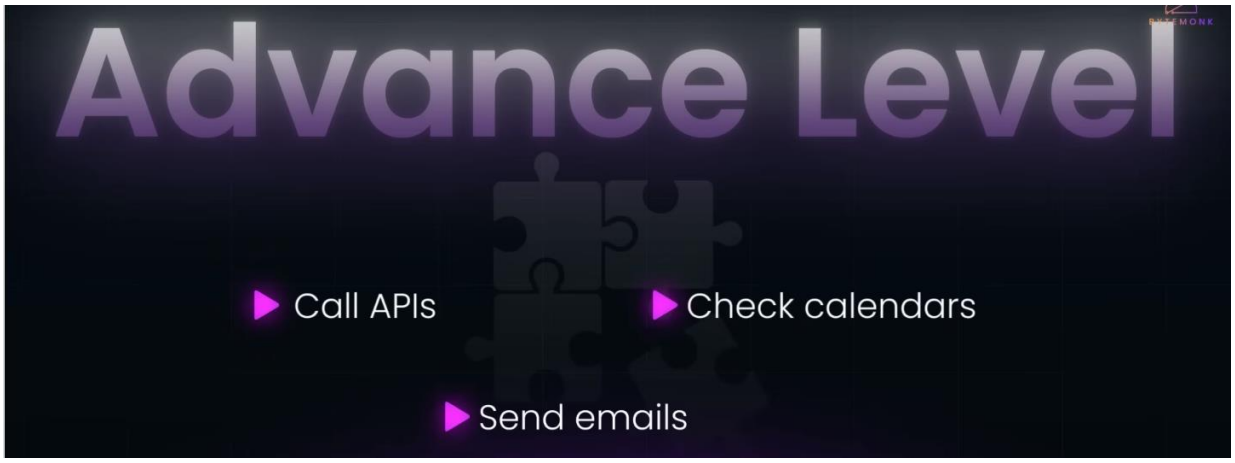
Not logic or facts




Basic level


Enriched Level
Past Conversations
Open browser tabs
Search results

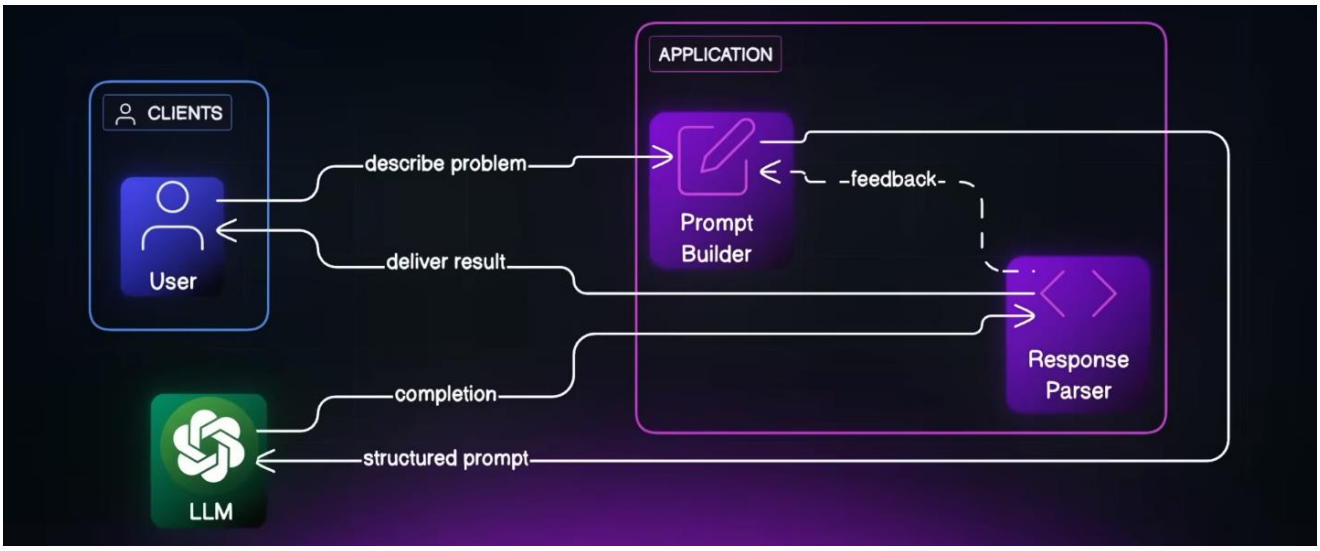

Enriched Level
Contextual
Memory Aware

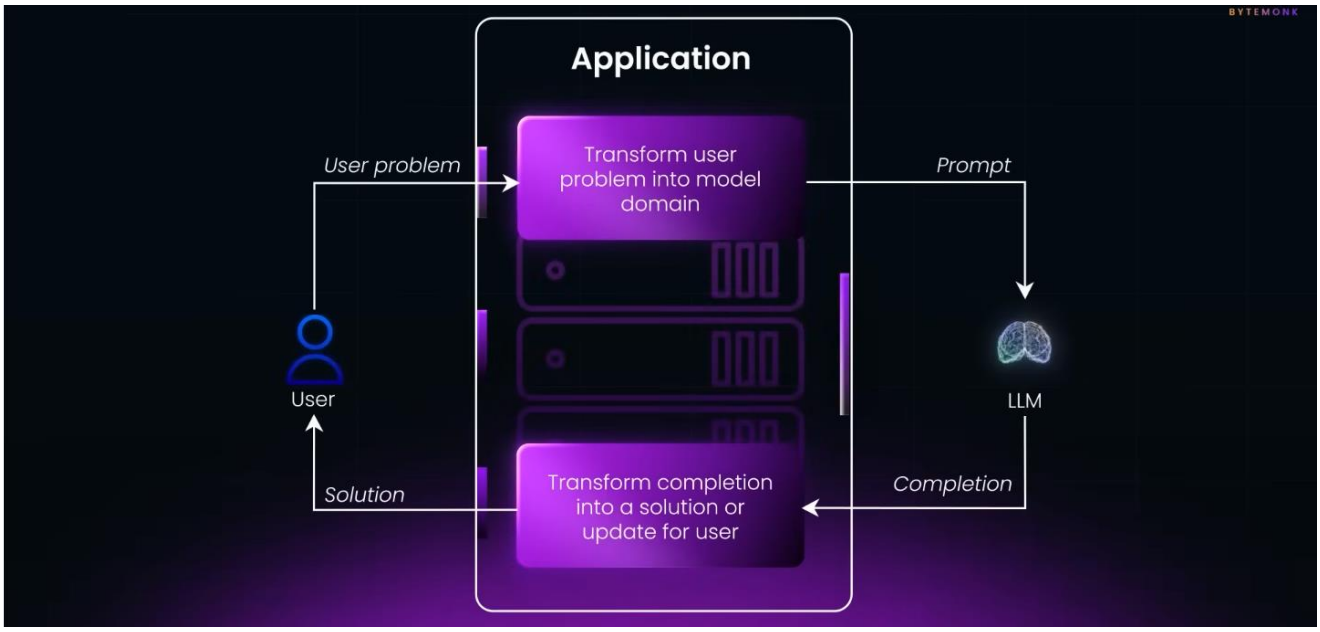

Advance Level
Call APIs
Check calendars
Send emails

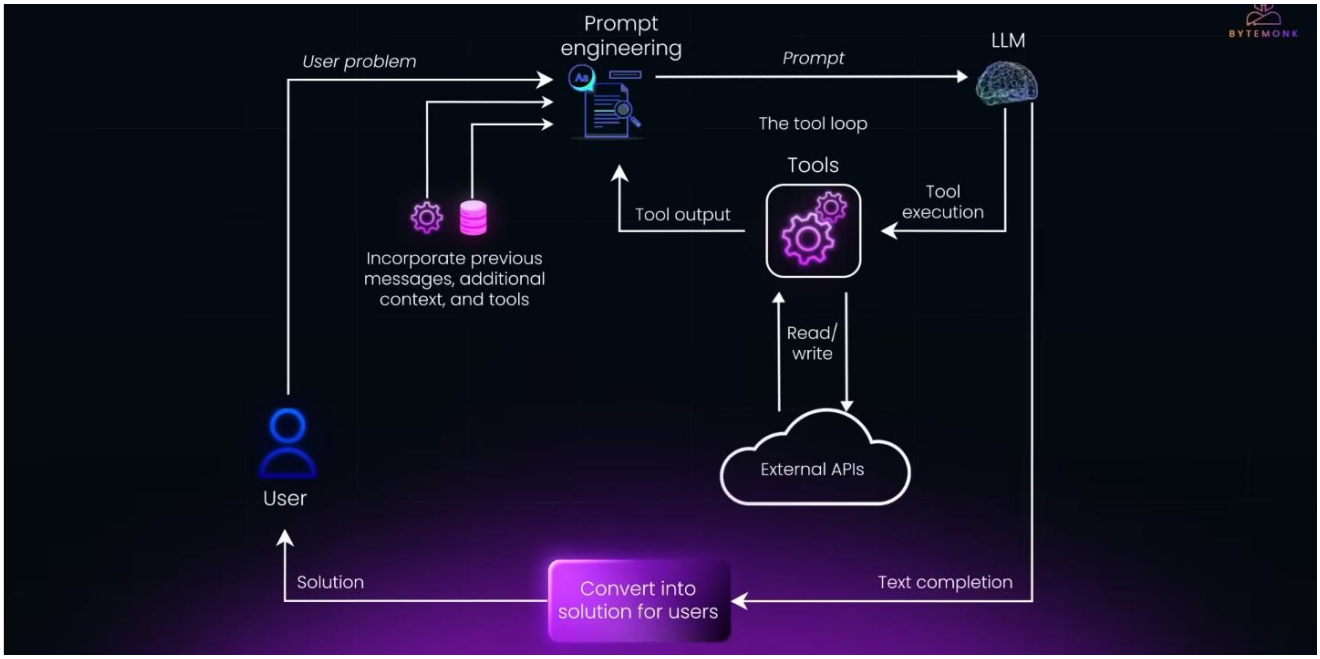
Advance Level

- Manage tool use
- Handle errors
- Pass state
- Keep the loop safe

Prompt Engineering




BYTEMONK
Application
User problem
Transform user problem into model domain
Prompt
User
LLM
Solution
Transform completion into a solution or update for user
Completion


BYTEMONK
Prompt engineering
LLM
User problem
Prompt
The tool loop
Tools
Tool output
Tool execution
Incorporate previous messages, additional context, and tools
Read/ write
External APIs
User
Solution
Convert into solution for users
Text completion