Hands-On LLM Engineering

Gain LLM Expertise: Level up your skills to build and deploy AI solutions with RAG, QLoRA and Agents

Ed Donner

Co-Founder and CTO, Nebula.io

Pearson


Prepare to get HANDS-ON!

THANK YOU!!

For spending 4 hours on Friday with me.

I'll make it worth every minute!

THANK YOU!!

For spending 4 hours on Friday with me.

I'll make it worth every minute!

The next 4 hours will be highly educational

THANK YOU!!

For spending 4 hours on Friday with me.

I'll make it worth every minute!

The next 4 hours will be highly educational

…and a whole lot of fun, too

Agenda

Segment 1:
Road to LLM Engineering

Agenda

Segment 1:
Road to LLM Engineering

Segment 2:
Models, Tools, Techniques

Agenda

Segment 1:
Road to LLM Engineering

Segment 2:
Models, Tools, Techniques

Segment 3:
Hands-on!

Agenda

Segment 1:
Road to LLM Engineering

Segment 2:
Models, Tools, Techniques

Segment 3:
Hands-on!

*This will **level up** your ability to select, apply and deploy LLMs to solve **real-world commercial problems**.*

Pearson

Setting you up to get the most out of the training

- Bring up the code in Jupyter Lab from the course repo
- After the Live Event, run the code yourself, confirm the results and reach out with questions
- Ask me questions relentlessly! That's what I'm here for.

And…

Setting you up to get the most out of the training

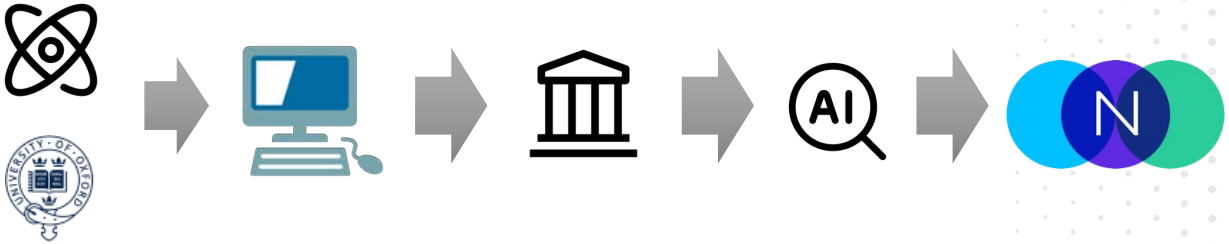
- Bring up the code in Jupyter Lab from the course repo
- After the Live Event, run the code yourself, confirm the results and reach out with questions
- Ask me questions relentlessly! That's what I'm here for.

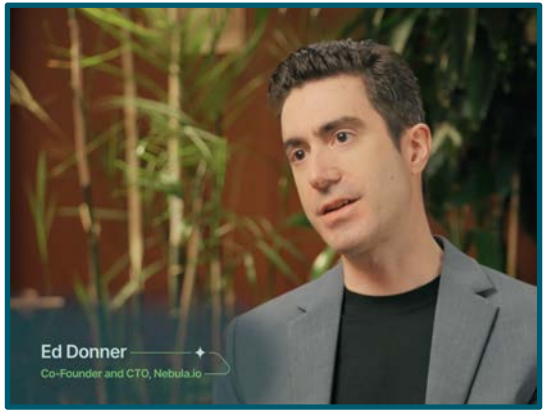
And…

I LOVE EMOJIS!!! ❤️👍

Introducing me..

Co-Founder and CTO of Nebula.io; 20-year career as Technology and Data Science leader, entrepreneur and coach





Pearson

Connect with me: https://www.linkedin.com/in/eddonner/

POLL: Introducing YOU

What best describes your experience with LLMs (select one):

A

I've used ChatGPT, done prompt engineering

B

I've made API calls to LLMs

C

I've used techniques like RAG or QLoRA fine-tuning

D

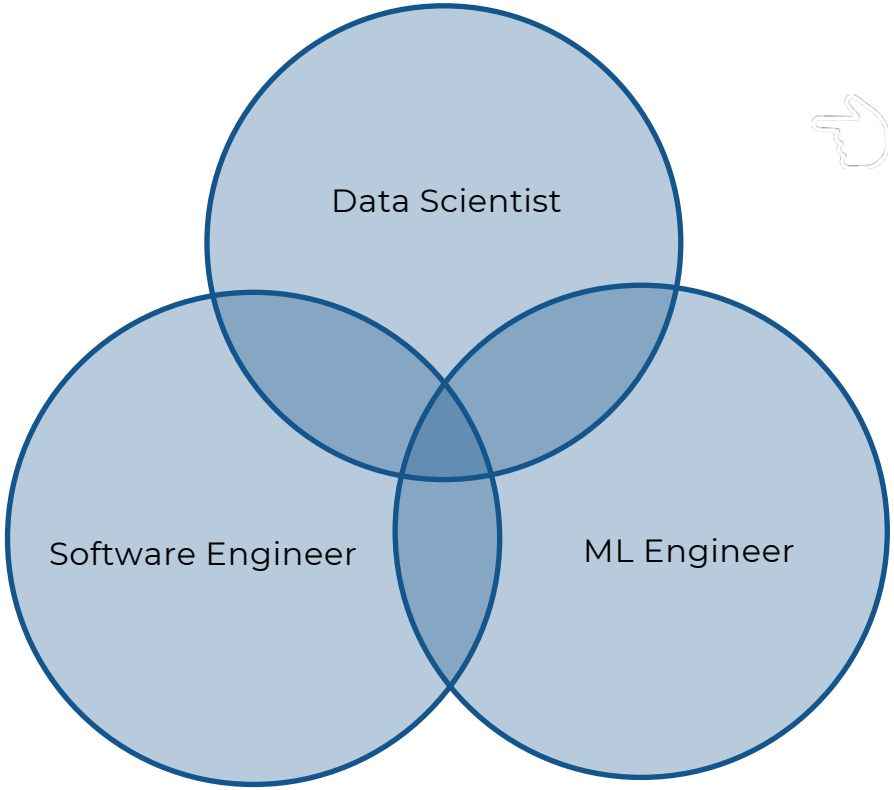
I've created a full Agentic AI Architecture from the ground up

The Cherry On Top

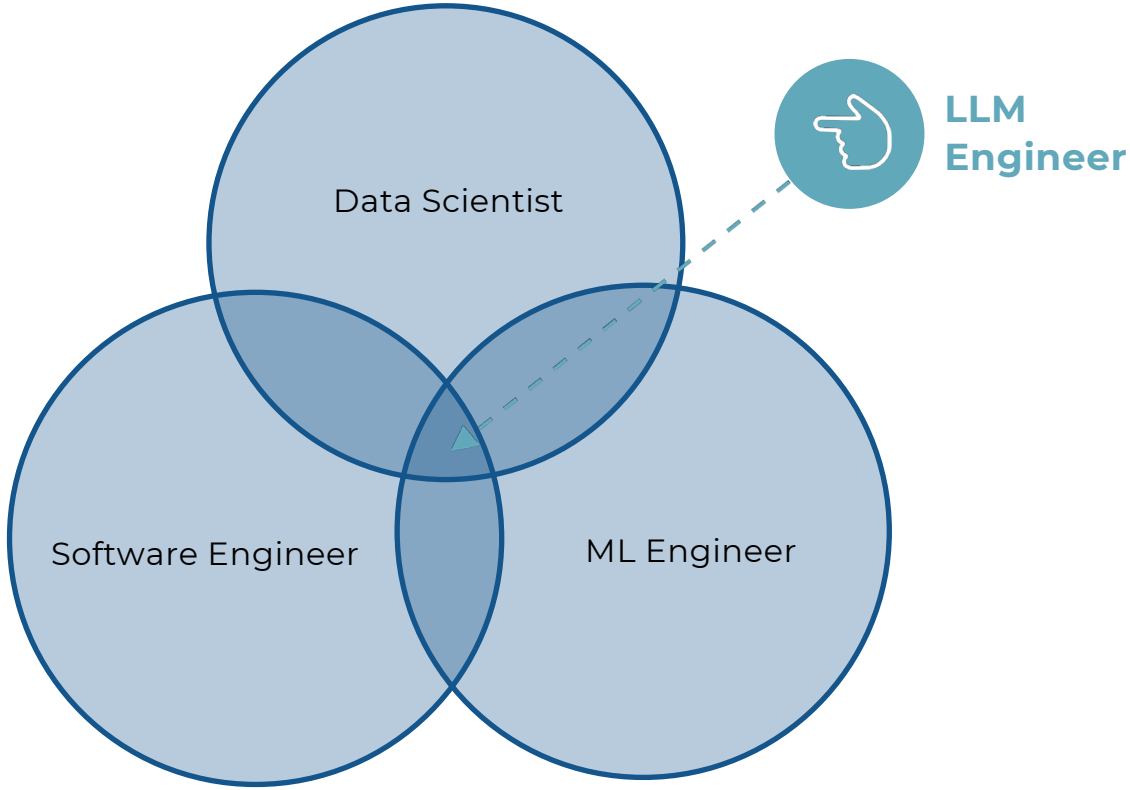
- Caters for all levels of experience
  - Too hard? Build intuition
  - Too easy? Hang in there…
  - Attended my prior courses? There's more!
- Culminating in a MASSIVE project
  1. Autonomous Agentic AI
  2. Proprietary frontier LLM
  3. Powerful RAG pipeline
- And a final treat at the end
- **NOW – let's get started!**

Segment 1: Path to LLM Engineering

LLM Engineer is an evolving hybrid role with Gen AI focus



LLM Engineer is an evolving hybrid role with Gen AI focus



An LLM Engineer combines 5 areas of expertise

Platforms
Architectures
Deployment

Techniques

Tools and
Frameworks

Models and
APIs

Theory

An LLM Engineer combines 5 areas of expertise

Platforms Architectures Deployment	Jupyter Lab VSCode Google Colab Sagemaker	LangGraph	Autogen Crew AI Swarm	Advanced RAG workflows	Agentic design patterns	Production deployment architectures	Modal Runpod Lightning AI
Techniques	Prompting, Memory, Context, Multi-shot	Structured outputs Prompt caching Batch API	Tool Use / Function calling	RAG	Fine-tuning closed-source models	Fine-tuning open-source models	Agentic AI
Tools and Frameworks	Gradio Streamlit	Ollama llama.cpp	Numpy, scipy Scikit-learn Pandas Matplotlib	PyTorch	HuggingFace hub Pipelines Tokenizers Models	LangChain	Vector DBs: Chroma, Pinecone, Weaviate
Models and APIs	Chat UI vs. Cloud API vs. Direct inference	Artifacts, Canvas, Computer Use, Reasoning	Closed-source models	Amazon Bedrock Google Vertex AI Azure AI LiteLLM, Groq	Open-source models	Multi-modal and coding models	Benchmarks Leaderboards Arenas
Theory	LLM Foundations: tokens, prompts, costs, auto-regressive LLMs	Traditional Machine Learning	Deep Learning Transformers LLMs	Data Science R&D Data Curation	Vector embeddings and Encoders	LLM training, fine-tuning and QLoRA	Build a Transformer from scratch

An LLM Engineer combines 5 areas of expertise

Platforms Architectures Deployment	Jupyter Lab VSCode Google Colab Sagemaker	LangGraph	Autogen Crew AI Swarm	Advanced RAG workflows	Agentic design patterns	Production deployment architectures	Modal Runpod Lightning AI
Techniques	Prompting, Memory, Context, Multi-shot	Structured outputs Prompt caching Batch API	Tool Use / Function calling	RAG	Fine-tuning closed-source models	Fine-tuning open-source models	Agentic AI
Tools and Frameworks	Gradio Streamlit	Ollama llama.cpp	Numpy, scipy Scikit-learn Pandas Matplotlib	PyTorch	HuggingFace hub Pipelines Tokenizers Models	LangChain	Vector DBs: Chroma, Pinecone, Weaviate
Models and APIs	Chat UI vs. Cloud API vs. Direct inference	Artifacts, Canvas, Computer Use, Reasoning	Closed-source models	Amazon Bedrock Google Vertex AI Azure AI LiteLLM, Groq	Open-source models	Multi-modal and coding models	Benchmarks Leaderboards Arenas
Theory	LLM Foundations: tokens, prompts, costs, auto-regressive LLMs	Traditional Machine Learning	Deep Learning Transformers LLMs	Data Science R&D Data Curation	Vector embeddings and Encoders	LLM training, fine-tuning and QLoRA	Build a Transformer from scratch

Four Powerful Techniques for optimizing LLM performance

Four Powerful Techniques for optimizing LLM performance

Fine-Tuning

Four Powerful Techniques for optimizing LLM performance

Fine-Tuning

Multi-shot Prompting

Four Powerful Techniques for optimizing LLM performance

Fine-Tuning

Multi-shot Prompting

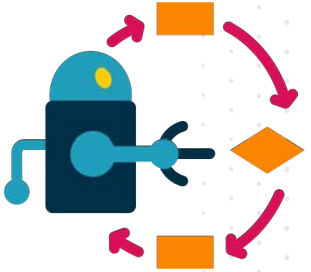
RAG

Four Powerful Techniques for optimizing LLM performance

Fine-Tuning

Multi-shot Prompting

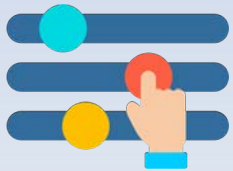
RAG



Agentic Workflows

Four Powerful Techniques for optimizing LLM performance

TRAINING TIME

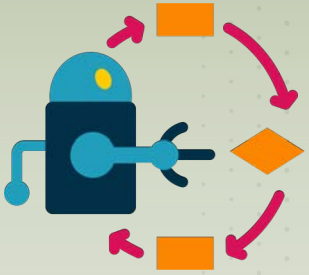


Fine-Tuning

INFERENCE TIME

Multi-shot Prompting

RAG

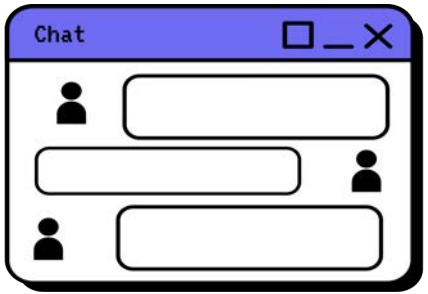


Agentic Workflows

In this first segment, we will experiment in 3 ways:

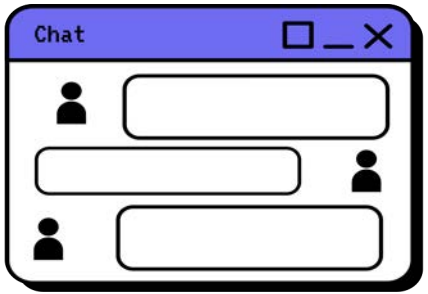
In this first segment, we will experiment in 3 ways:

Chat UI

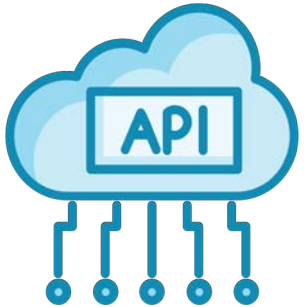


In this first segment, we will experiment in 3 ways:

Chat UI

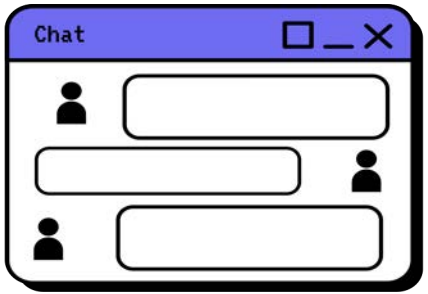


Cloud API

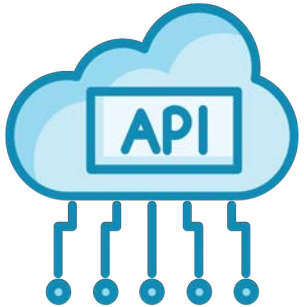


In this first segment, we will experiment in 3 ways:

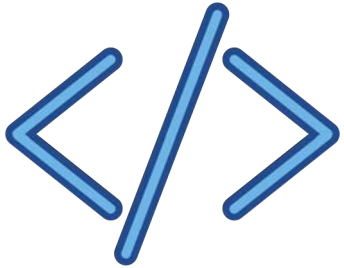
Chat UI



Cloud API



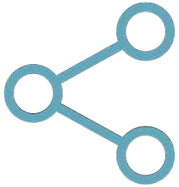
Directly executing
Open-Source Models



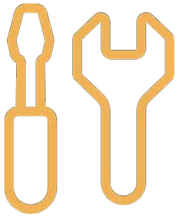

Time for ACTION!

Segment 2: Models, Tools, Techniques

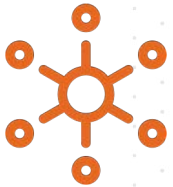
The Large Language Model (LLM)



Deep Neural Network trained to understand and generate language



Typically, Transformer Architecture

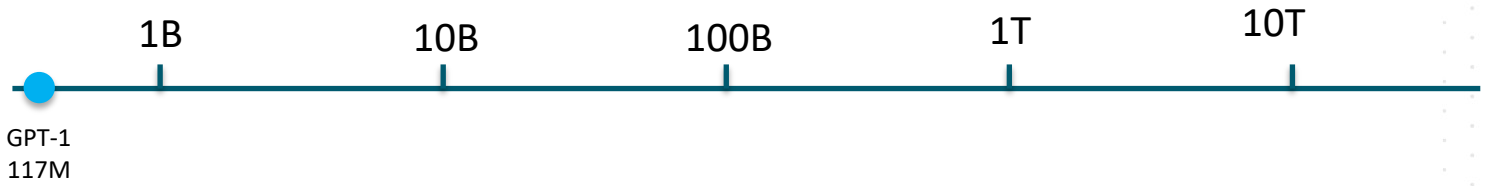


Takes “tokens” of several characters as inputs

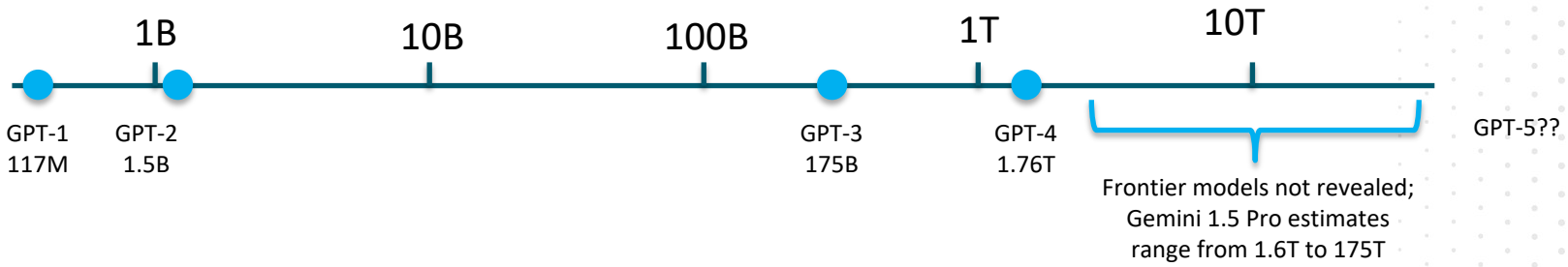
Generates text a token at a time

Can only look back across a limited number of tokens – the “context window”

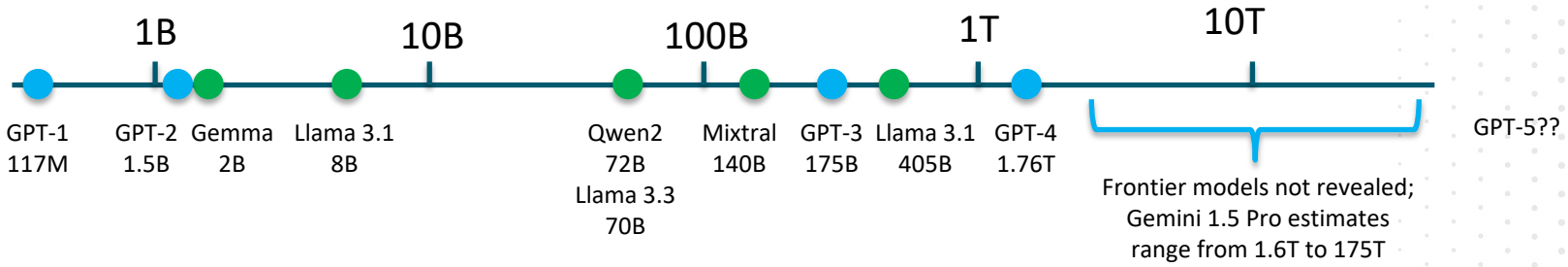
Number of parameters in LLMs (log scale)



Number of parameters in LLMs (log scale)



Number of parameters in LLMs (log scale)



Forward Pass vs Backprop

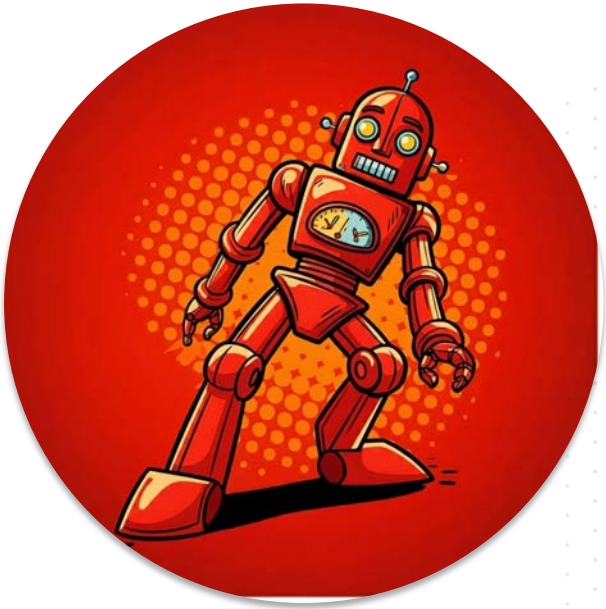
The Forward Pass

Given inputs, calculate the output(s).
Used during inference.

Forward Pass vs Backprop

The Forward Pass

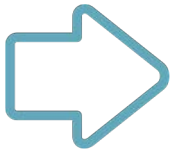
Given inputs, calculate the output(s).
Used during inference.



The Backward Pass

Use a technique called “Back Propagation” or “Backprop” to determine how the output is affected by a small change in weights.
Used during training (with the forward pass)

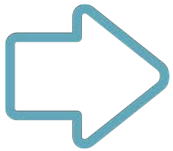
The Four Steps in Training



Forward pass

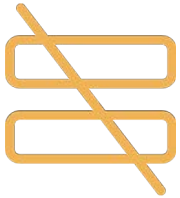
Predict the output given inputs

The Four Steps in Training



Forward pass

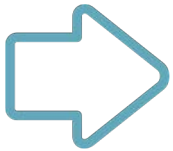
Predict the output given inputs



Loss calculation

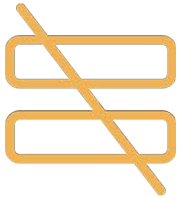
How different was the prediction to the ground truth

The Four Steps in Training



Forward pass

Predict the output given inputs



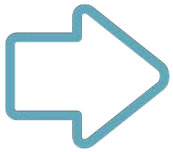
Loss calculation

How different was the prediction to the ground truth

Backward pass

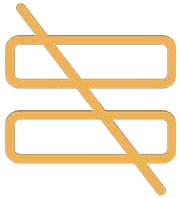
How should we tweak parameters to do better next time (the "gradients")

The Four Steps in Training



Forward pass

Predict the output given inputs



Loss calculation

How different was the prediction to the ground truth

Backward pass

How should we tweak parameters to do better next time (the "gradients")

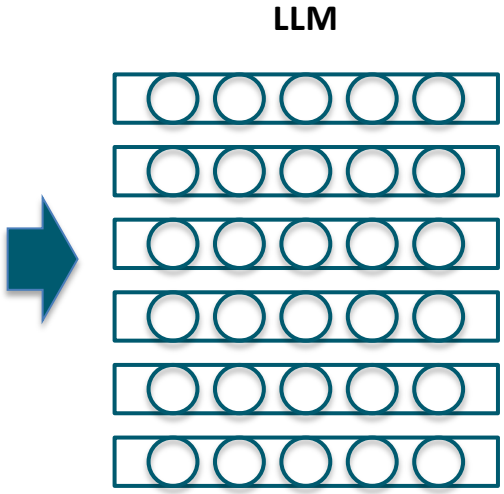
Optimization

Update parameters a tiny step to do better next time

1. Forward Pass

Input Prompt

We are living in a material <???>



Predicted Next Token

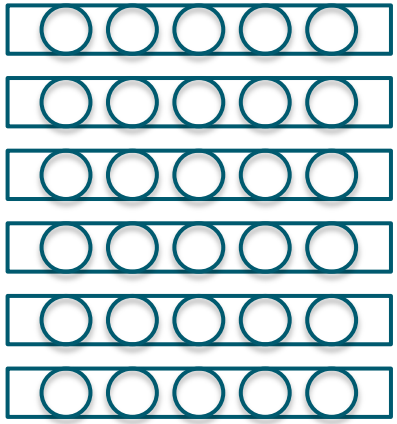
We are living in a material **place**

1. Forward Pass

Input Prompt

We are living in a material <???>

LLM



Predicted Next Token

We are living in a material **place**

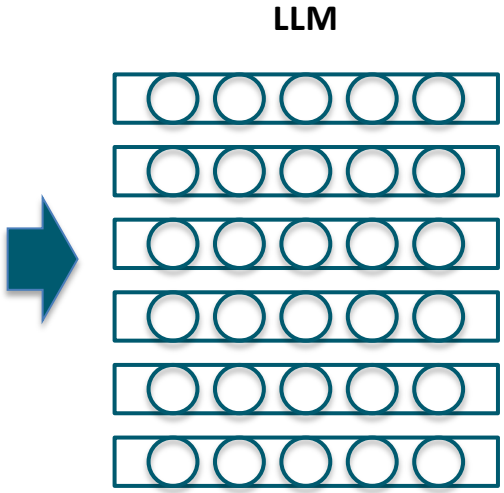
To be more precise:

- Outputs represent probabilities that the next token is *any given token*
- A common approach: pick the token with the highest probability and call it the 'predicted next token'
- In this example, it may have given **place** a probability of 40%, and **world** a probability of 10%

2. Loss Calculation

Input Prompt

We are living in a material <???>



Predicted Next Token

We are living in a material **place**

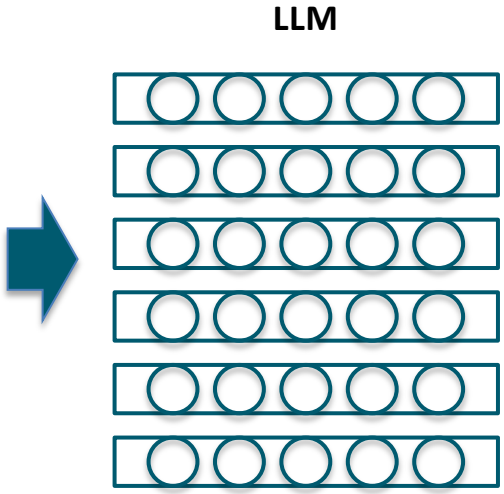
Actual Next Token

We are living in a material **world**

2. Loss Calculation

Input Prompt

We are living in a material <???>



Predicted Next Token

We are living in a material **place**

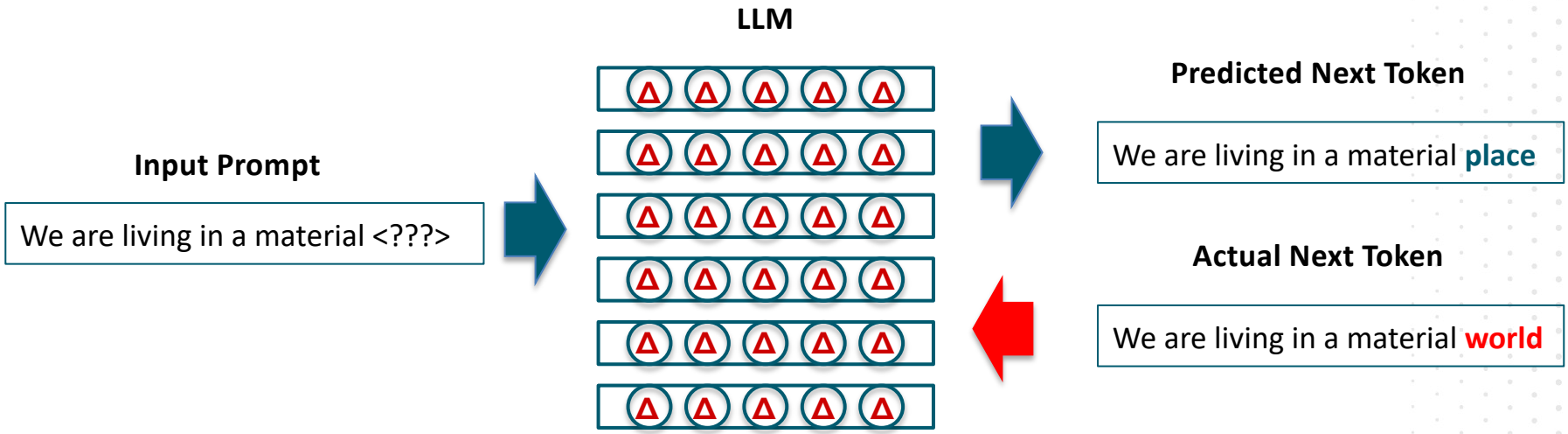
Actual Next Token

We are living in a material **world**

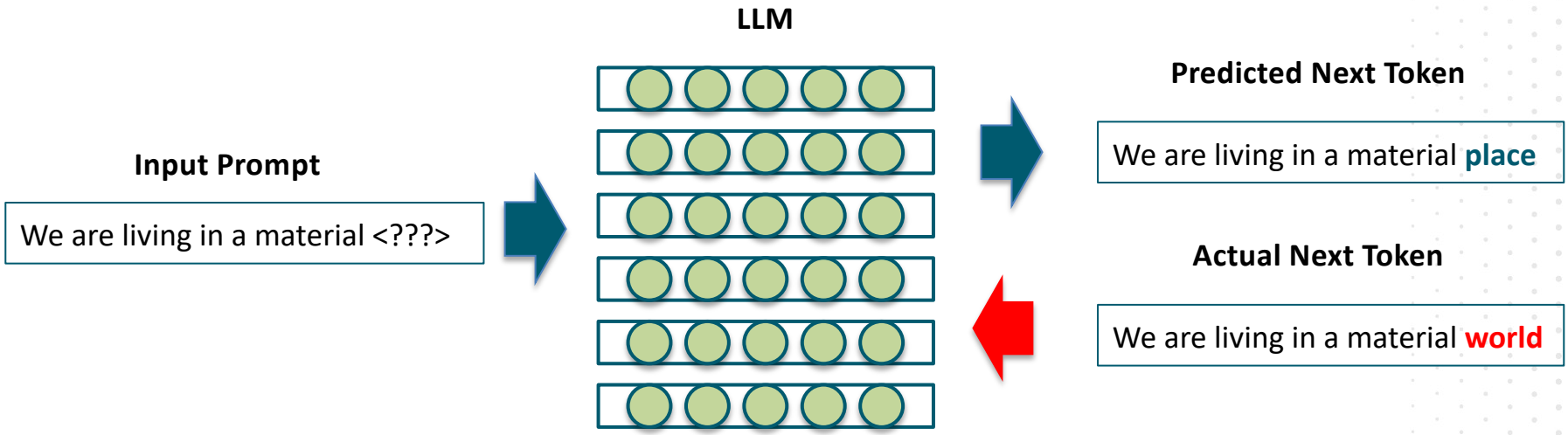
To be more specifc:

- Loss calculation uses the probability given to the actual next token **world**
- Formula is “cross entropy loss” or –log(probability_given_to_actual_token)
- So if **world** was given 10% probability, then loss is –log(0.1) = 2.3
- A perfect model would predict **world** with 100% probability so loss is -log(1) = 0

3. Backward Pass



4. Optimization



Which is the best LLM??

~~Which is the best LLM??~~

How to compare LLMs **for a given task**

How to compare LLMs **for a given task**

1

How to evaluate success?

Model metrics

Business outcomes

Cost and time to market

How to compare LLMs **for a given task**

1

How to evaluate success?
Model metrics
Business outcomes
Cost and time to market

2

Then start with the Basics
Parameters,
Context length,
Cost, etc

How to compare LLMs **for a given task**

1

How to evaluate success?
Model metrics
Business outcomes
Cost and time to market

2

Then start with the Basics
Parameters,
Context length,
Cost, etc

3

Then look at Results
Benchmarks,
Leaderboards,
Arenas

How to compare LLMs **for a given task**

1

How to evaluate success?
Model metrics
Business outcomes
Cost and time to market

2

Then start with the Basics
Parameters,
Context length,
Cost, etc

3

Then look at Results
Benchmarks,
Leaderboards,
Arenas

4

Then R&D
Select models
Prototype
Measure metrics

How to compare LLMs for a given task - BASICS

Start with the basics

- Open Source or Closed?
- Release date
- Parameters
- Training tokens
- Context length
- Knowledge cut-off
- Cost!
- License

Usually available on a Model Card

How to compare LLMs for a given task - RESULTS

Then investigate the results:

- Benchmarks
- Leaderboards
- Arenas

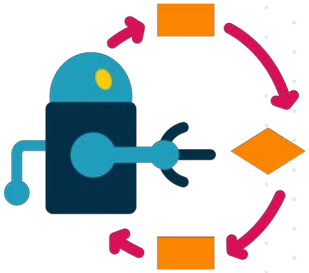
Live Event coming up on Feb 7th on Choosing Right LLM

- Looking at benchmarks and leaderboards
- How to interpret
- Their limitations

Deep Dive on 3 techniques to improve performance

RAG

Fine-Tuning

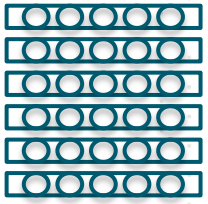


Agentic AI

The small idea behind RAG

Chat

Code



LLM

Knowledge Base

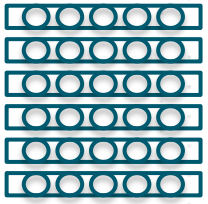
Pearson

The small idea behind RAG

Chat

Question

Code

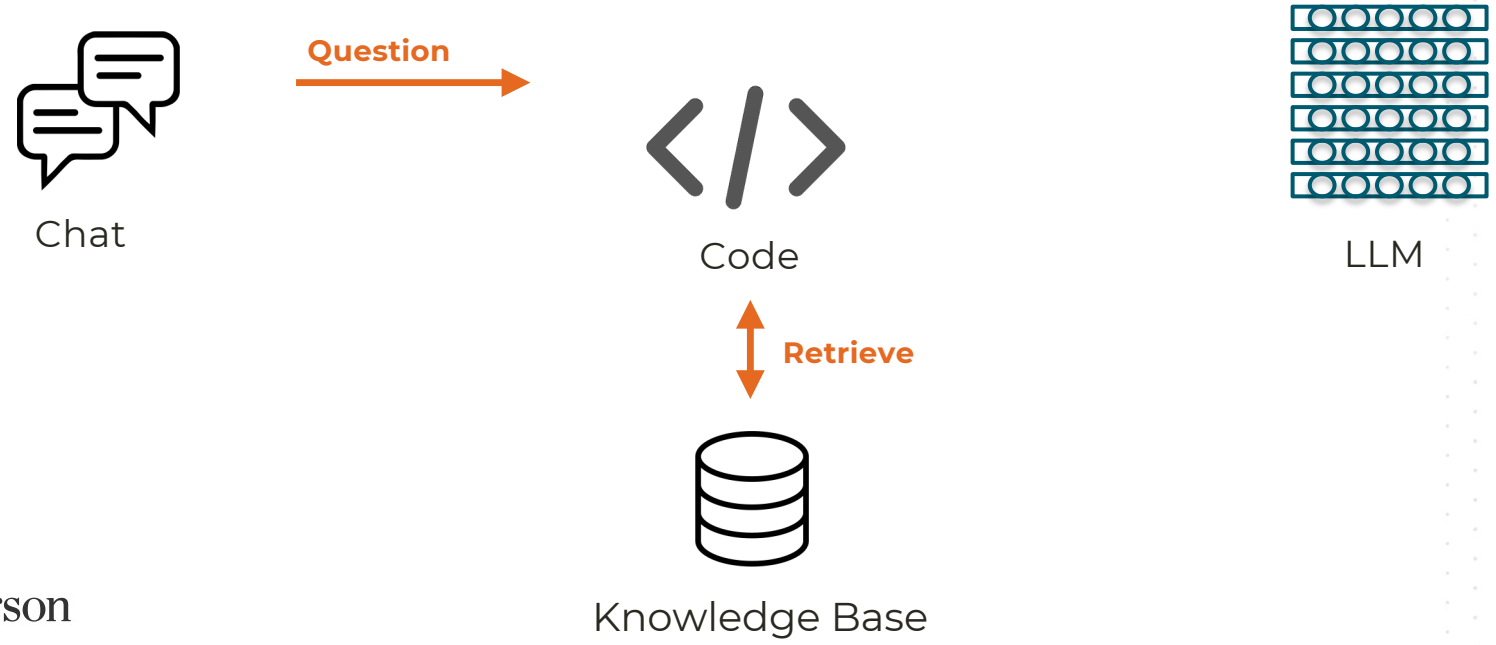


LLM

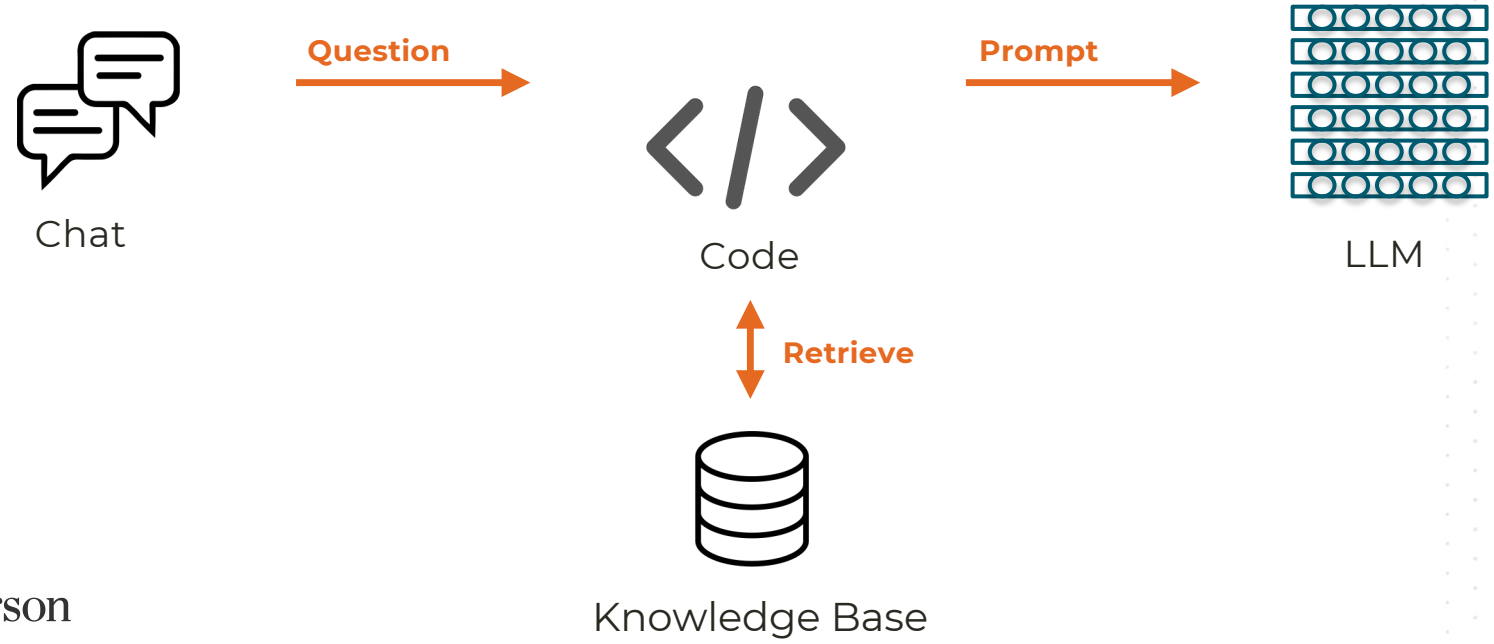
Knowledge Base

Pearson

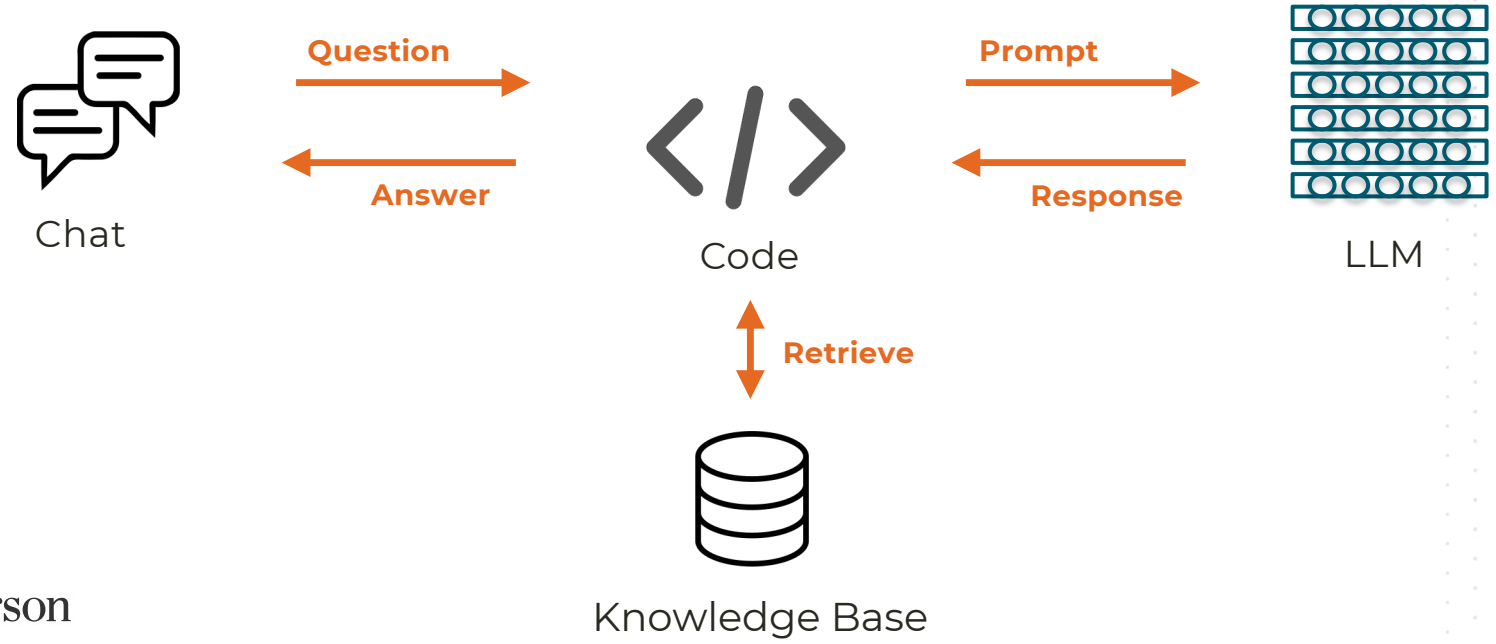
The small idea behind RAG



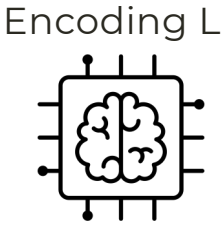
The small idea behind RAG

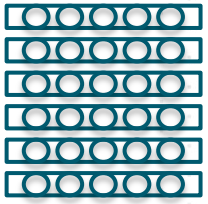


The small idea behind RAG



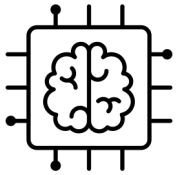
The big idea behind RAG





The big idea behind RAG

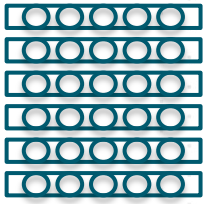
Encoding LLM



- A different type of LLM that doesn't predict the next token
- It produces numbers that reflect the meaning of the input
- If these numbers are interpreted as a vector in space, then inputs with similar meaning will be located 'close' to each other

Chat

Code

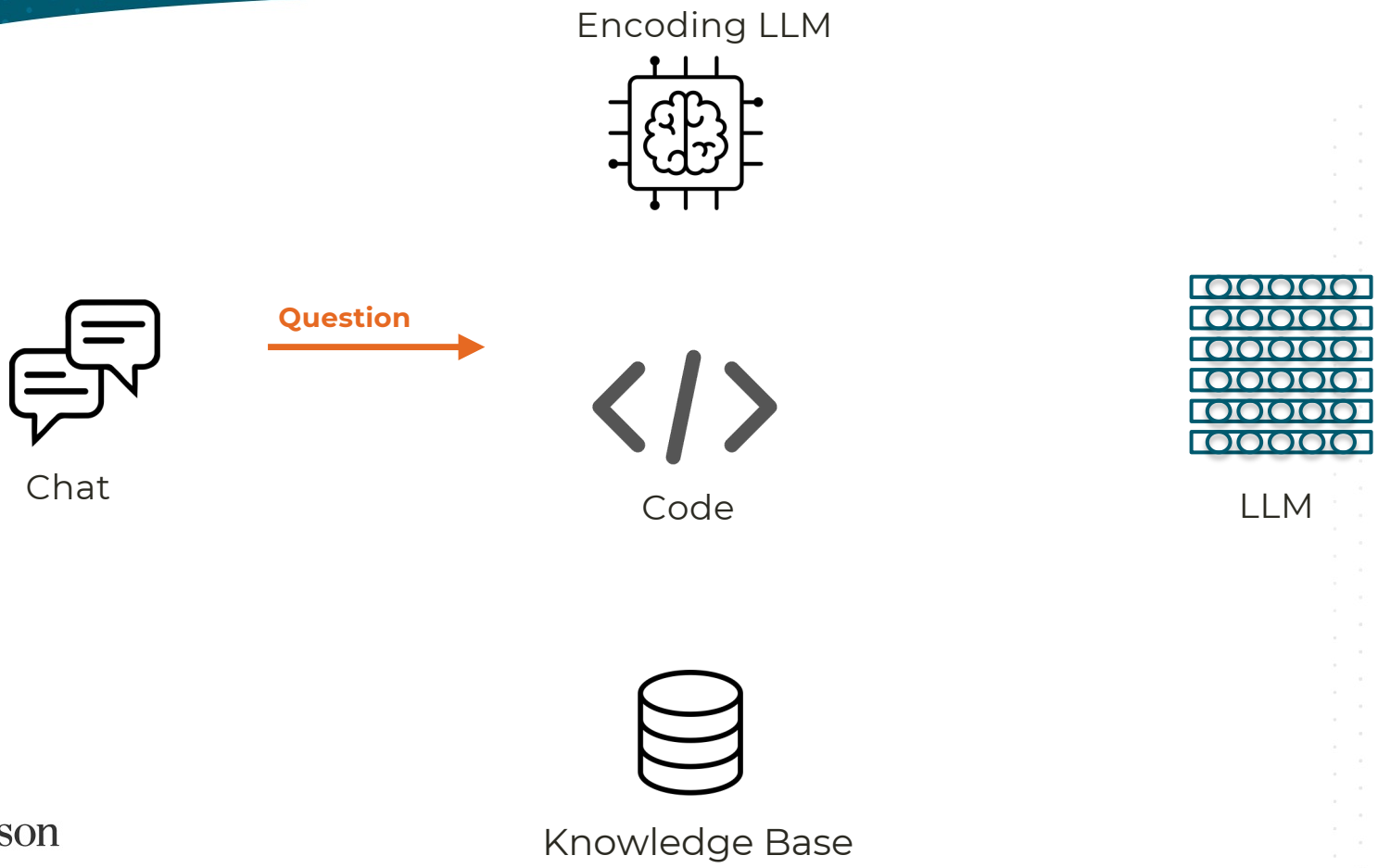


LLM

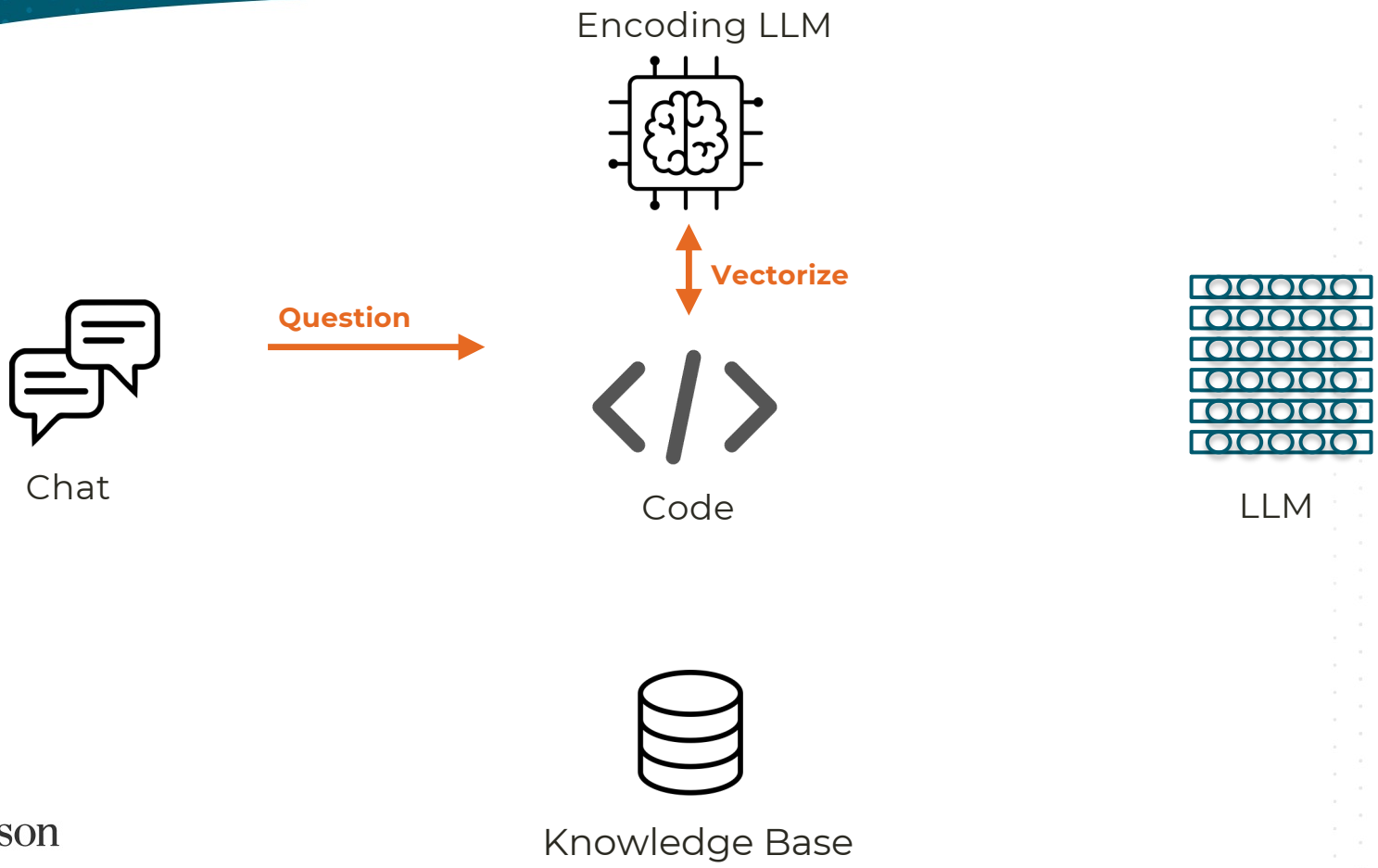
Knowledge Base

Pearson

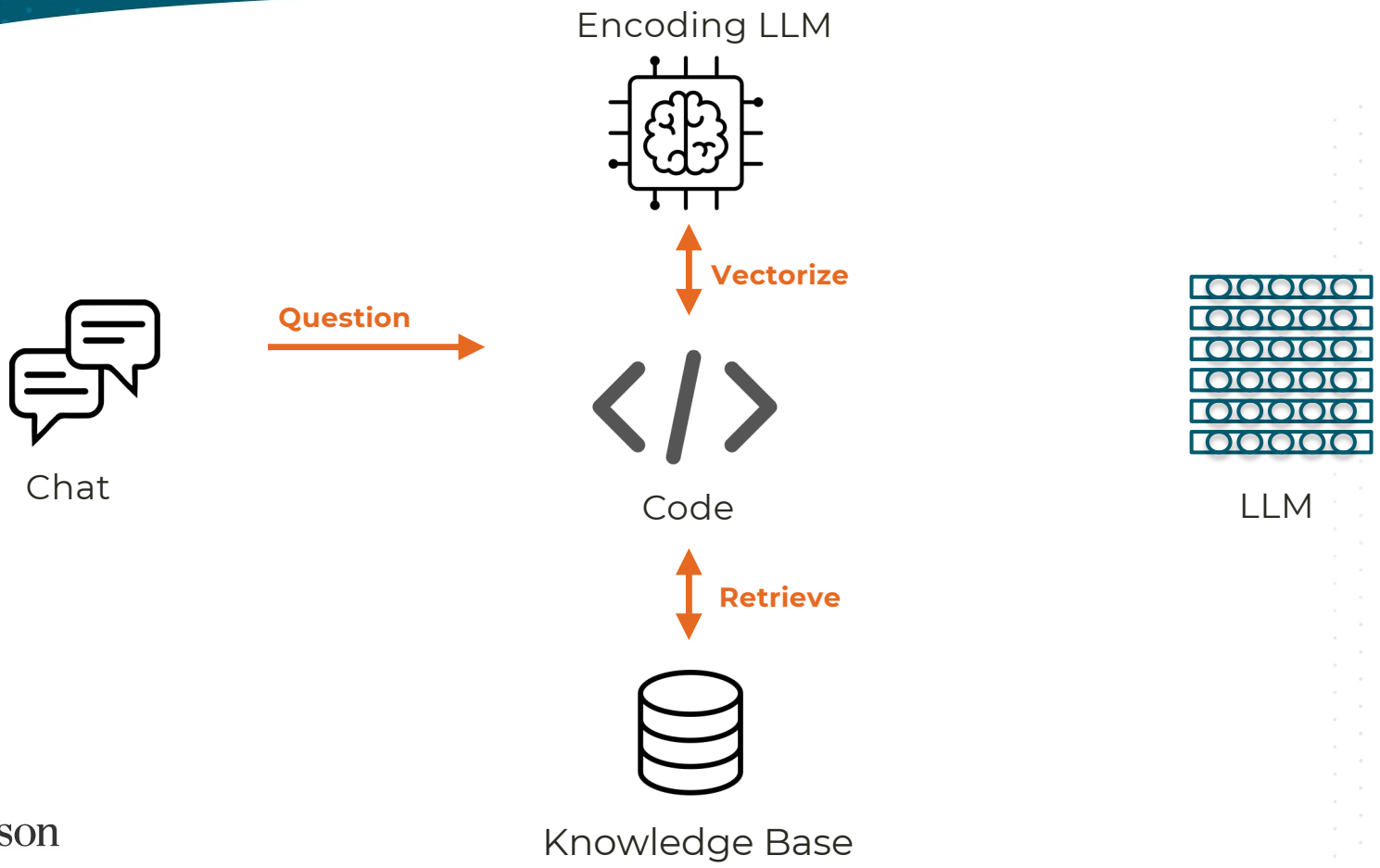
The big idea behind RAG



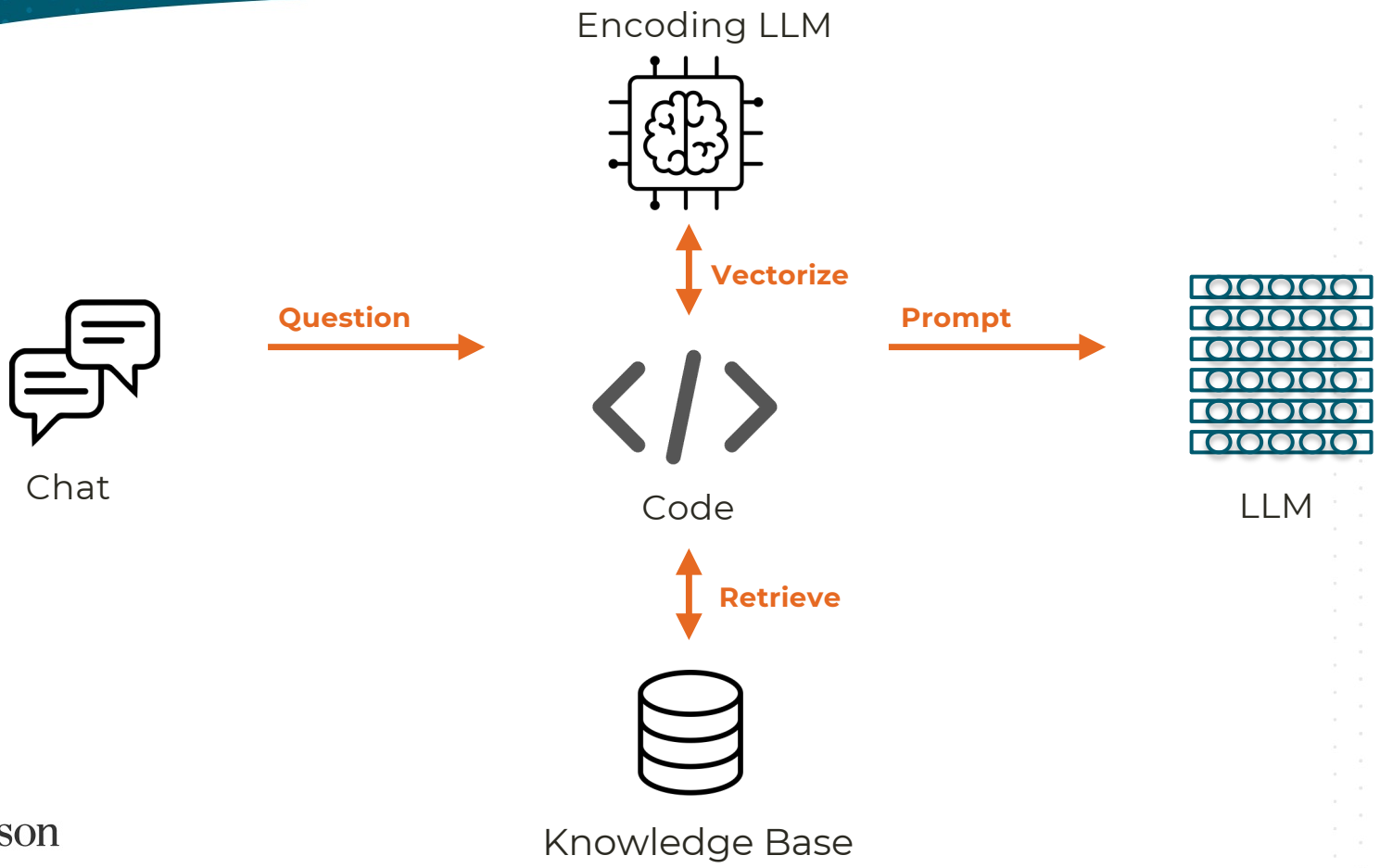
The big idea behind RAG



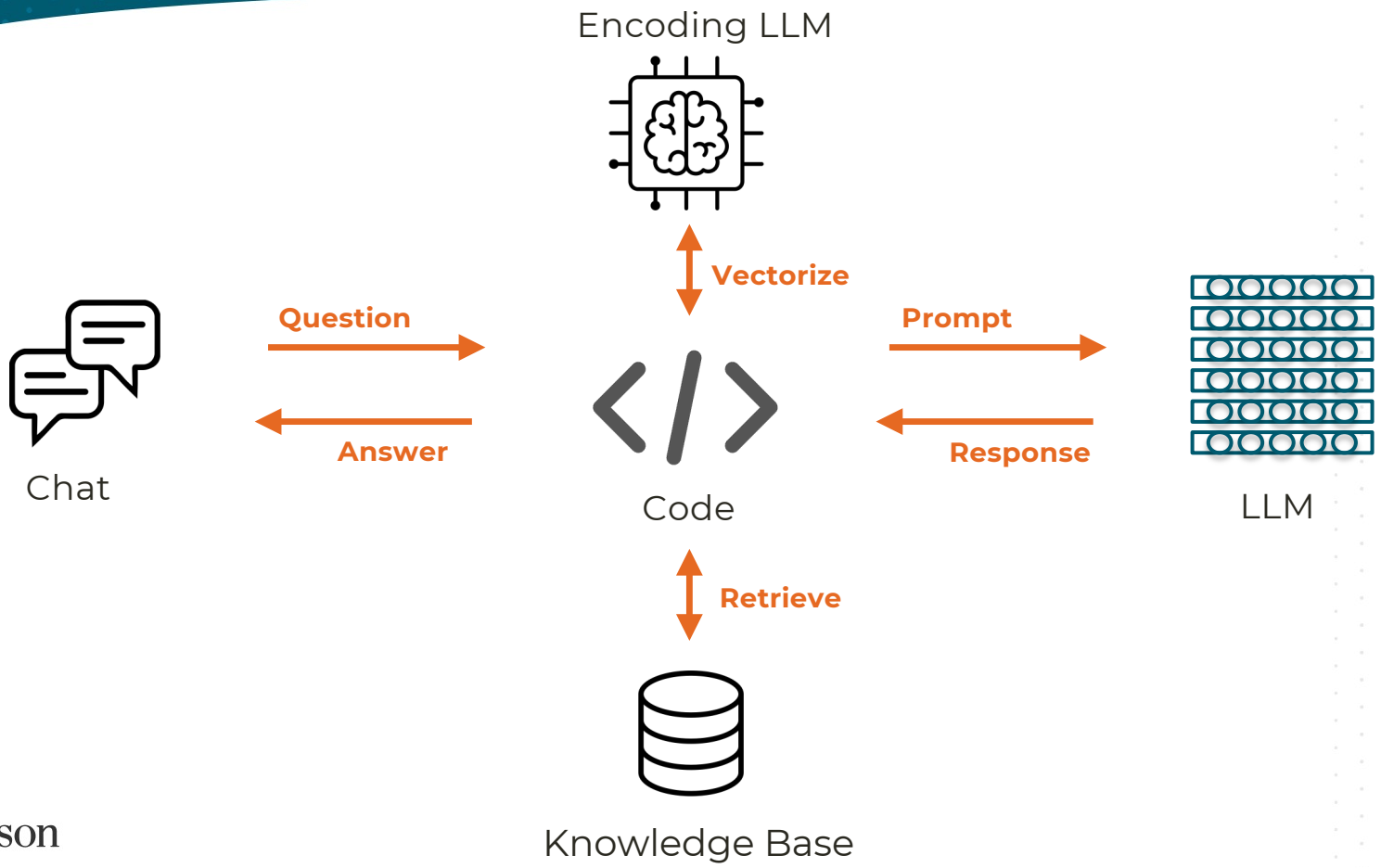
The big idea behind RAG



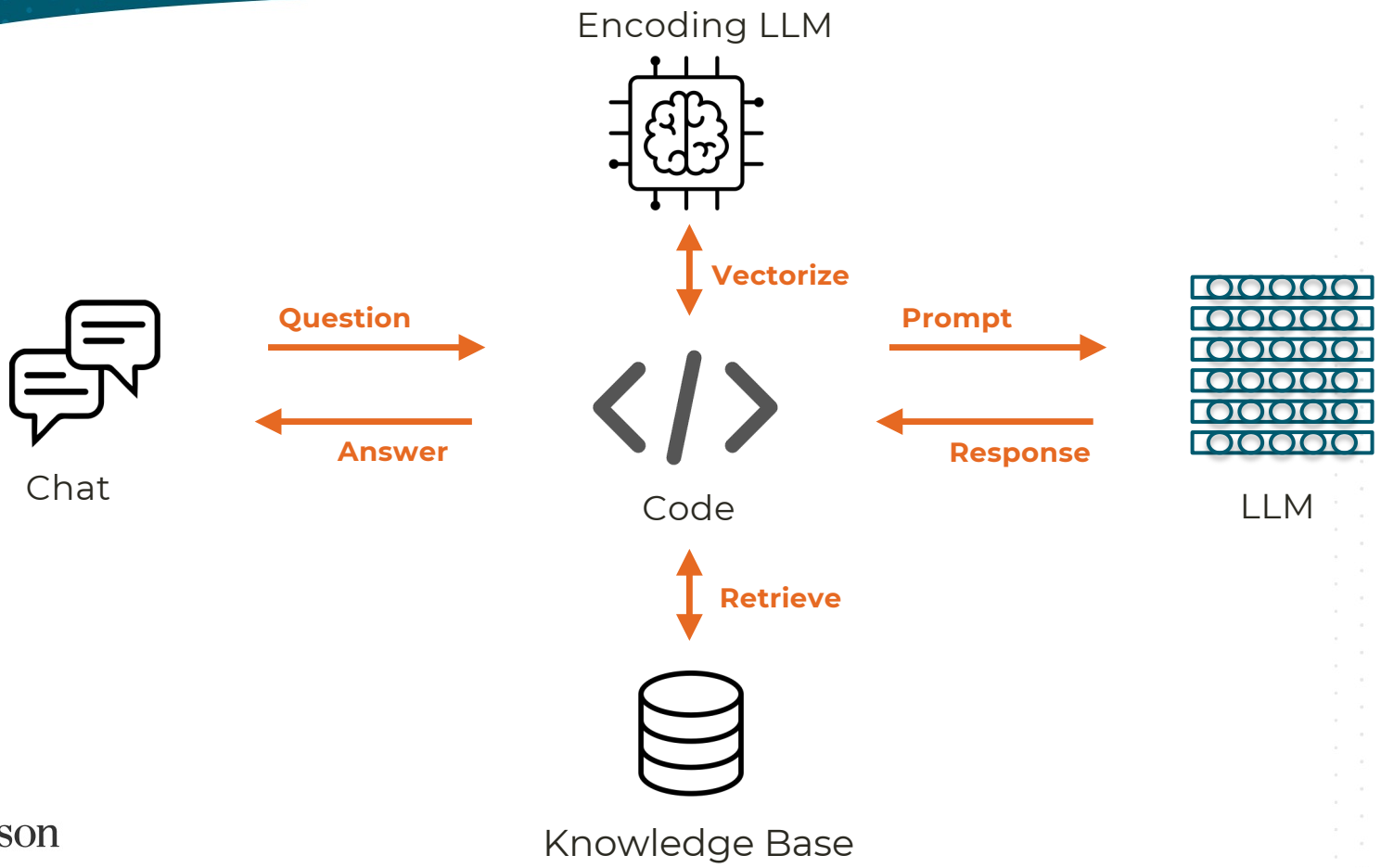
The big idea behind RAG



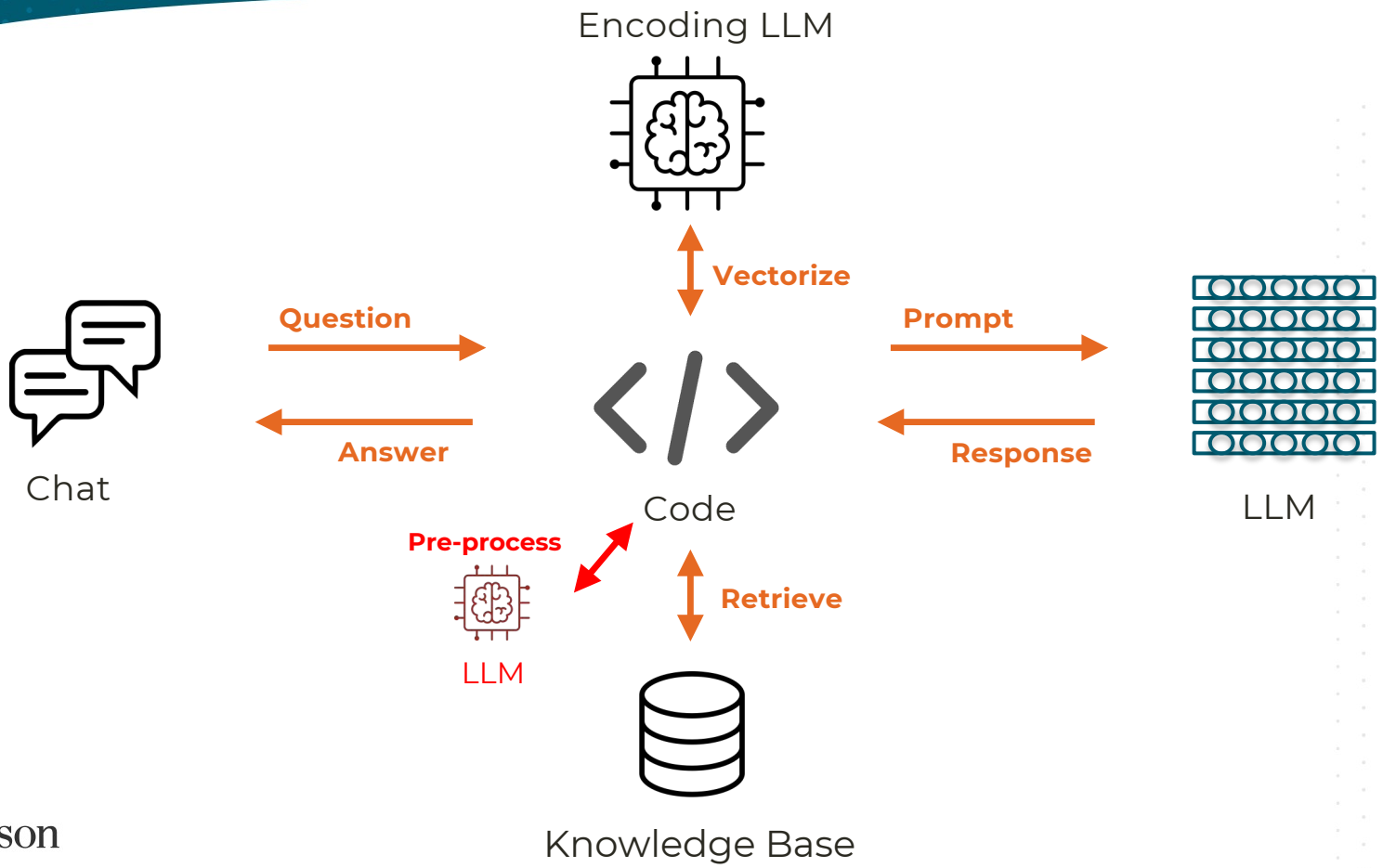
The big idea behind RAG



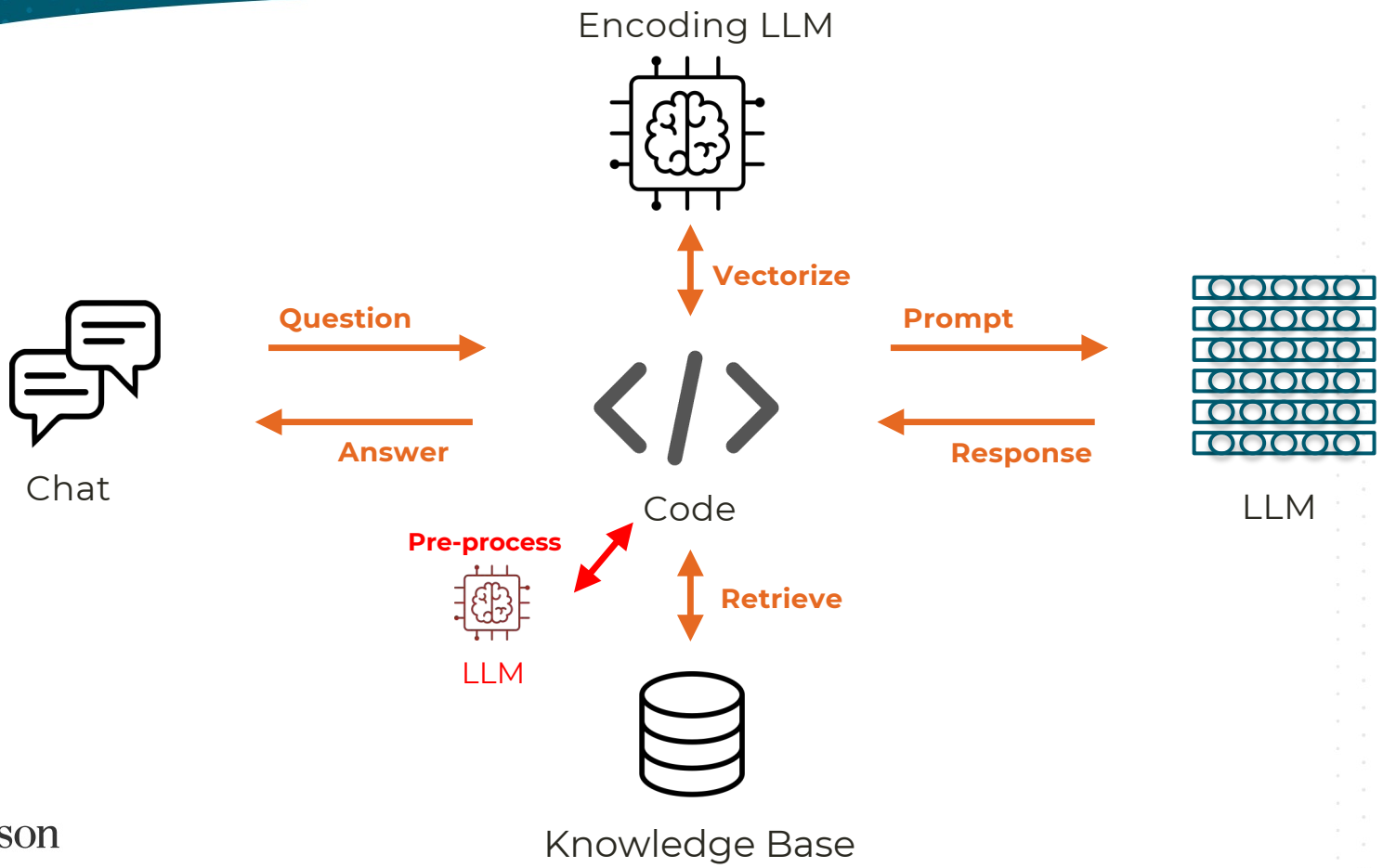
Advanced techniques: document pre-processing



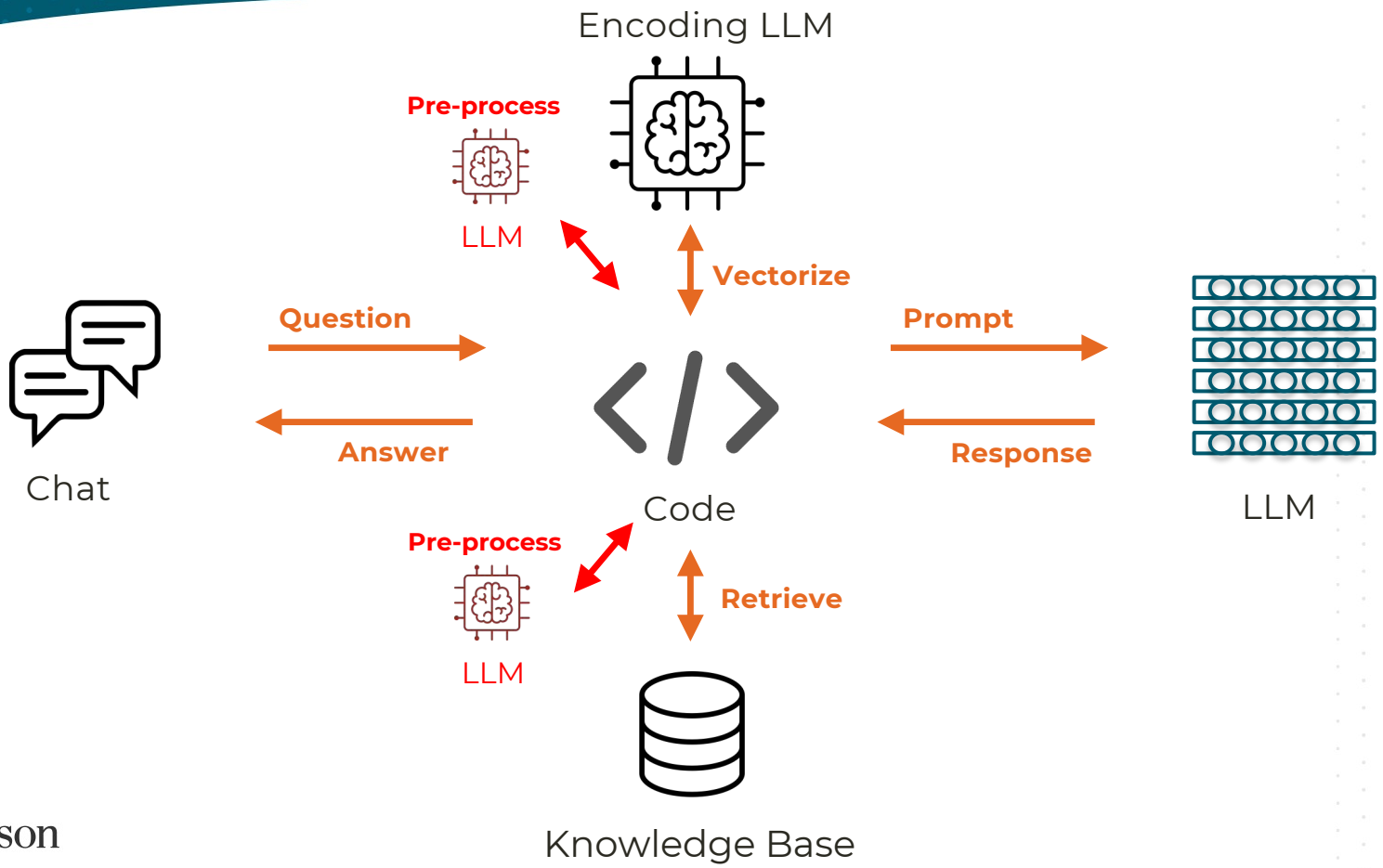
Advanced techniques: document pre-processing



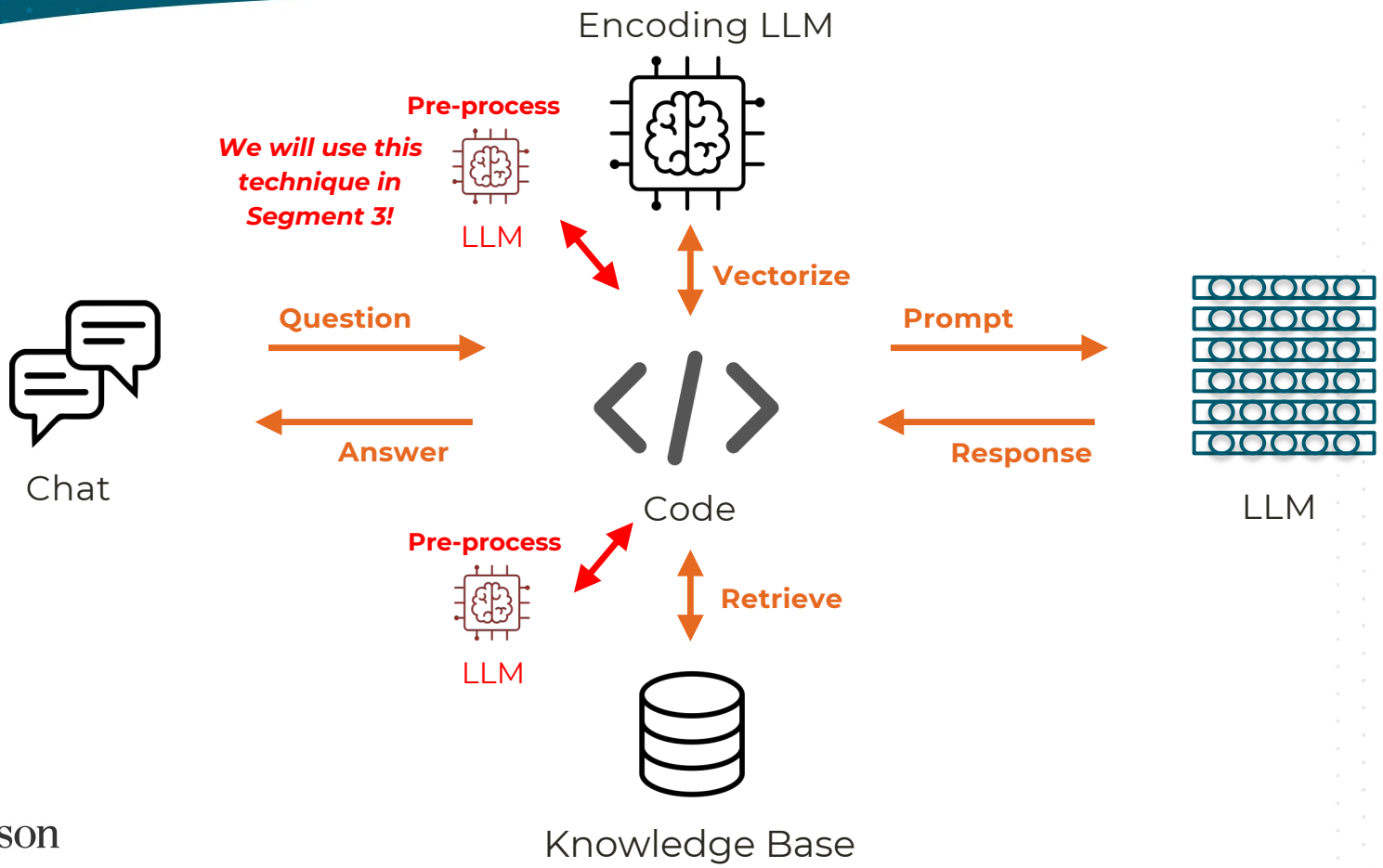
Advanced techniques: query pre-processing / rewriting



Advanced techniques: query pre-processing / rewriting



Advanced techniques: query pre-processing / rewriting



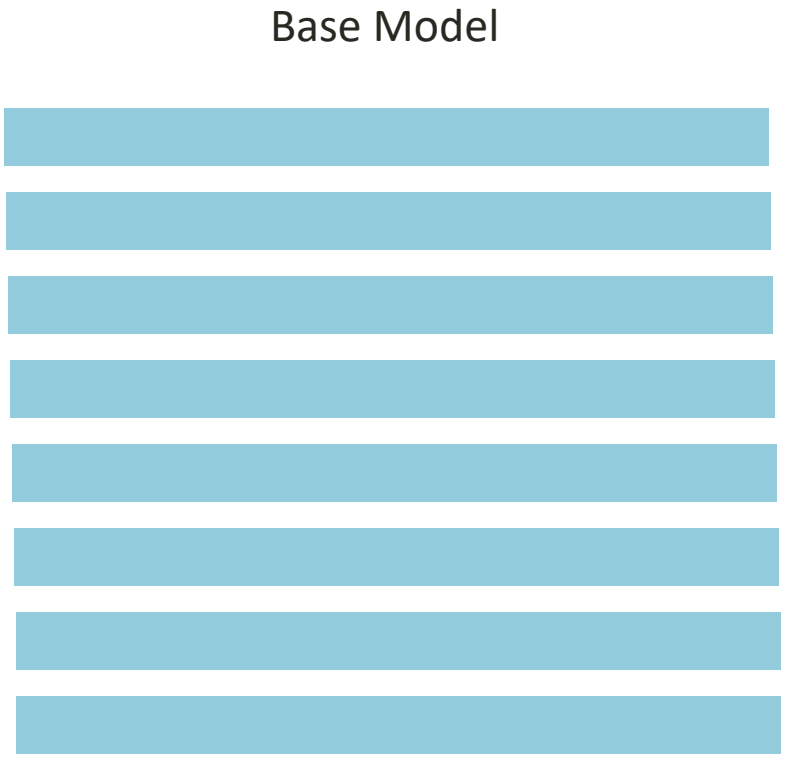
Pearson

QLoRA Fine-Tuning - motivation

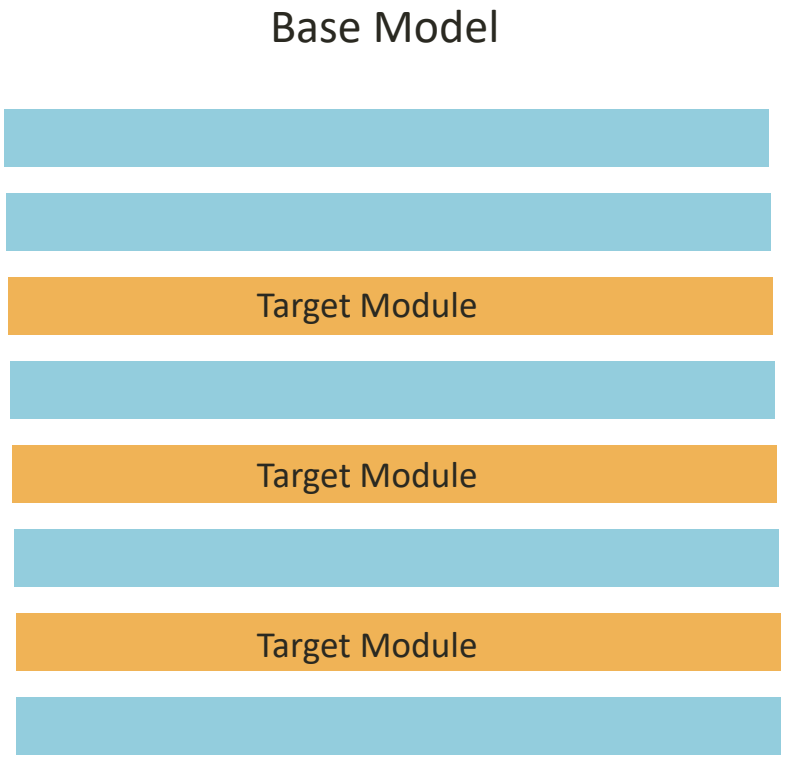
Take an existing “base” model like Llama 3.1, and train it to perform well at a specialized task, using “transfer learning”

But even ‘small’ models like Llama 3.1 8B have 8 billion parameters..

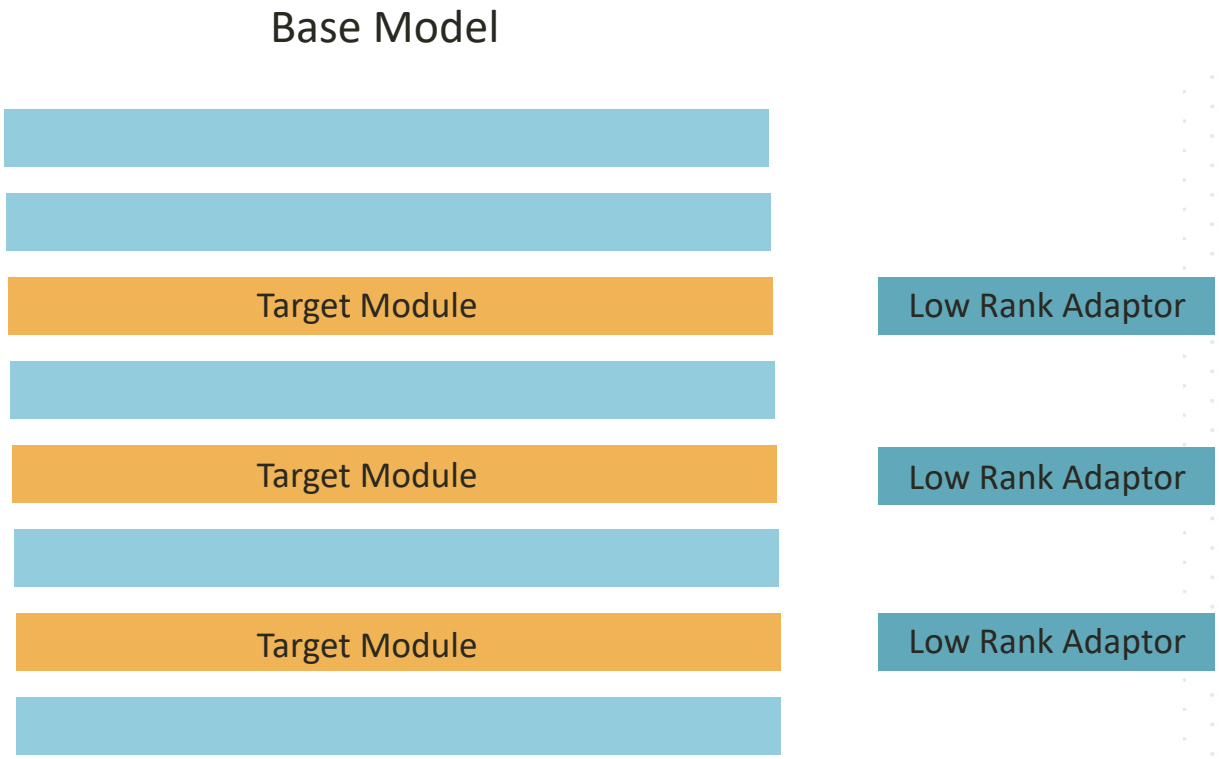
Step 1: Freeze the model weights



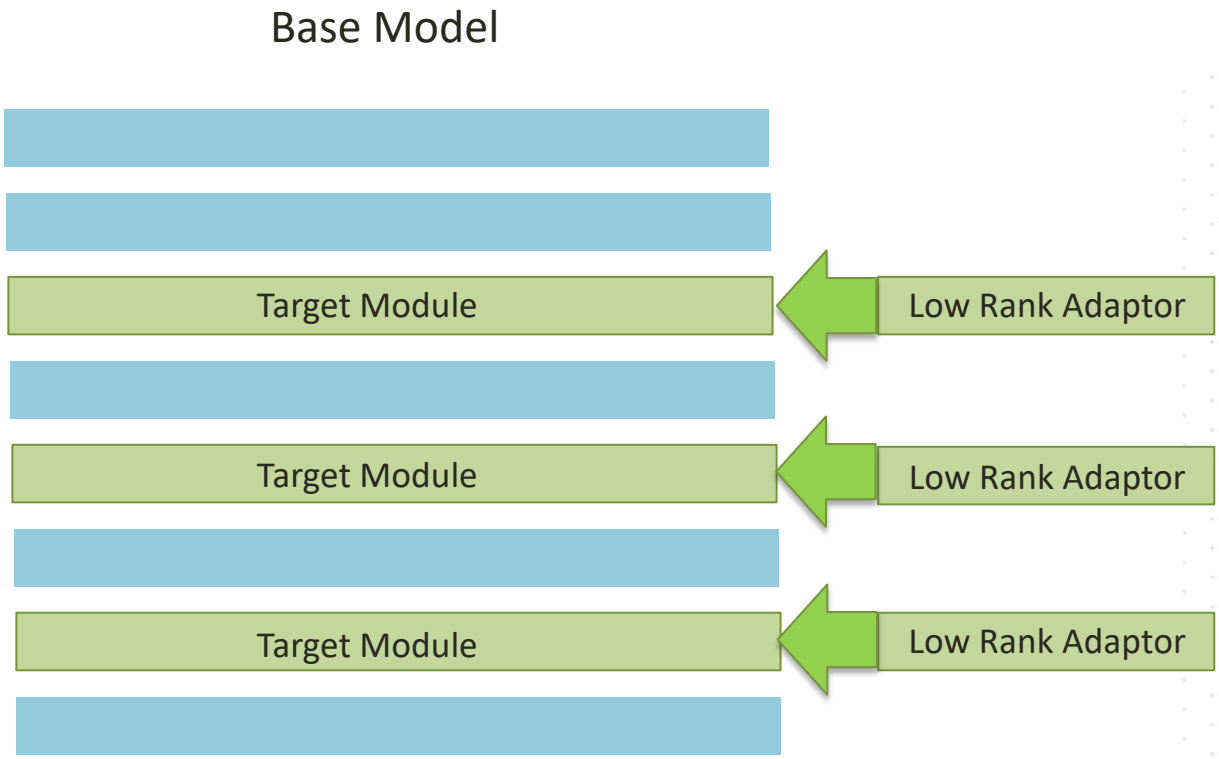
Step 2: identify a subset of layers called “Target Modules”



Step 3: New "adaptor" matrices with lower dimensions

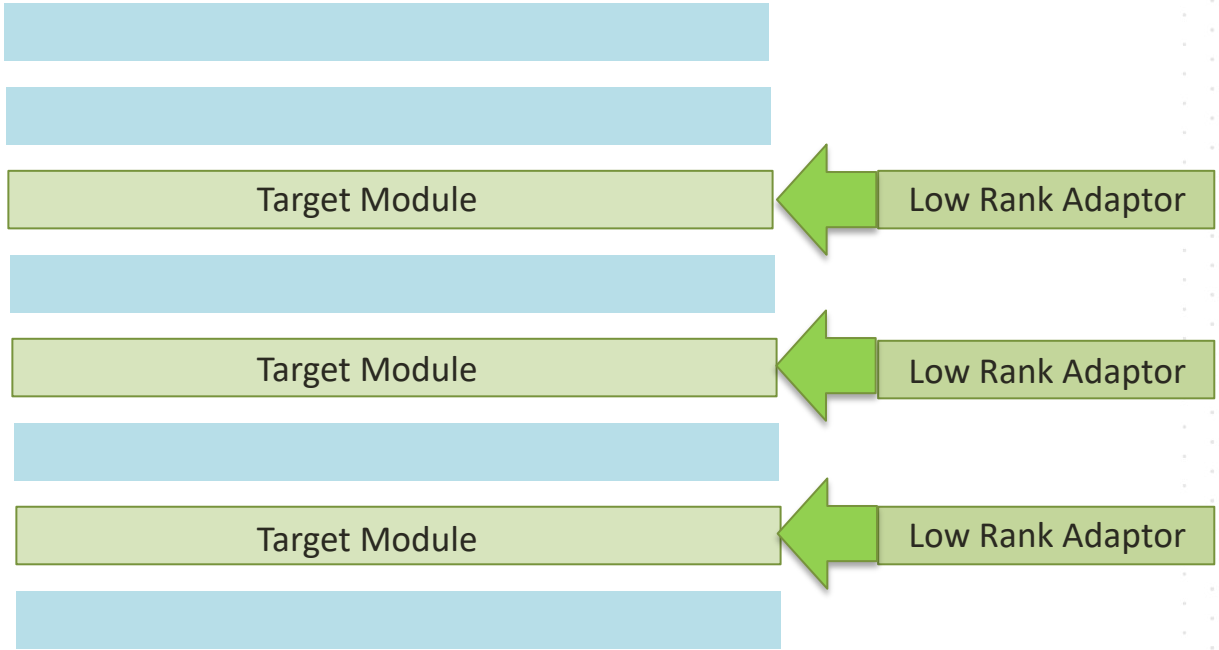


Step 4: Train these and apply them to the target modules



Step 5: Final trick: “Quantization”

Reduce the precision of the Base Model
down to 8 bits or even 4 bits



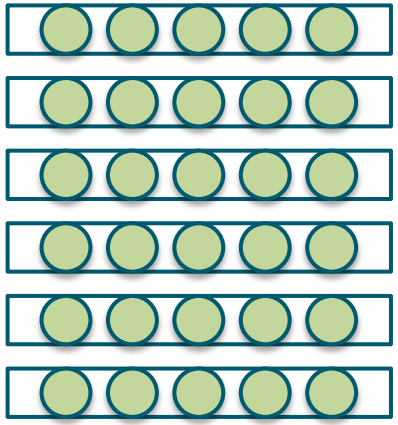
QLoRA = Quantized Low Rank Adaptation

Remember this slide on LLM training?

Input Prompt

We are living in a material <???>

LLM



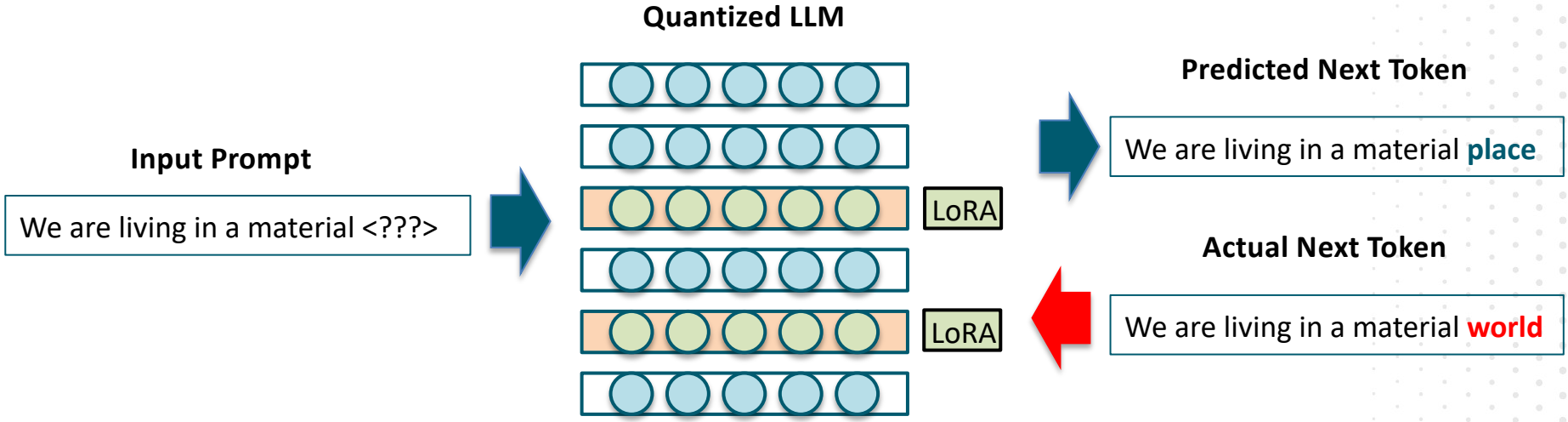
Predicted Next Token

We are living in a material **place**

Actual Next Token

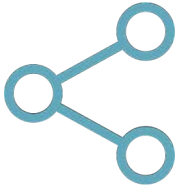
We are living in a material **world**

…this is how it looks under QLoRA

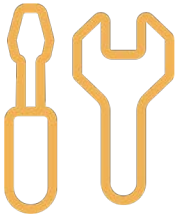


Pearson

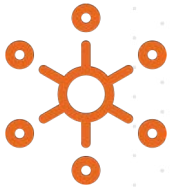
The Hallmarks of an Agentic AI solution



Breaking a larger problem into smaller steps



Using Tools / Function Calling and Structured Outputs

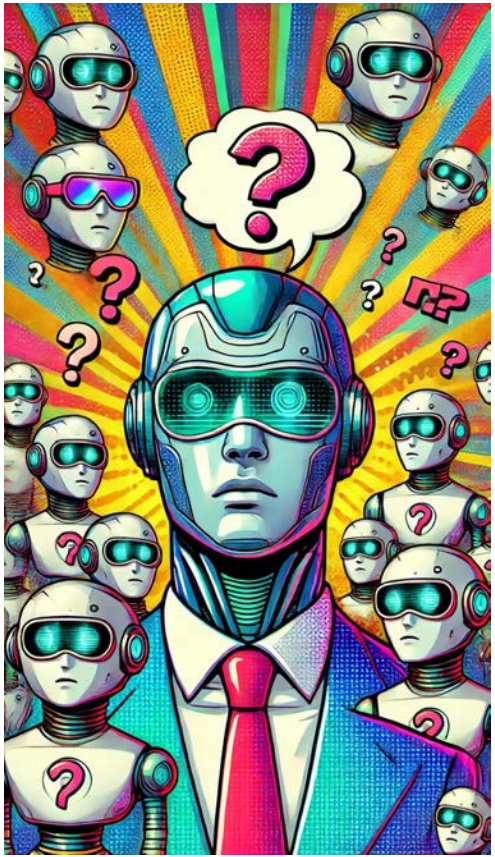


An Environment where Agents collaborate

A Planning Agent that coordinates activities

Autonomy & Memory - persists beyond a chat with a human

Decision Points with Agentic AI



- Use an abstraction layer?
  - LangGraph
  - Crew AI
  - Autogen
  - Or, direct LLM API
- How much autonomy
  - Workflow vs Agentic
  - Guard-rails for production

Do you agree with this statement?

By 2026 we will have **Artificial Super Intelligence** in which AI models will **outperform** humans across a wide range of cognitive tasks

Segment 3: The Hands-On Challenge

Our Agentic AI Architecture

Agent 1
Scanner
Pick 5 Great online deals

- Parse RSS feeds then scape deal web pages
- Call gpt-4o-mini with multi-shot prompting to parse and select
- Use Structured Outputs to guarantee output format

Pearson

Our Agentic AI Architecture

Agent 1
Scanner
Pick 5 Great online deals

- Parse RSS feeds then scape deal web pages
- Call gpt-4o-mini with multi-shot prompting to parse and select
- Use Structured Outputs to guarantee output format

Agent 2
Frontier
Predict value from similar

- RAG workflow with gpt-4o-mini with similar products in Chroma
- Use Ollama + llama3.2 to preprocess
- Use Sentence Transformer + all-MiniLM-L6-v2 to vectorize

Our Agentic AI Architecture

Agent 1
Scanner
Pick 5 Great online deals

- Parse RSS feeds then scape deal web pages
- Call gpt-4o-mini with multi-shot prompting to parse and select
- Use Structured Outputs to guarantee output format

Agent 2
Frontier
Predict value from similar

- RAG workflow with gpt-4o-mini with similar products in Chroma
- Use Ollama + llama3.2 to preprocess
- Use Sentence Transformer + all-MiniLM-L6-v2 to vectorize

Agent 3
Specialist
Predict value using skills

- Use Llama 3.1 8B
- Fine-tuned using QLoRA using Google Colab GPU boxes
- Running on serverless AI platform modal.com

Our Agentic AI Architecture

Agent 1
Scanner
Pick 5 Great online deals

- Parse RSS feeds then scape deal web pages
- Call gpt-4o-mini with multi-shot prompting to parse and select
- Use Structured Outputs to guarantee output format

Agent 2
Frontier
Predict value from similar

- RAG workflow with gpt-4o-mini with similar products in Chroma
- Use Ollama + llama3.2 to preprocess
- Use Sentence Transformer + all-MiniLM-L6-v2 to vectorize

Agent 3
Specialist
Predict value using skills

- Use Llama 3.1 8B
- Fine-tuned using QLoRA using Google Colab GPU boxes
- Running on serverless AI platform modal.com

Agent 4
Messenger
Send push notification

- Send push notifications to your phone
- Use Claude 3.5 Sonnet to craft the notification

Pearson

Our Agentic AI Architecture

Agent 5
Planner
Notify you of the best deal

- Use gpt-4o-mini to orchestrate with Tools
- 2 frontier models and 3 open-source models
- For each deal surfaced, 23 LLM calls take place!!

Agent 1
Scanner
Pick 5 Great online deals

- Parse RSS feeds then scape deal web pages
- Call gpt-4o-mini with multi-shot prompting to parse and select
- Use Structured Outputs to guarantee output format

Agent 2
Frontier
Predict value from similar

- RAG workflow with gpt-4o-mini with similar products in Chroma
- Use Ollama + llama3.2 to preprocess
- Use Sentence Transformer + all-MiniLM-L6-v2 to vectorize

Agent 3
Specialist
Predict value using skills

- Use Llama 3.1 8B
- Fine-tuned using QLoRA using Google Colab GPU boxes
- Running on serverless AI platform modal.com

Agent 4
Messenger
Send push notification

- Send push notifications to your phone
- Use Claude 3.5 Sonnet to craft the notification

Pearson

Our Agentic AI Architecture

Agent 5
Planner
Notify you of the best deal

- Use **gpt-4o-mini** to orchestrate with **Tools**
- 2 frontier models and 3 open-source models
- For each deal surfaced, 23 LLM calls take place!!

Agent 1
Scanner
Pick 5 Great online deals

- Parse RSS feeds then scape deal web pages
- Call **gpt-4o-mini** with **multi-shot prompting** to parse and select
- Use **Structured Outputs** to guarantee output format

Agent 2
Frontier
Predict value from similar

- **RAG** workflow with **gpt-4o-mini** with similar products in **Chroma**
- Use **Ollama** + **llama3.2** to preprocess
- Use **Sentence Transformer** + **all-MiniLM-L6-v2** to vectorize

Agent 3
Specialist
Predict value using skills

- Use **Llama 3.1 8B**
- Fine-tuned using **QLoRA** using **Google Colab** GPU boxes
- Running on serverless AI platform **modal.com**

Agent 4
Messenger
Send push notification

- Send push notifications to your phone
- Use **Claude 3.5 Sonnet** to craft the notification

Pearson


To The Lab

Wrapping up

- **Segment 1** reviewed the role of LLM Engineer and model APIs
- **Segment 2** covered models, tools and techniques
- **Segment 3** built an Agentic AI solution with RAG, Tools, Techniques

Please go back and run the Jupyter Notebooks in your own time!

More information linked in the GitHub README.md
to help you directly apply this to your business

Final extra example projects for those who stayed to the end!

Fine-tuning an LLM on your own Text Message history to create a simulation of yourself!

https://edwarddonner.com/2024/01/02/fine-tuning-an-llm-on-240k-text-messages/

See LLMs try to outwit each other:

https://edwarddonner.com/outsmart/

Next steps

1. Clone the GitHub repo and run & tweak the examples
2. Review the resources, reach out with questions
3. Attend related Live Events

Next steps

1. Clone the GitHub repo and run & tweak the examples
2. Review the resources, reach out with questions
3. Attend related Live Events

…and apply these models, tools and techniques to your business!

THANK YOU!!

Stay in touch:
https://www.linkedin.com/in/eddonner/
ed@edwarddonner.com

Please reach out if I can help. Or for advice on applying these techniques to your business!

You'll be receiving a feedback survey – all feedback is massively appreciated and is instrumental in helping me improve.