# Getting Started with Llama2

O'REILLY®

# Querying Your Local Files Privately with Llama2

**Lucas Soares**

**21-08-2023**

# Intro

## Hi!

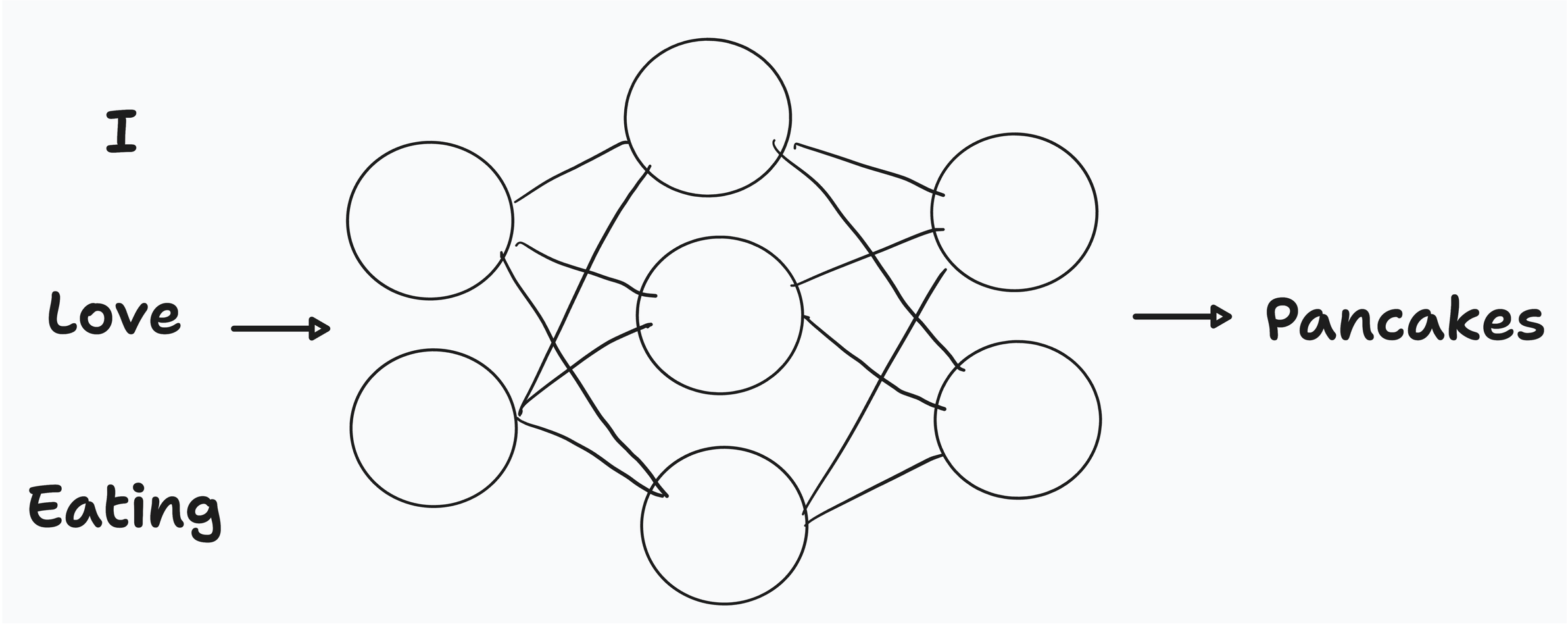
- Me!
- ML Engineer, O’Reilly Instructor.

Quick Survey to get to know everyone!

# Interactive Methodology for this Live-Training

# Large Language Models

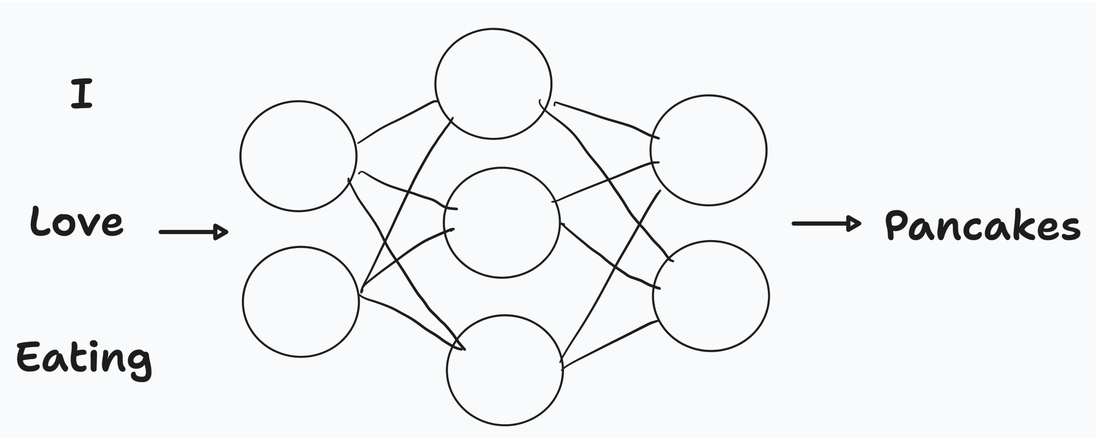
A definition



# Large Language Models

## As Probability Distributions

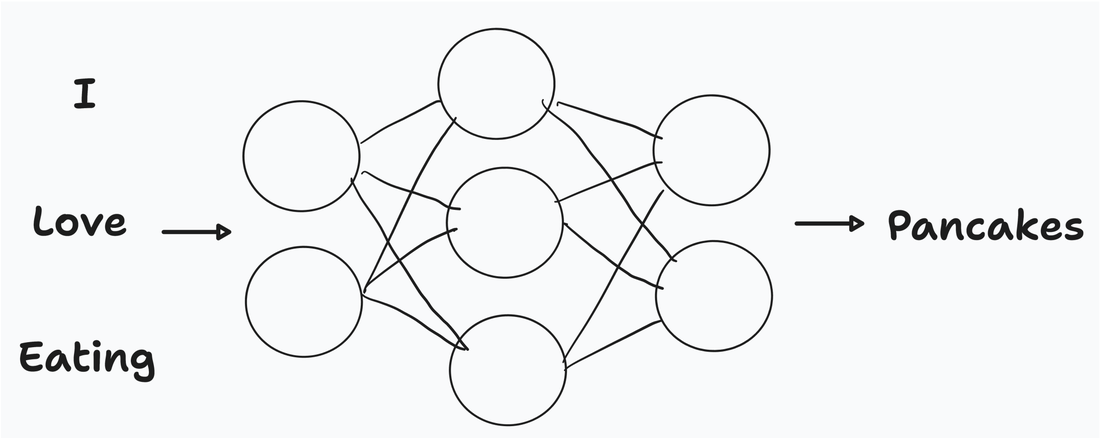
At their core, LLMs can be seen as distributions over words.



# Large Language Models

## As Probability Distributions

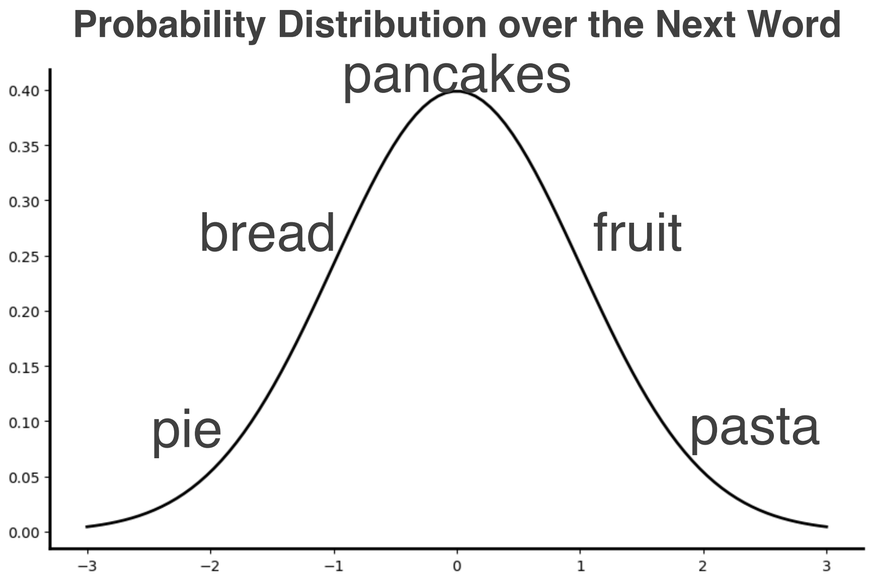
- At their core, LLMs can be seen as distributions over words.
- Use statistical models to capture patterns in text data.



# Large Language Models

## As Probability Distributions

- At their core, LLMs can be seen as distributions over words.
- Use statistical models to capture patterns in text data.
- They calculate the likelihood of each word occurring given the context.

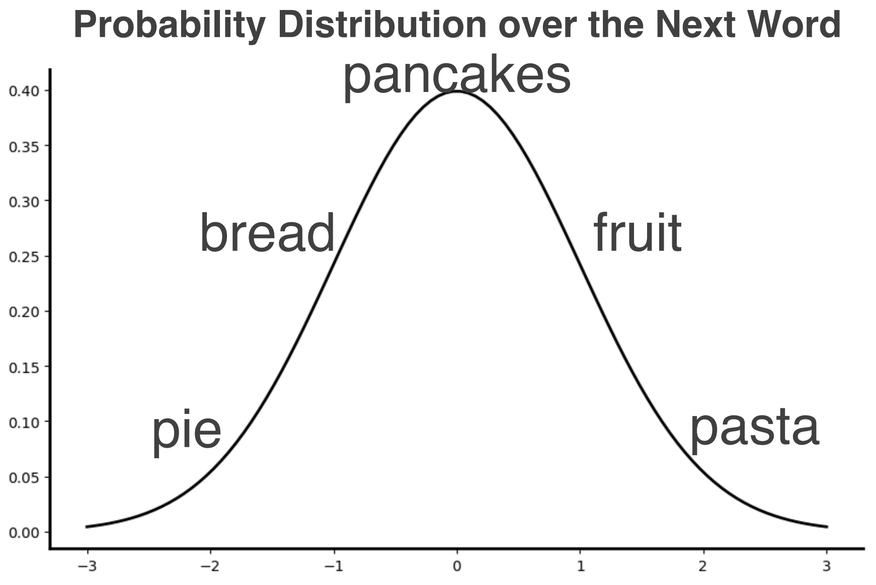



"I love eating…." → ?

# Large Language Models

## As Probability Distributions

- At their core, LLMs can be seen as distributions over words.
- Use statistical models to capture patterns in text data.
- They calculate the likelihood of each word occurring given the context.




"I love eating…." → ?

# Large Language Models

## N-gram models

- **Unigram model**: each token/word is independently modeled.

# Large Language Models

## N-gram models

- **Unigram model**: each token/word is independently modeled.
- Sentence S = *"When the bough breaks, the cradle will fall."*

**Unigram Word Probabilities**

| p = 0.86 | p = 0.93 | p = 0.22 | p = 0.66 | p = 0.66 | p = 0.37 | p = 0.15 | p = 0.1 |
|---|---|---|---|---|---|---|---|
| When | the | bough | breaks | the | cradle | will | fall |

# Large Language Models

## N-gram models

```python
import random
sentence = "When the bough breaks the craddle will fall"

tokens = sentence.split(" ")
# First, the token_probs list is created with random values
token_probs = [random.random() for _ in tokens]
# Then, the probabilities dictionary is created
# The dictionary has keys of the form 'word1 word2', where 'word1' and
'word2' are consecutive words in the sentence
# The value for each key is the conditional probability of 'word2' given
'word1'
probabilities = {}
for i in range(len(tokens)):
    if tokens[i] in probabilities.keys():
        probabilities[f"{str(tokens[i])+'-2'}"] = round(token_probs[i],2)
    else:
        probabilities[f"{tokens[i]}"] = round(token_probs[i],2)
probabilities
```

**Unigram Word Probabilities**

| p = 0.86 | p = 0.93 | p = 0.22 | p = 0.66 | p = 0.66 | p = 0.37 | p = 0.15 | p = 0.1 |
|---|---|---|---|---|---|---|---|
| When | the | bough | breaks | the | cradle | will | fall |

$$P(S) = P(\text{"When"}) \times P(\text{"the"}) \times P(\text{"bough"}) \times P(\text{"breaks"}) \times P(\text{"the"}) \times P(\text{"cradle"}) \times P(\text{"will"}) \times P(\text{"fall"})$$

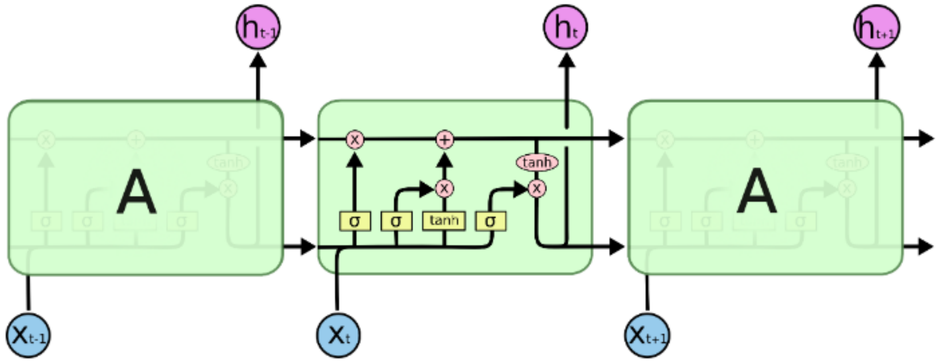
The probability of the sentence is the product of the individual probabilities:

# Large Language Models

## Sequence to Sequence Models: LSTMs

**LSTMs**, forgetting mechanism to model longer sentences and maintain context.

(Hochreiter and Jürgen Schmidhuber 1997)

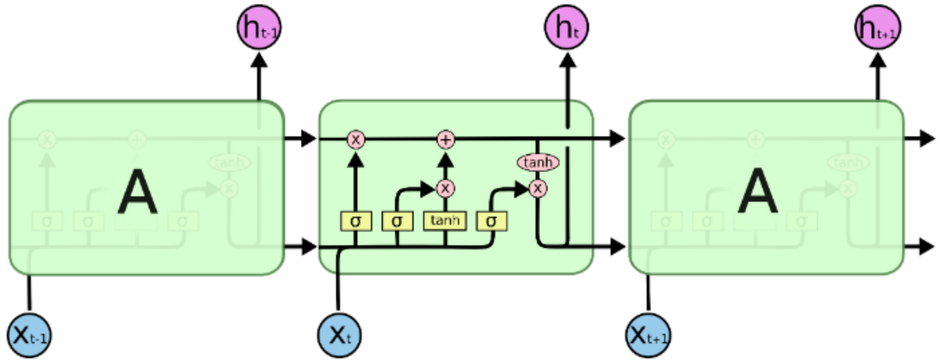



Understanding LSTM Networks by Christopher Olah

# Large Language Models

## Sequence to Sequence Models: LSTMs

- **LSTMs**, forgetting mechanism to model longer sentences and maintain context.
  (Hochreiter and Jürgen Schmidhuber 1997)
- The probability of the sentence now is influenced by context

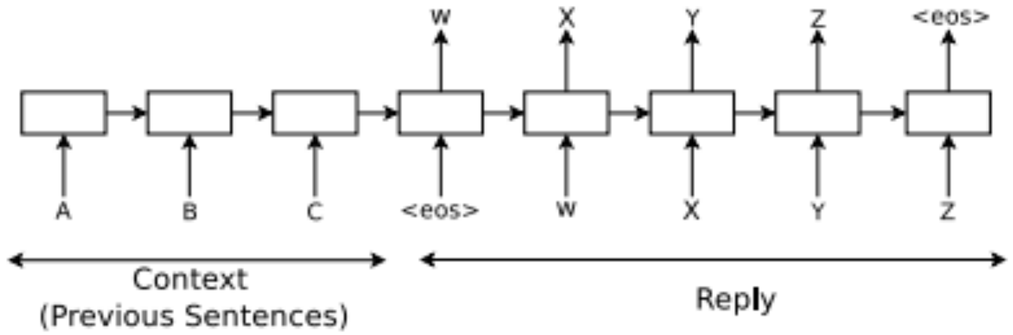



Understanding LSTM Networks by Christopher Olah

# Large Language Models

## Sequence to Sequence Models: LSTMs

**Seq2Seq** models require **processing input sequentially**

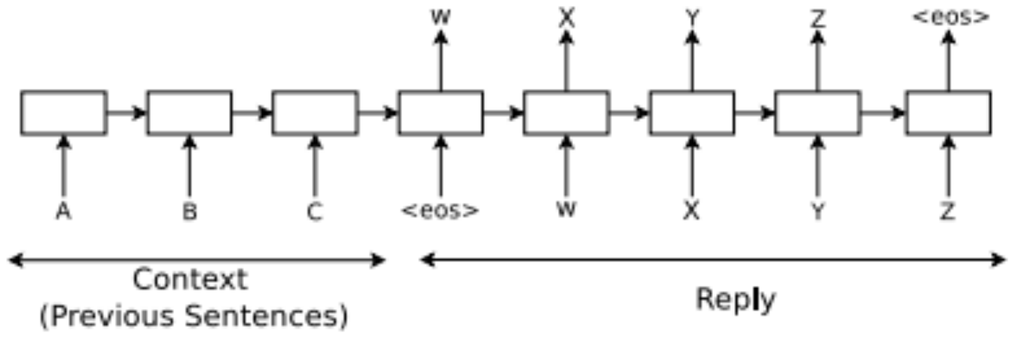



(Vinyals & Le, 2015)

# Large Language Models

Sequence to Sequence Models: LSTMs

- **Seq2Seq** models require **processing input sequentially**
- Struggle with really long complex inputs

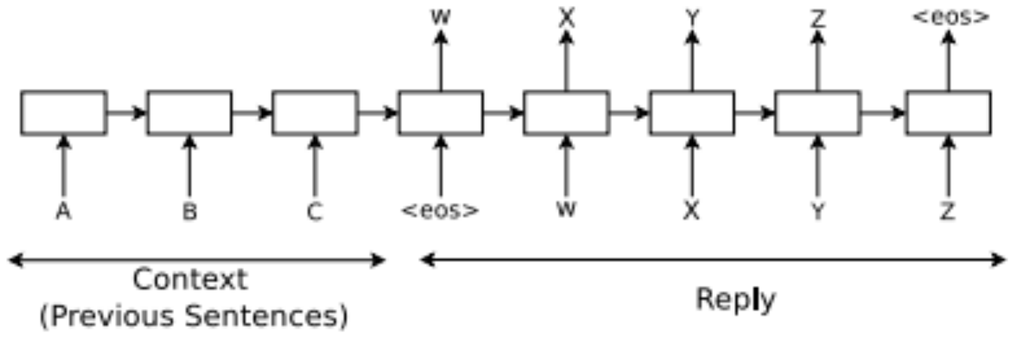



(Vinyals & Le, 2015)

# Large Language Models

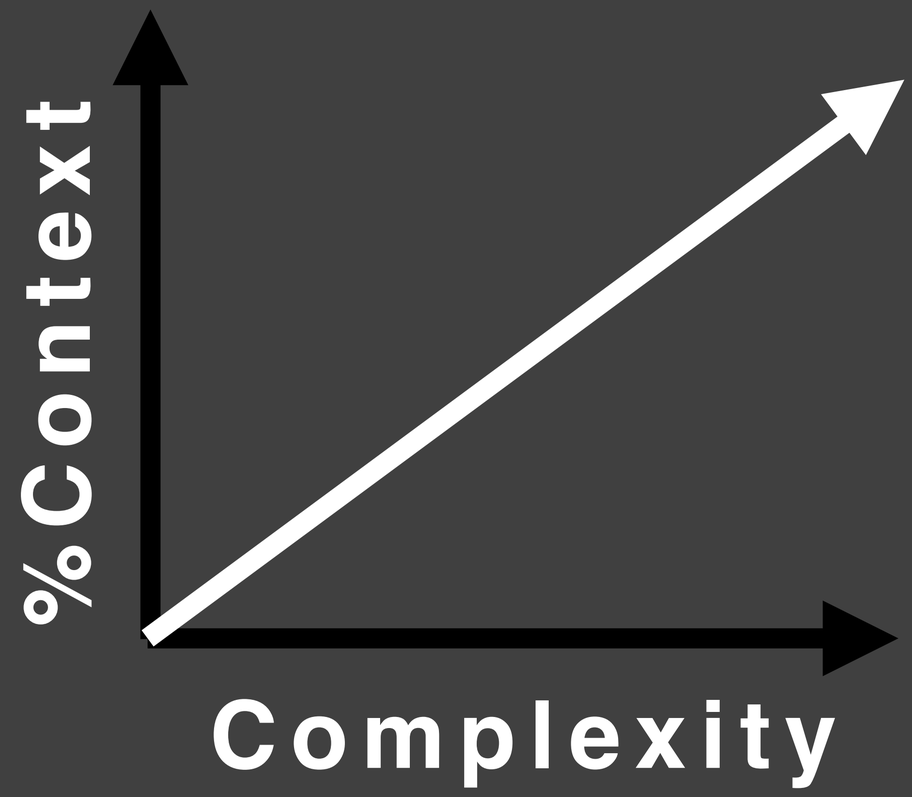
## Sequence to Sequence Models: LSTMs

- **Seq2Seq** models require **processing input sequentially**
- Struggle with really long complex inputs




(Vinyals & Le, 2015)

**Can't capture really long contexts!**


%Context
Complexity

# Essence of LLMs

Beyond pattern matching

They capture context, understand dependencies, and predict text based on patterns learned during training.

# Ok, but how?

## LLMs as pattern matchers

- LLMs are akin to probabilistic programs.
- They predict outcomes based on probabilities learned from training data

```python
import numpy as np

def llm_model(context, next_word):
    possible_next_words = ["text", "pie", "pizza", "motor", "cake"]
    next_word_probs = [0.2, 0.3, 0.4, 0.001, 0.6]
    return next_word_probs[possible_next_words.index(next_word)]

context  = ["This", "is", "a", "piece", "of"]
possible_next_words = ["text", "pie", "pizza", "motor", "cake"]
probs = []
for w in possible_next_words:
    probs.append(llm_model(context, w))

next_word = possible_next_words[np.argmax(probs)]
next_wor
```


Context/Past
LM
Future

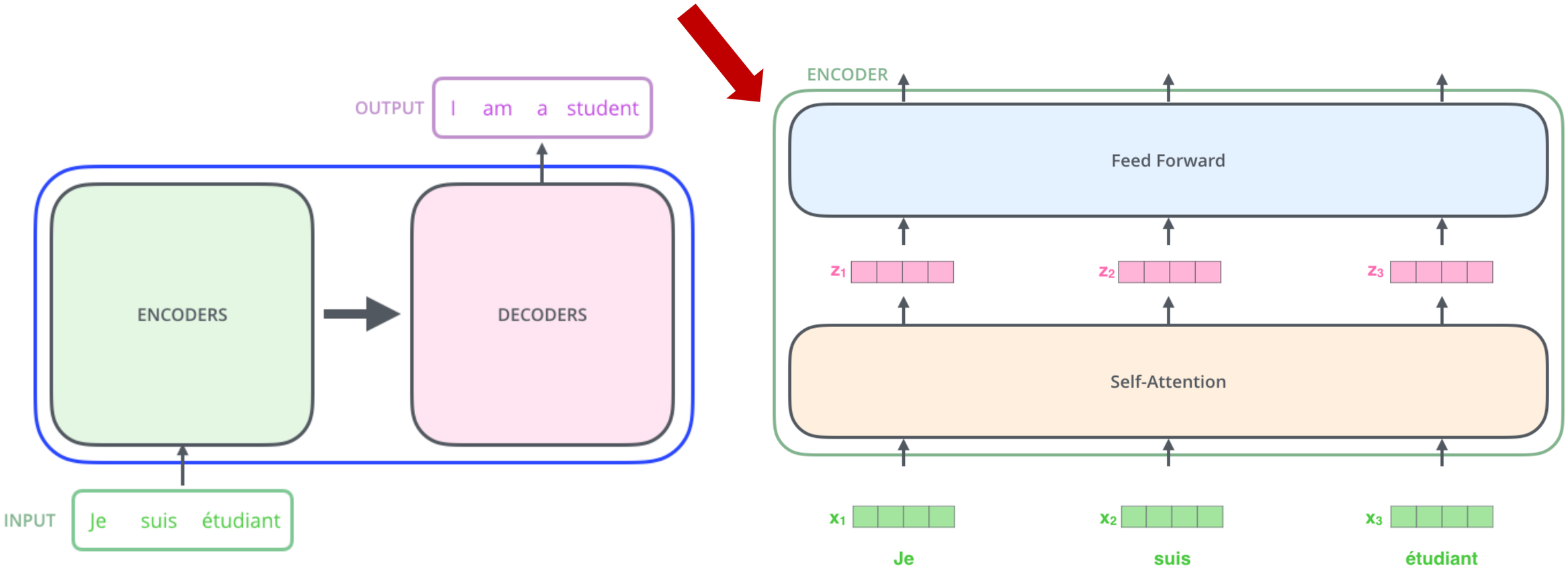
# How LLMs work

## Transformers Architecture

- Traditional sequential models struggle with context (Vaswani et al 2017)
- Transformers use attention mechanisms to capture global dependencies, enabling contextual understanding. (Vaswani et al 2017)
- The attention mechanism allows Transformers to focus on different parts of input simultaneously. (Vaswani et al 2017)
- Transformers can understand and predict based on long-range dependencies. (Vaswani et al 2017)

# How LLMs work

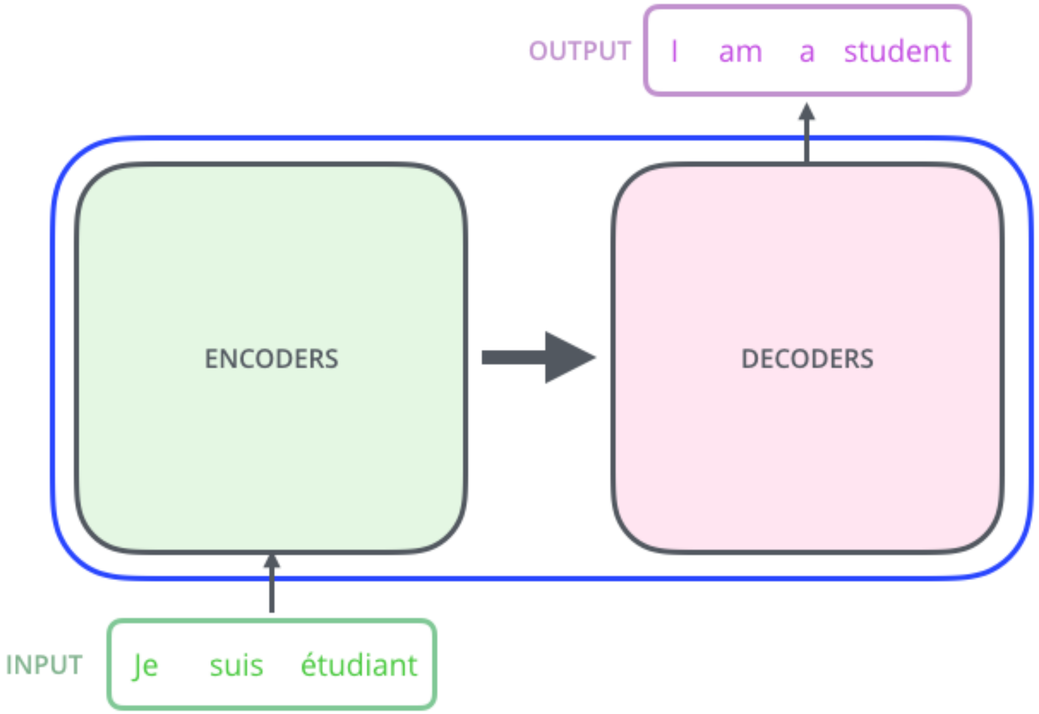
## Transformers Architecture

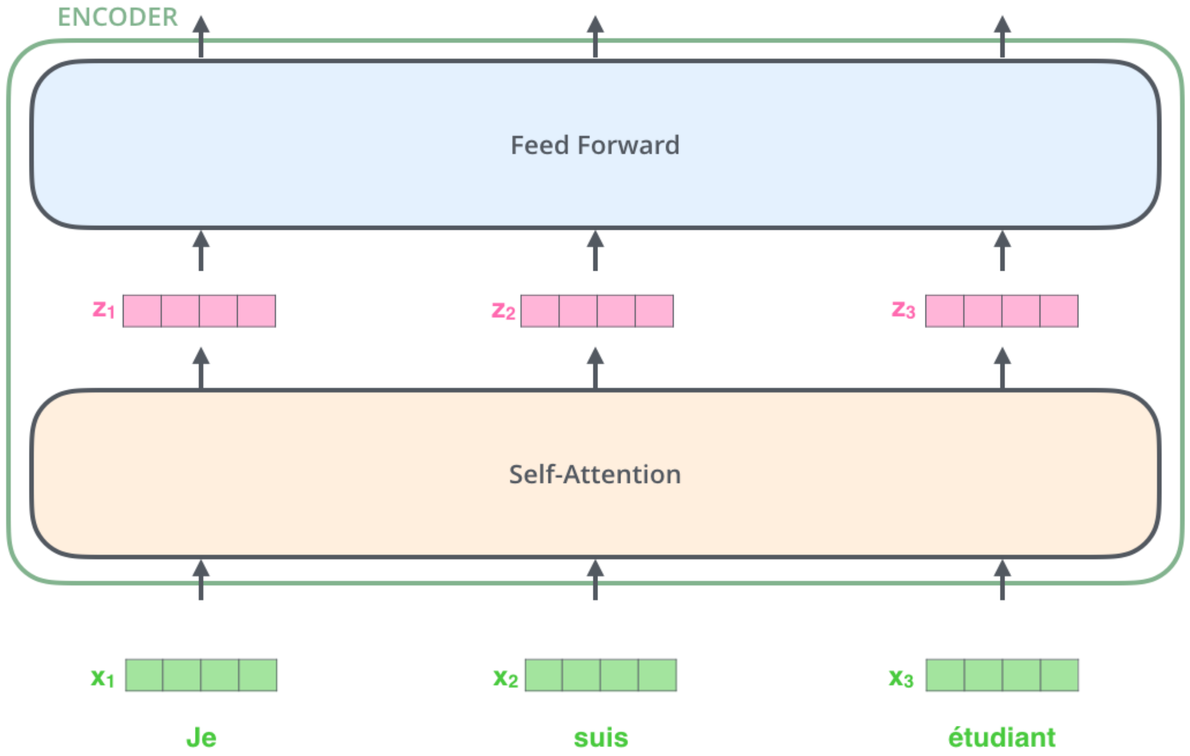



**Inputs are processed in parallel!**

# How LLMs work

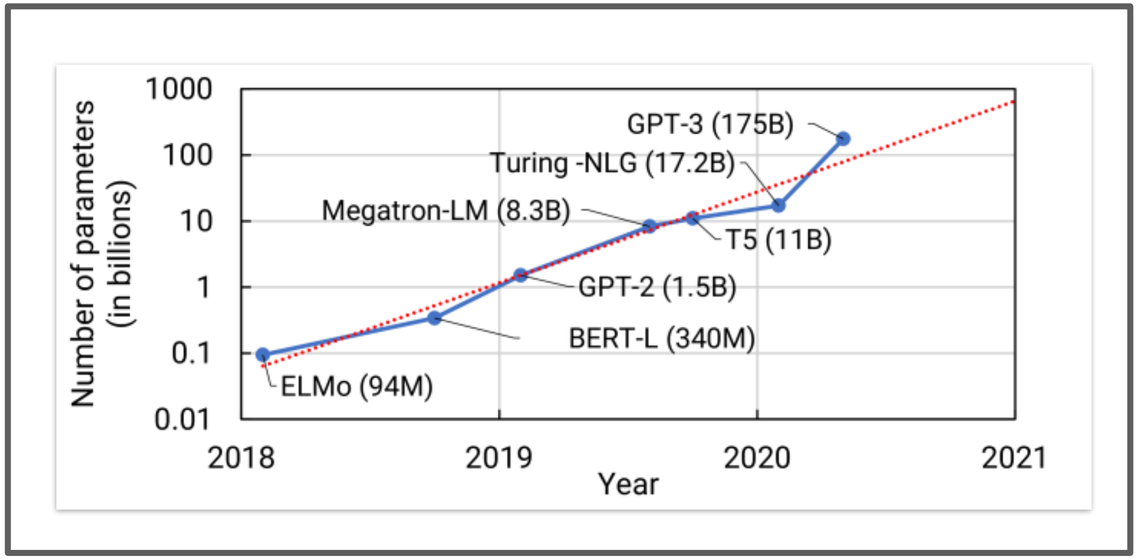
## Transformers Architecture



# Why “Large” Language Models?

- Large → Big number of parameters



# Benefits of Large Language Models

Multi-task, fine tuning, scalability

- **Multi-task Capability:** LLMs find applications in content generation, question answering, translation, tutoring, and personal assistants.
- **Fine-tunning:** LLMs can usually be fine tuned with a relatively small amount of data, making them adaptable to a wide range of tasks.
- **Scalability**: LLMs demonstrate excellent scalability to very large capacity networks and huge datasets.

# Applications of Large Language Models

- Content generation
- Q&A
- Translation
- Tutoring
- Personal Assistants

# Questions and Discussion

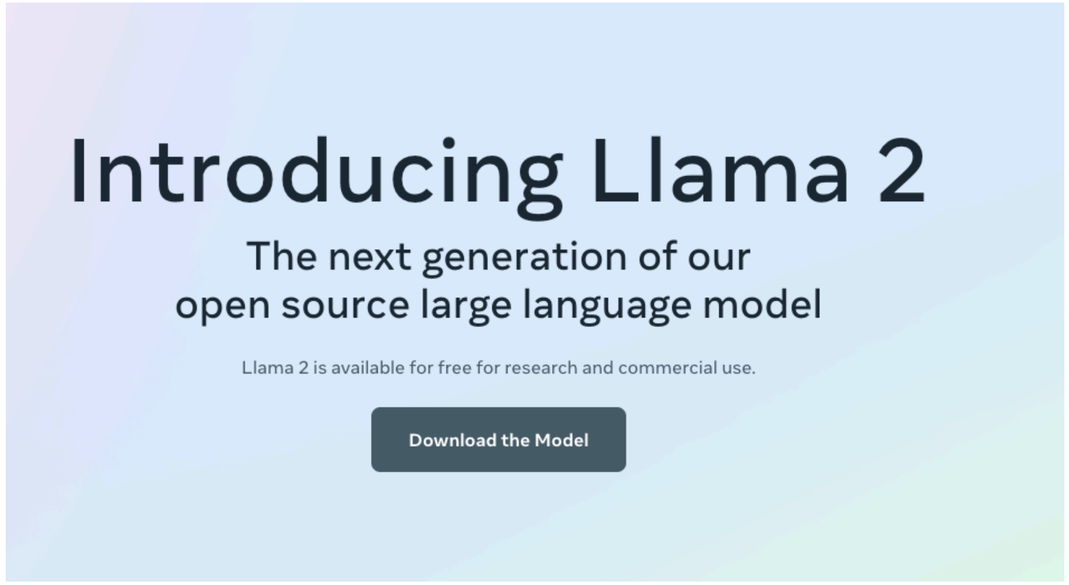
# Introduction to Llama2

Table of Contents

# Introduction to Llama2

**What**, why & how Llama2?

- LLM Released by Meta in July of 2023
- Open source with a Comercial license

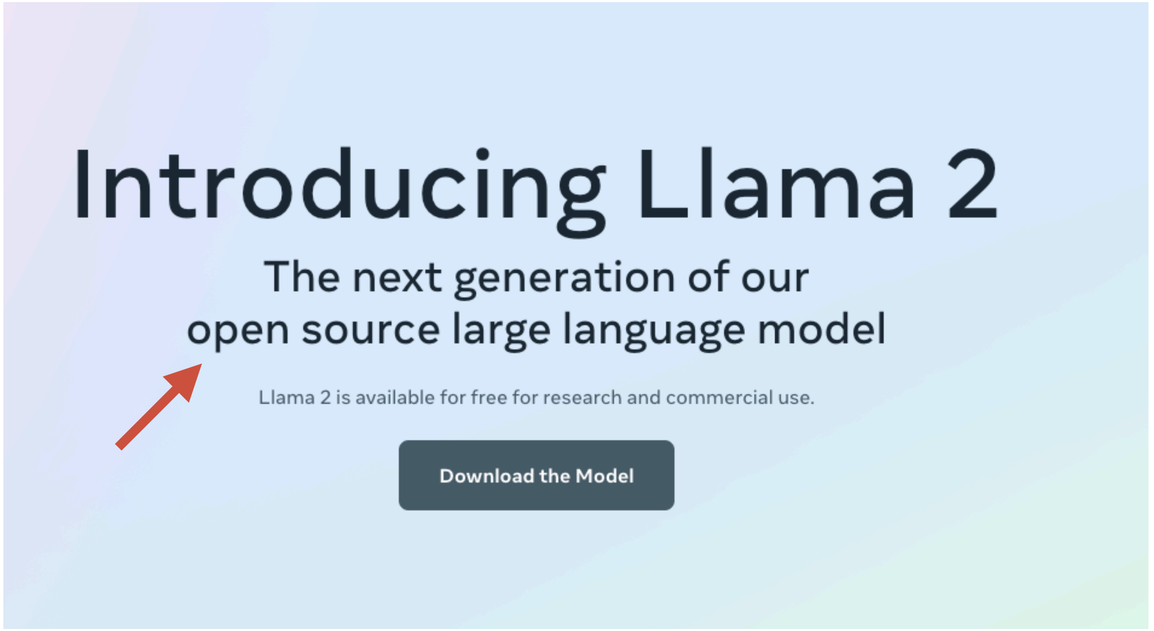



https://ai.meta.com/llama/#resources

# Introduction to Llama2

What, **why** & how Llama2?

## Llama2 is OPEN SOURCE



# Introduction to Llama2

What, **why** & how Llama2?

- **Llama2 is OPEN SOURCE**
- **Comes in 3 different sizes:**
  7b, 13B & 70B parameters

Llama 2 was trained on **40% more data** than Llama 1,
and has double the context length.

**Llama 2**

| MODEL SIZE (PARAMETERS) | PRETRAINED | FINE-TUNED FOR CHAT USE CASES |
|---|---|---|
| 7B | **Model architecture:** | **Data collection for helpfulness and safety:** |
| 13B | **Pretraining Tokens:** 2 Trillion | **Supervised fine-tuning:** Over 100,000 |
| 70B | **Context Length:** 4096 | **Human Preferences:** Over 1,000,000 |

# Introduction to Llama2

What, **why** & how Llama2?

- **Llama2 is OPEN SOURCE**
- **Comes in 3 different sizes:** 7b, 13B & 70B parameters
- **Data:** Trained on 2 trillion tokens of text data

Llama 2 was trained on **40% more data** than Llama 1, and has double the context length.

**Llama 2**

| MODEL SIZE (PARAMETERS) | PRETRAINED | FINE-TUNED FOR CHAT USE CASES |
|---|---|---|
| 7B | **Model architecture:** | **Data collection for helpfulness and safety:** |
| 13B | **Pretraining Tokens:** 2 Trillion | **Supervised fine-tuning:** Over 100,000 |
| 70B | **Context Length:** 4096 | **Human Preferences:** Over 1,000,000 |

# Introduction to Llama2

What, **why** & how Llama2?

- **Llama2 is OPEN SOURCE**
- **Comes in 3 different sizes:** 7b, 13B & 70B parameters
- **Data:** Trained on 2 trillion tokens of text data
- **Context Window**: 4096 tokens

Llama 2 was trained on **40% more data** than Llama 1, and has double the context length.

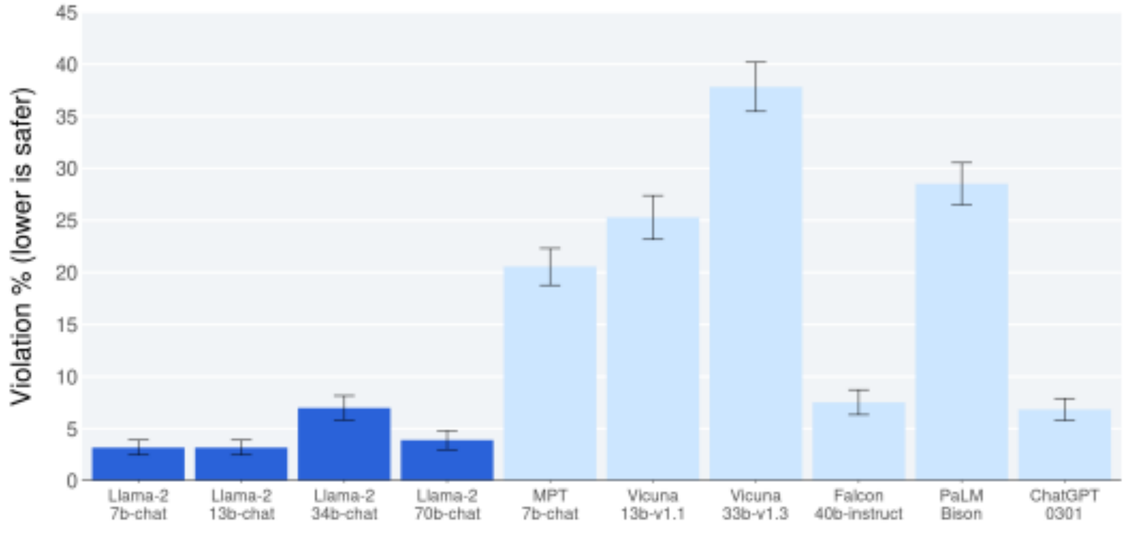
**Llama 2**

| MODEL SIZE (PARAMETERS) | PRETRAINED | FINE-TUNED FOR CHAT USE CASES |
|---|---|---|
| 7B | **Model architecture:** | **Data collection for helpfulness and safety:** |
| 13B | **Pretraining Tokens:** 2 Trillion | **Supervised fine-tuning:** Over 100,000 |
| 70B | **Context Length:** 4096 | **Human Preferences:** Over 1,000,000 |

# Introduction to Llama2

What, **why** & how Llama2?

- **Llama2 is OPEN SOURCE**
- **Comes in 3 different sizes:** 7b, 13B & 70B parameters
- **Data:** Trained on 2 trillion tokens of text data
- **Context Window**: 4096 tokens
- **Safety & Helpfulness**



# Introduction to Llama2

What, **why** & how Llama2?

- **Llama2 is OPEN SOURCE**
- **Comes in 3 different sizes:** 7b, 13B & 70B parameters
- **Data:** Trained on 2 trillion tokens of text data
- **Context Window**: 4096 tokens
- **Safety & Helpfulness**

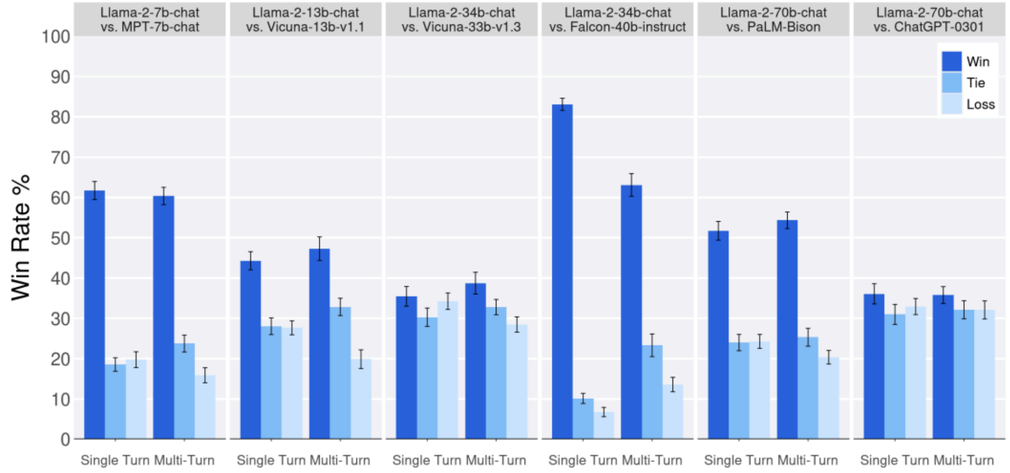
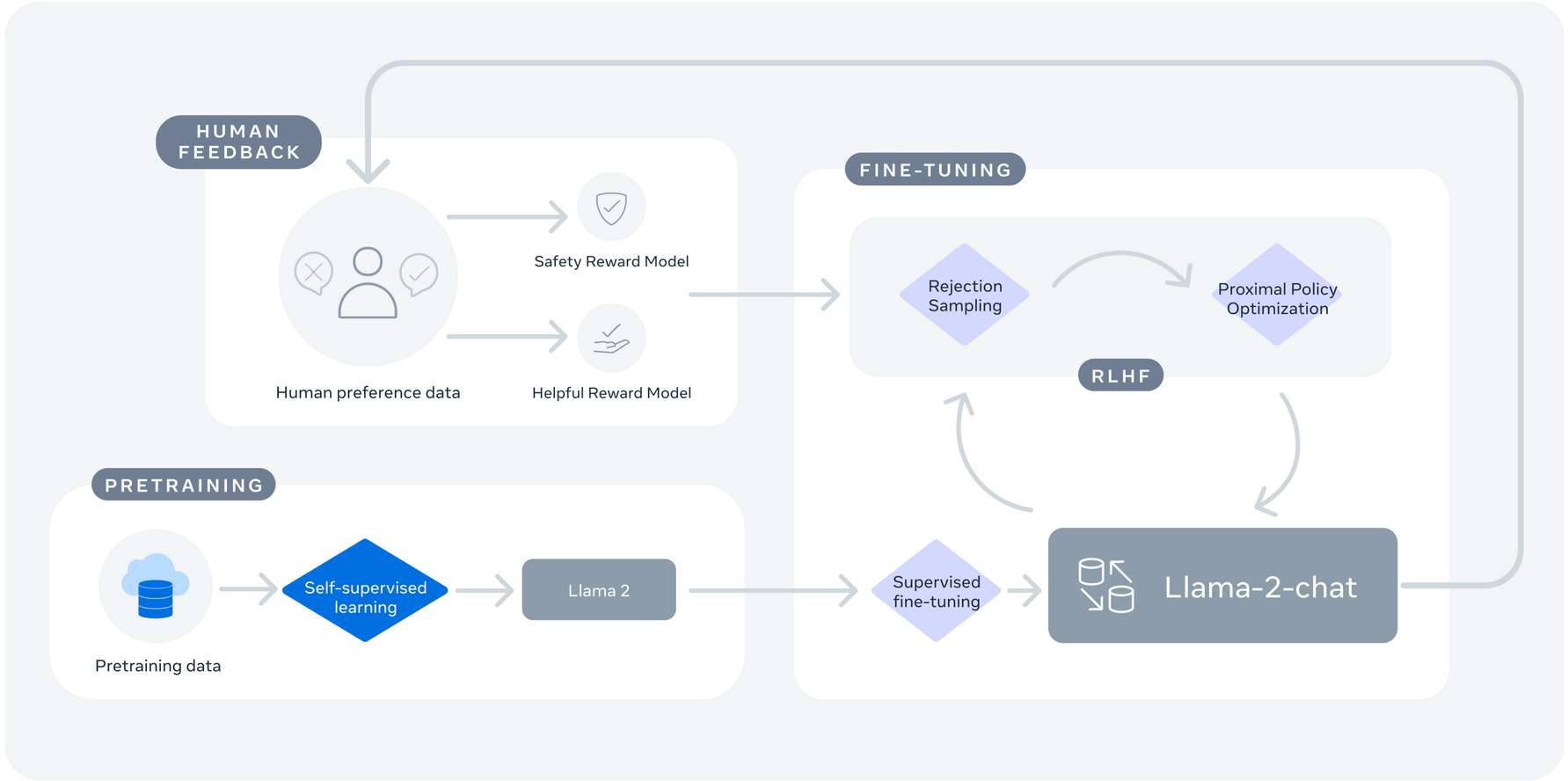



**Figure 12: Human evaluation results** for LLAMA 2-CHAT models compared to open- and closed-source models across ~4,000 helpfulness prompts with three raters per prompt.

# Introduction to Llama2

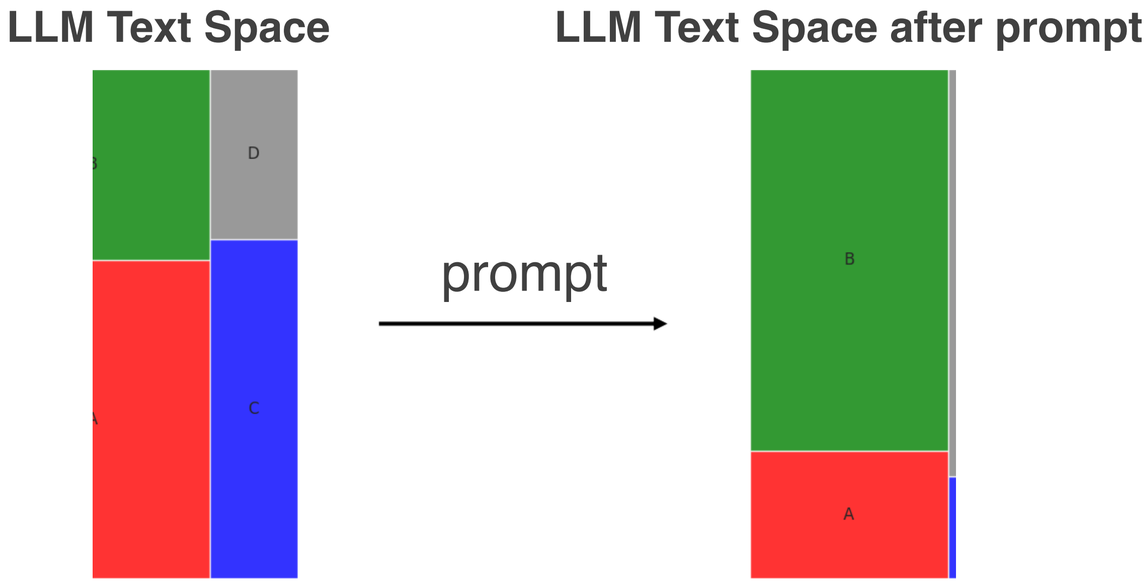
What, why & **how** Llama2?




https://arxiv.org/pdf/2307.09288.pdf

# Prompt Basics

- A prompt is a piece of text that conveys to the LLM the user’s intention.
- Question → Instruction → Behavior
- It constrains the space of possibilities in the LLM's text space



1

# Components of a prompt

Task, input, context, style

- **Task description:** where you describe what you want
- **Input data:** data the model has not seen to illustrate what you need
- **Context information:** background info on what you are requesting, the data you are providing etc
- **Prompt style:** Its about how you ask the thing you want to the model

# Prompt Engineering Guide

## What is prompt engineering?

- **Prompt engineering:** discipline for engineering prompts
- Means by which LLMs can be programmed through prompting.
- Process of creating a prompting function that results in the most effective performance on the downstream task.

The basic goal of prompt engineering is designing appropriate inputs for prompting methods.

# Prompt Engineering Techniques

A simplified guide of prompting techniques

- Zero-shot Prompting
- Few-shot Prompting
- Chain-of-Thought
- Self-Consistency
- Generate Knowledge
- Tree of thoughts (ToT)

# Zero-shot Prompting

- **Zero-shot prompting** is when you solve the task without showing any examples of what a solution might look like

- One can use this as the first try at a model to see what kind of tasks LLM can already solve out of the box

# Zero-shot Prompting

Example

*Classify the sentiment in this sentence as negative or positive:*
***Text**: I will go to a vacation.*
**Sentiment**:

# Few-shot Prompting

Provide information in the form of examples to the LLM

**Few-shot Prompting:** technique where you show a few examples of what a solution might look like.

# Few-shot Prompting

## Example

*A "whatpu" is a small, furry animal native to Tanzania.*
*An example of a sentence that uses the word whatpu is: We were traveling in Africa and we saw these very cute whatpus.*
*To do a "farduddle" means to jump up and down really fast. An example of a sentence that uses the word farduddle is:*

Example taken from (Brown et al. 2020)

# Chain-of-Thought

Induce step-by-step reasoning and planning

**Chain-of-thought (CoT)** enables complex reasoning capabilities through intermediate reasoning steps (Wei et al. 2022).

# Chain-of-Thought

## Example

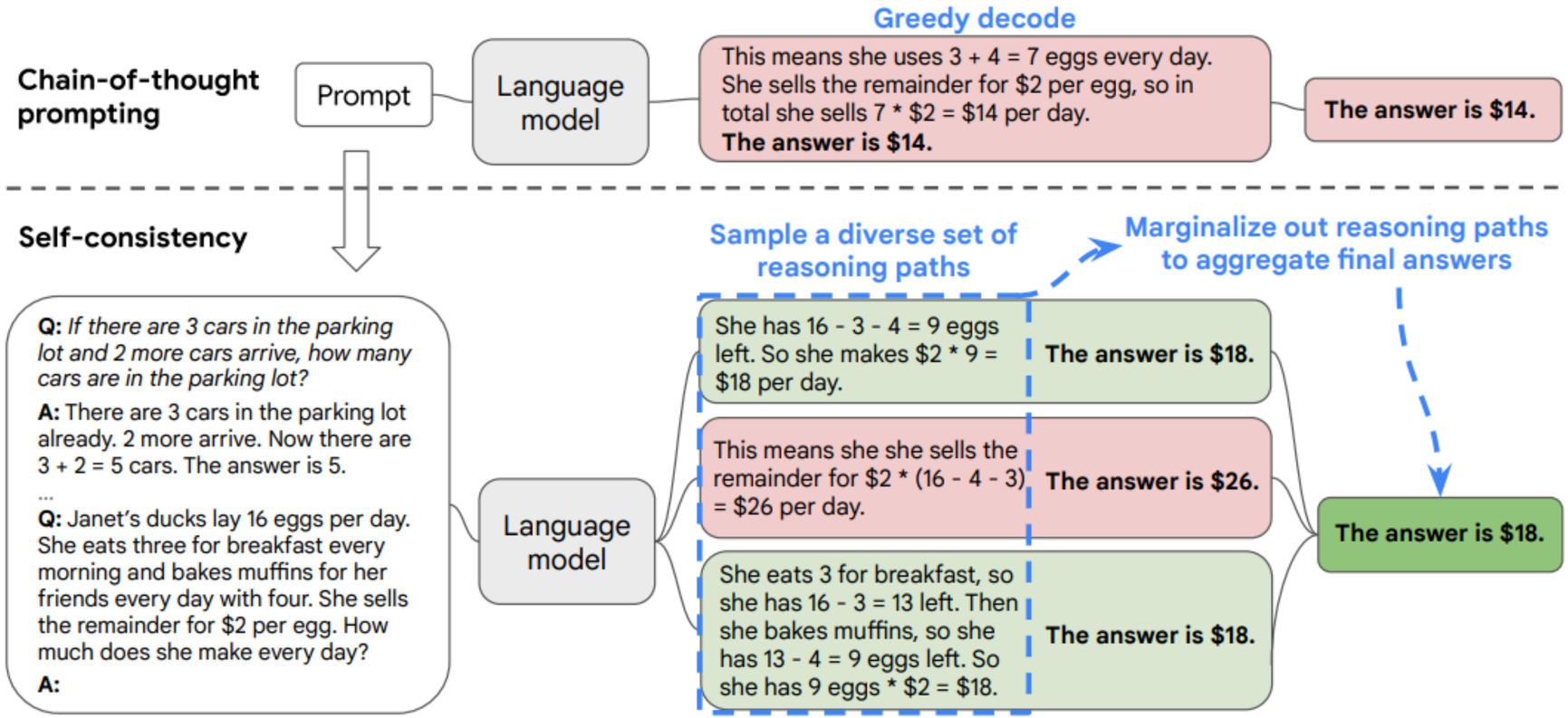
**Q:** I have one sister and one brother. I am 20 years of age. My sister is 5 years older and my brother 2 years younger than my sister. How old is my brother?

**A:** If I am 20 years of age and my sister is 5 years older, my sister is 20+5=25 years old. If my brother is 2 years younger than my sister, my brother is 25-2=23 years old. The answer is 23 years old.

**Q:** I have 2 friends, Jack and Sally. Jack is 2 years older than Sally. Sally is 5 years younger than me. I am 17 years old. How old is Jack?

# Self-Consistency

## Example

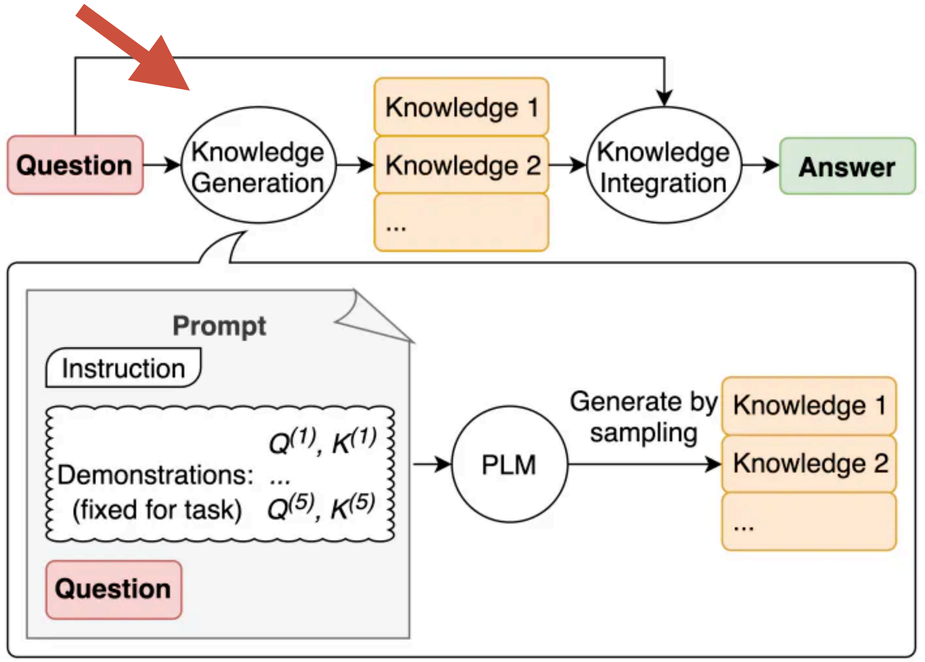



https://arxiv.org/pdf/2203.11171.pdf

# Generate Knowledge

## Example

- **Knowledge Generation**: Generating facts related to the question.

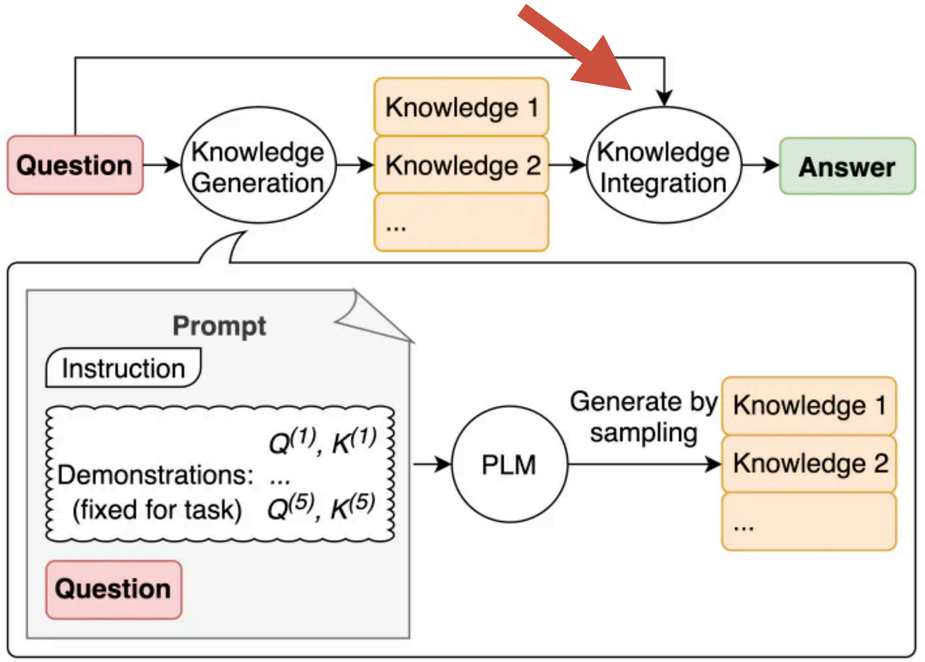



https://arxiv.org/pdf/2110.08387.pdf

# Generate Knowledge

## Example

- **Knowledge Generation**: Generating facts related to the question.
- **Knowledge Integration**: Using generated facts to answer the question comprehensively.




https://arxiv.org/pdf/2110.08387.pdf

**There are many more prompt engineering techniques that grow in complexity, such as:**

- ToT (Yao et al. 2023)
- Retrieval Augmented Generation (Lewis et el. (2021))
- Automatic Prompt Engineer (Zhou et al., (2022))
- React Prompting (Yao et al., 2022)
- Graph Prompting (Liu et al., 2023)
- Skeleton of Thought (Ning et al. 2023)
- Step Back Prompting (Zheng et al. 2023)

# A Framework for Building Good Prompts

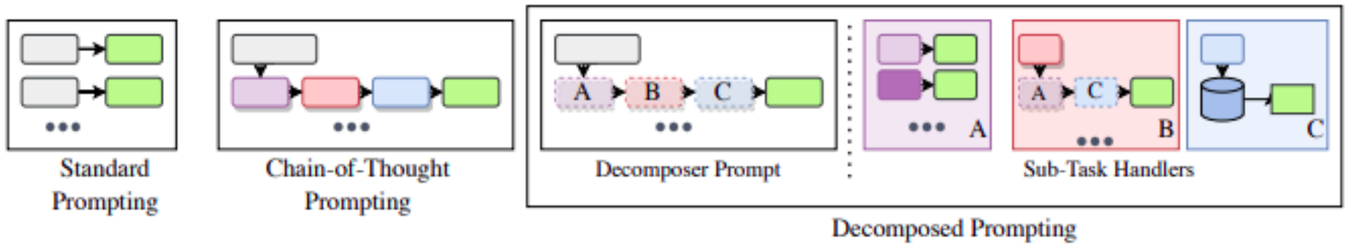
- Operate on Structured Text
- 
- 
- 

```
<python 3 shebang>

<module docstring>

<imports>


<do not include email dunder>

<initialize dotenv>
<set key using OPENAI_API KEY env var>

def complete(prompt: str, **openai_kwargs) -> str:
    <one-line docstring; no params>
    <use default kwargs: model=text-davinci-003, top_p=0.7,
max_tokens=512, 'engine' parameter is deprecated>
    <get completion>
    <strip whitespace before returning>

<as script, demo using prompt "English: Hello\nFrench:">
```

@goodside

# A Framework for Building Good Prompts

- Operate on Structured Text
- Introduce decomposition
- 
- 




QC: Concatenate the first letter of every word in "Jack Ryan" using spaces
Q1: [split] What are the words in "Jack Ryan"?
#1: ["Jack". "Ryan"]
Q2: (foreach) [str_pos] What is the first letter of #1?
#2: ["J", "R"]
Q3: [merge] Concatenate #2 with spaces
#3: "J R"
Q4: [EOQ]

...

decomp

Q: What are the words in "Elon Musk Tesla"?
A: ["Elon", "Musk", "Tesla"]

Q: What are the letters in "C++"?
A: ["C", "+", "+"]
...

split

Q: Concatenate ["n", "i", "e"]
A: "nie"

Q: Concatenate ["n", "i", "c", "e"] using spaces
A: "n i c e"
...

merge

[Khot et al 2023]

# A Framework for Building Good Prompts

- Operate on Structured Text
- Introduce decomposition
- 
- 

**Self-Ask**

GPT-3

Question: Who lived longer, Theodor Haecker or Harry Vaughan Watkins?
Are follow up questions needed here: Yes.
Follow up: How old was Theodor Haecker when he died?
Intermediate answer: Theodor Haecker was 65 years old when he died.
Follow up: How old was Harry Vaughan Watkins when he died?
Intermediate answer: Harry Vaughan Watkins was 69 years old when he died.
So the final answer is: Harry Vaughan Watkins

Question: Who was president of the U.S. when superconductivity was discovered?
Are follow up questions needed here: Yes.
Follow up: When was superconductivity discovered?
Intermediate answer: Superconductivity was discovered in 1911.
Follow up: Who was president of the U.S. in 1911?
Intermediate answer: William Howard Taft.
So the final answer is: William Howard Taft.

Press et al 2023

# A Framework for Building Good Prompts

- Operate on Structured Text
- Introduce decomposition
- 
- 

(d) Zero-shot-CoT (Ours)

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: ***Let's think step by step.***

*(Output) There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls.* ✔

Kojima et al 2023

# A Framework for Building Good Prompts

- Operate on Structured Text
- Introduce decomposition
- Self-criticism
- 

Q: Valerie earns $5000 per month, 1/2 of what her brother earns. If their mother earns twice their combined salary, what's the total amount of money they all have together?

**Zero-Shot**

A:
Valerie's brother earns $10,000 per month, while Valerie herself earns half of that amount. Together, they earn a total of $15,000 per month. Their mother's monthly earnings amount to $30,000, which is twice the combined salary of Valerie and her brother. Thus, the total amount of money they all have together is $30,000. **(Wrong)**

**Few-Shot-CoT**

<Few shot examples>
A:
Valerie earns $5000 per month, half of her brother's earnings, making his monthly income $10000. Combined, Valerie and her brother earn $15000 per month. Their mother earns double their combined salary, amounting to $30000 per month. In total, they have $30000 collectively. **(Wrong)**

**Zero-Shot-CoT**

A: Let's think step by step. First, Valerie's brother earns 2x and Valerie earns x (half of her brother's earnings). Next, their mother's income is twice the combined earnings of Valerie and her brother. In this case, their total income amounts to $30,000. **(Wrong)**

**RCI**

**<Initial output generation>**

A: Valerie's brother earns $10,000 per month, while Valerie herself earns half of that amount. Together, they earn a total of $15,000 per month. Their mother's monthly earnings amount to $30,000, which is twice the combined salary of Valerie and her brother. Thus, the total amount of money they all have together is $30,000.

**<Critique>**

Review your previous answer and find problems with your answer.

All individuals, including Valerie, her brother, and mother, should be taken into account when calculating the overall sum of money.

**<Improve>**

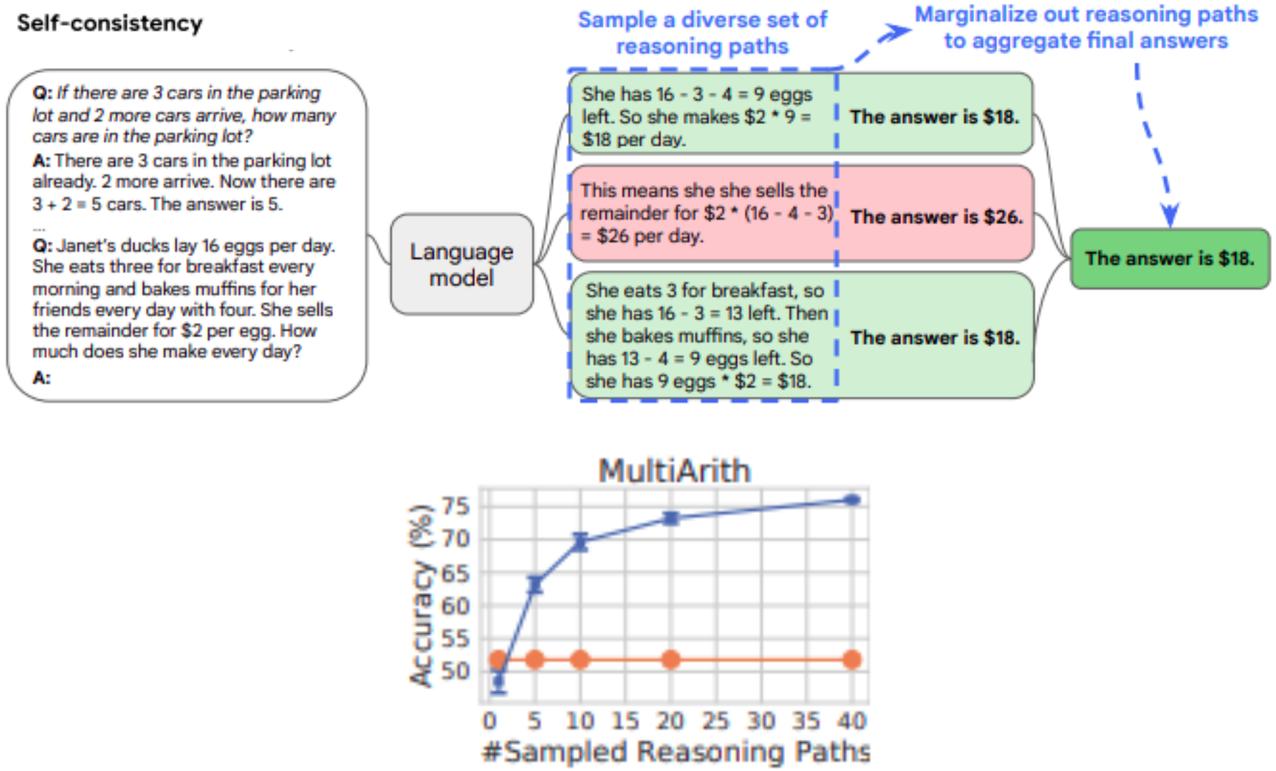
Based on the problems you found, improve your answer.

Valerie's brother earns $10,000 per month, while Valerie herself earns $5,000. Together, they earn a total of $15,000 per month. Their mother earns double their combined salary, which amounts to $30,000 per month. In total, they have $45,000 collectively. **(Correct)**

Kim, Baldi & McAleer 2023

# A Framework for Building Good Prompts

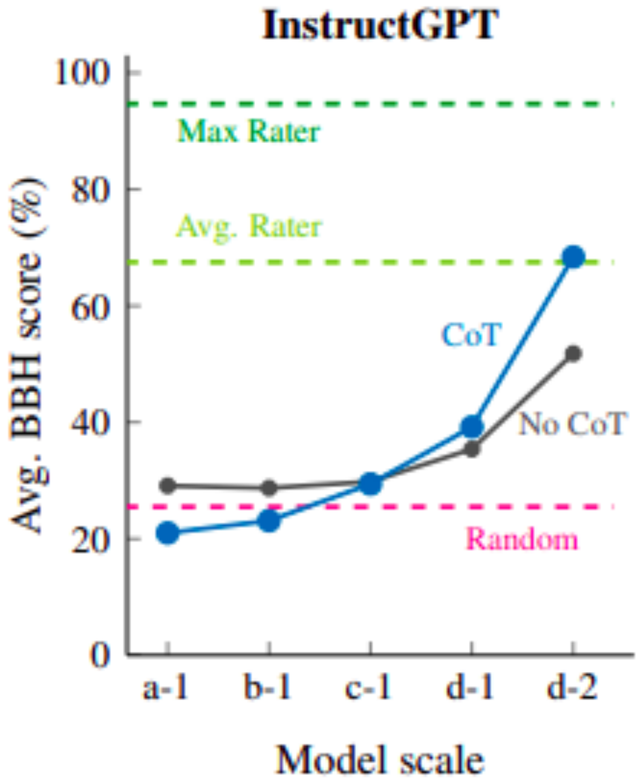
- Operate on Structured Text
- Introduce decomposition
- Self-criticism
- Ensembling




[Wang et al 2023]

# A Framework for Building Good Prompts

- Operate on Structured Text
- Introduce decomposition
- Self-criticism
- Ensembling




Combine for better performance!

[Suzgun et al 2022]

# Exercise / Lab

- Getting started with prompt engineering using Llama2
- Prompt engineering for text summarization and question answering with Llama2
- Extracting dates from unstructured data

# Query Your Docs Locally with Llama2

Private Q&As with docs using Llama2

- Need for LLMs with access to context-relevant data

# Query Your Docs Locally with Llama2

Private Q&As with docs using Llama2

- Need for LLMs with access to context-relevant data
- Privacy concern with closed source LLMs

# Query Your Docs Locally with Llama2

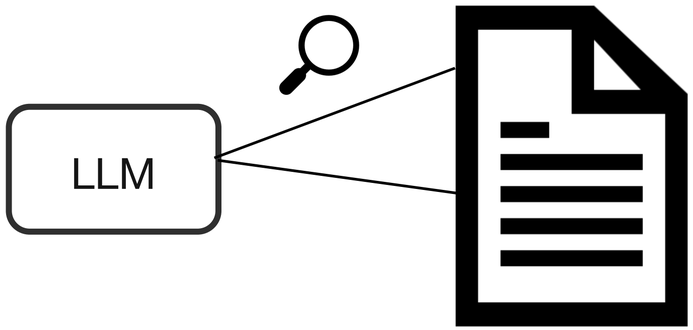
Private Q&As with docs using Llama2

- Need for LLMs with access to context-relevant data
- Privacy concern with closed source LLMs
- Solution? Local LLMs! (Aka **Llama2**!)

# Query Your Docs Locally with Llama2

What & Why RAGs?

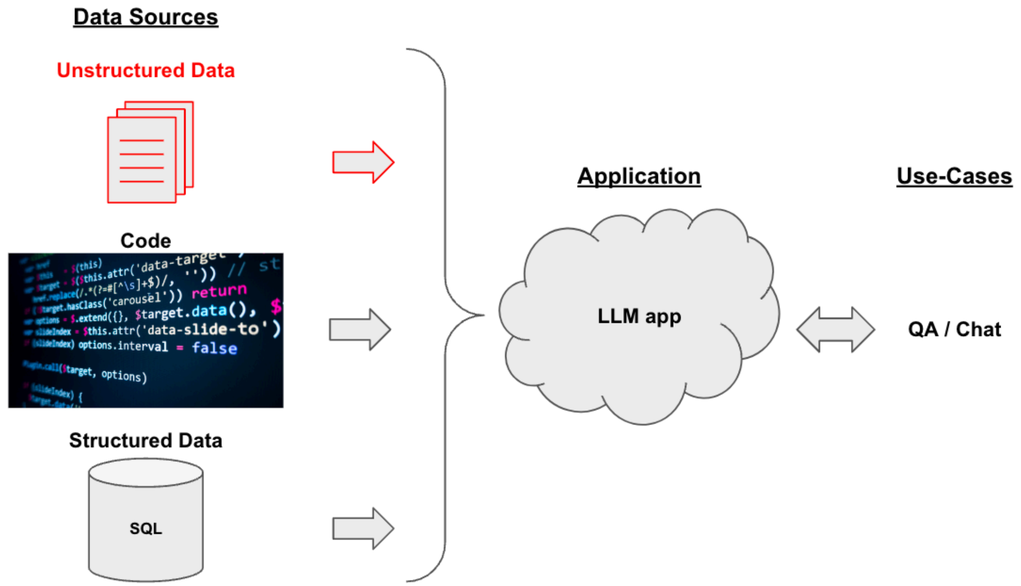
RAG - Retrieval Augmented Generation



# Query Your Docs Locally with Llama2

What & Why RAGs?

RAG - Retrieval Augmented Generation

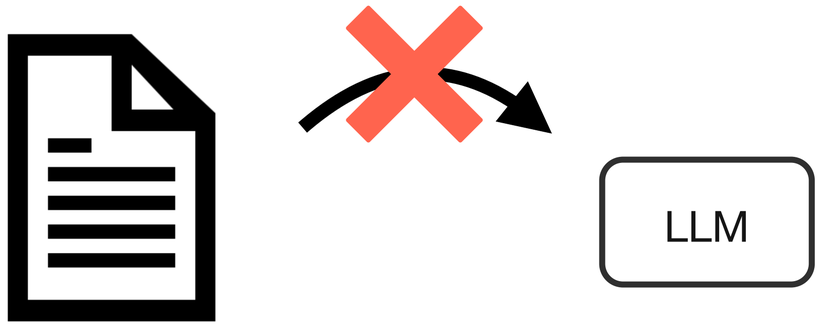



https://python.langchain.com/docs/use_cases/question_answering/

# Query Your Docs Locally with Llama2

What & Why RAGs?

RAG - Retrieval Augmented Generation




https://python.langchain.com/docs/use_cases/question_answering/

# Query Your Docs Locally with Llama2

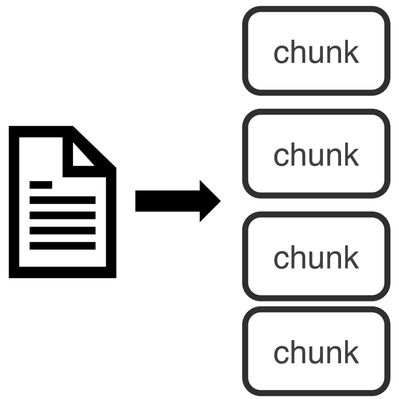
Private Q&As with docs using Llama2

- RAG - Retrieval Augmented Generation
- LLMs have a limited context length

# Query Your Docs Locally with Llama2

Private Q&As with docs using Llama2

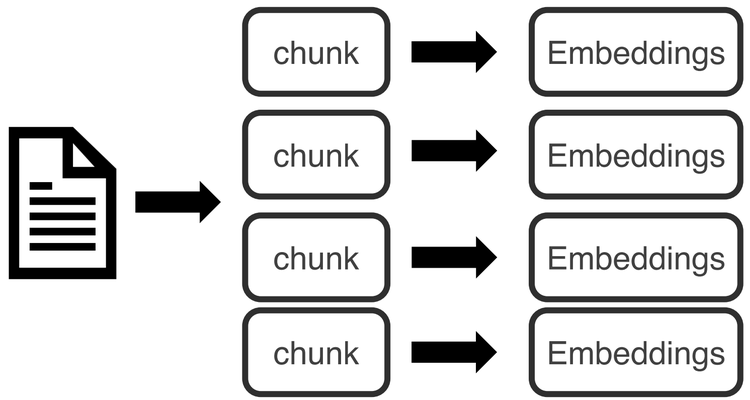
- RAG - Retrieval Augmented Generation
- LLMs have a limited context length



# Query Your Docs Locally with Llama2

Private Q&As with docs using Llama2

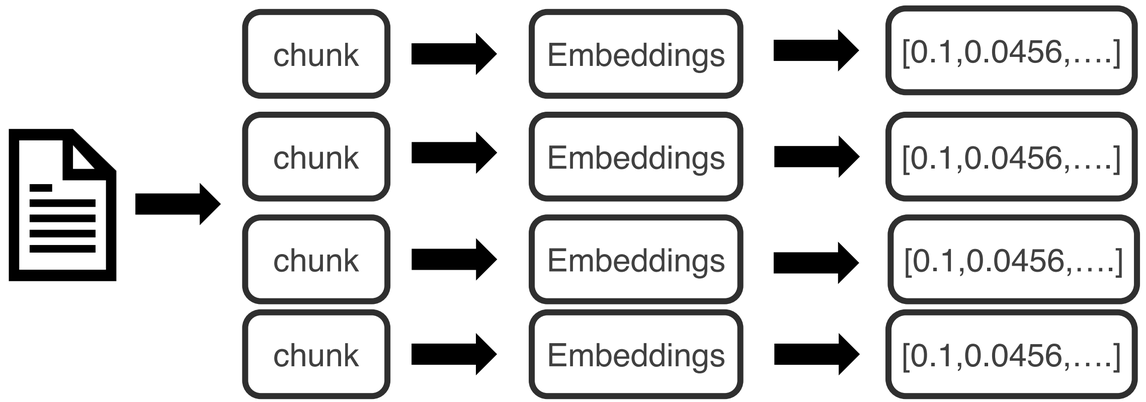
- RAG - Retrieval Augmented Generation
- LLMs have a limited context length



# Query Your Docs Locally with Llama2

Private Q&As with docs using Llama2

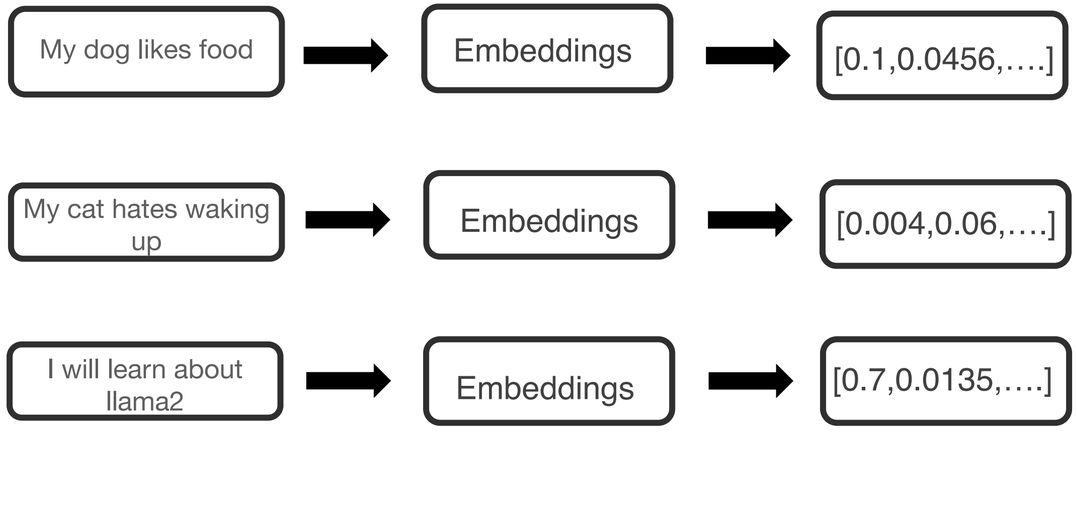
- RAG - Retrieval Augmented Generation
- LLMs have a limited context length



# Query Your Docs Locally with Llama2

Private Q&As with docs using Llama2

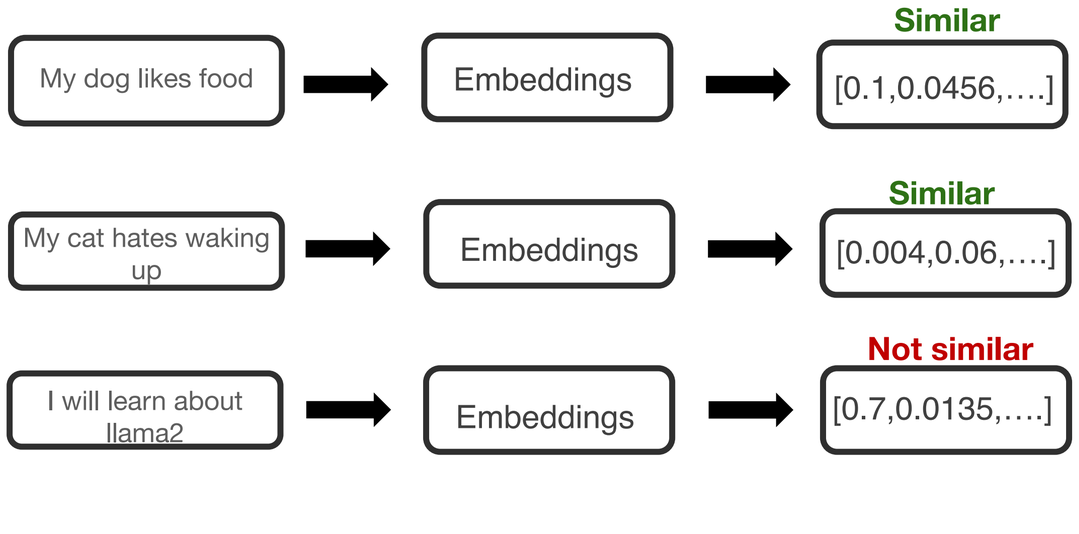
- RAG - Retrieval Augmented Generation
- LLMs have a limited context length
- **Embeddings**: capture content and meaning



# Query Your Docs Locally with Llama2

Private Q&As with docs using Llama2

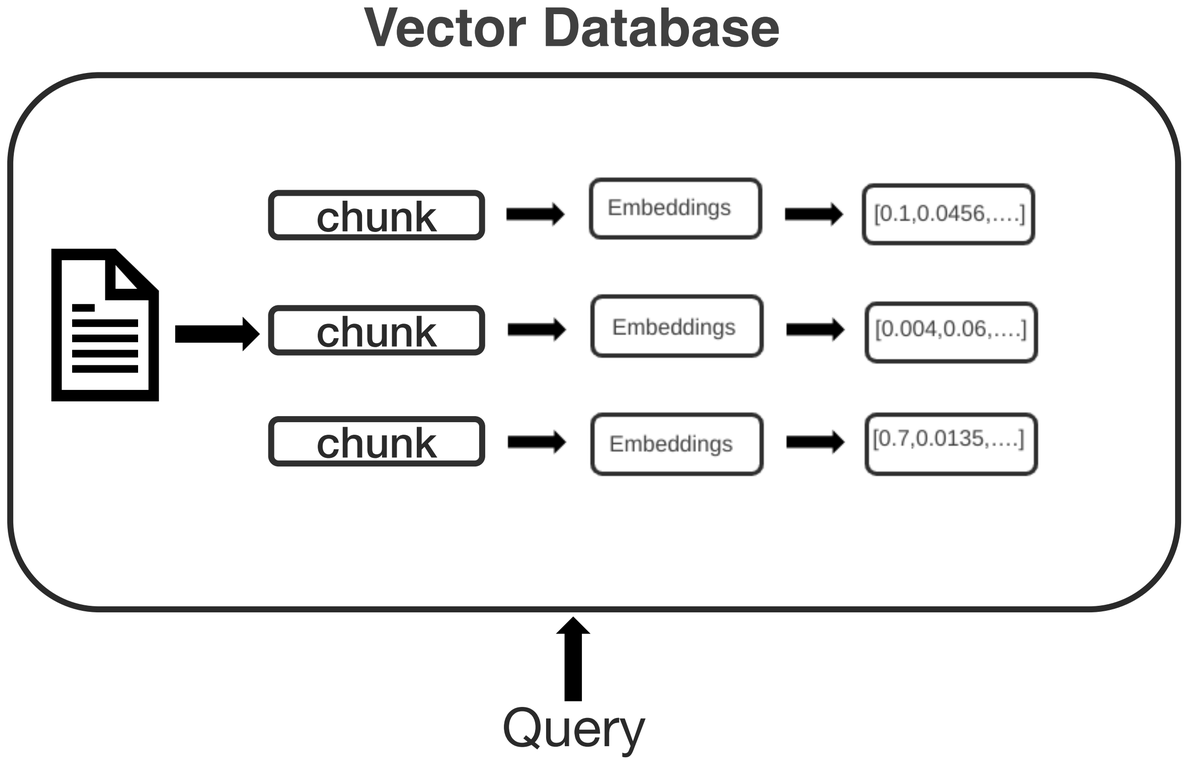
- RAG - Retrieval Augmented Generation
- LLMs have a limited context length
- **Embeddings**: capture content and meaning



# Query Your Docs Locally with Llama2

Private Q&As with docs using Llama2

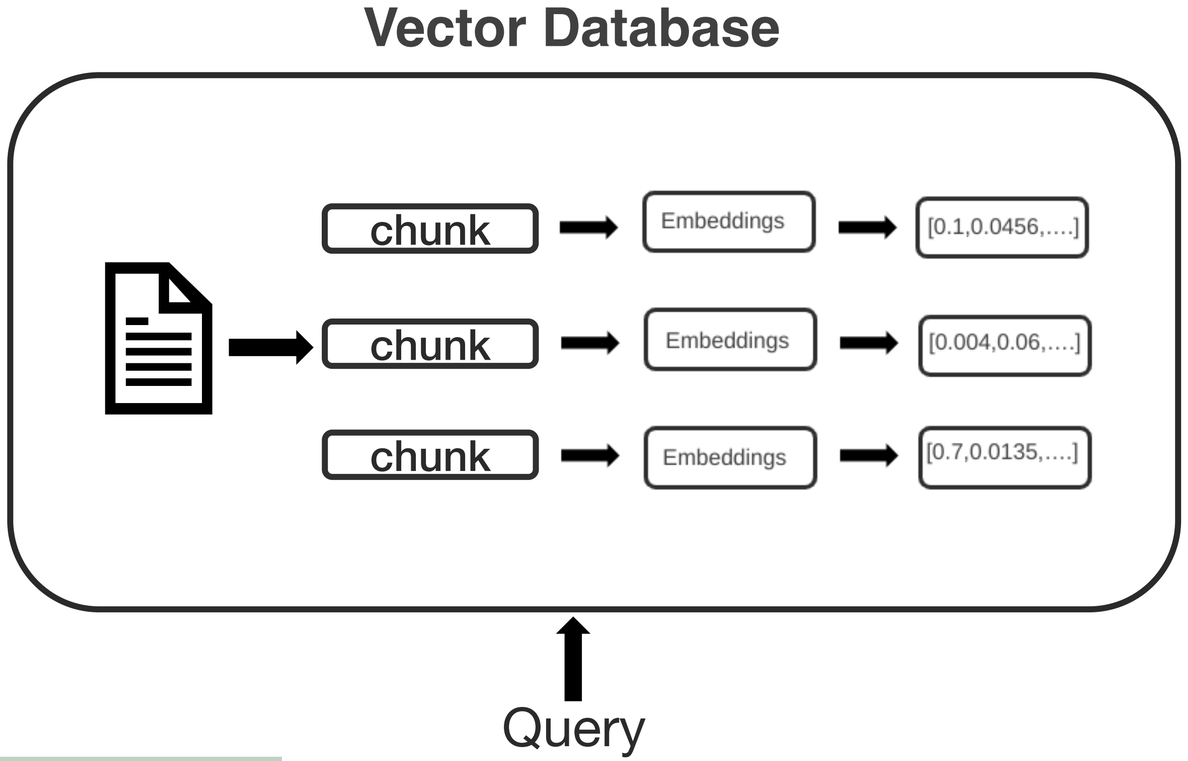
- RAG - Retrieval Augmented Generation
- LLMs have a limited context length
- **Embeddings**: capture content and meaning



# Query Your Docs Locally with Llama2

Private Q&As with docs using Llama2

- RAG - Retrieval Augmented Generation
- LLMs have a limited context length
- **Embeddings**: capture content and meaning

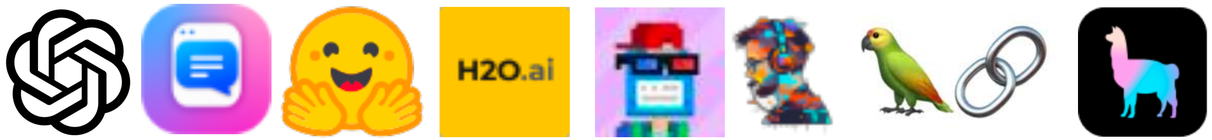



Notebook demo

# Query Your Docs Locally with Llama2

Framework for RAG Systems

Q&A Tech Friction of Access




***Friction of Access***

# Query Your Docs Locally with Llama2

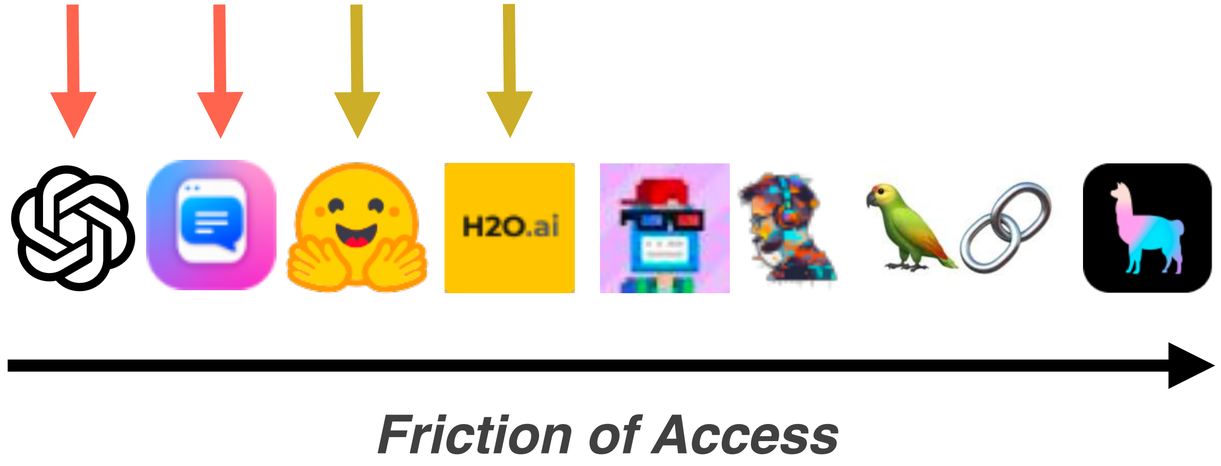
Framework for RAG Systems

- Q&A Tech Friction of Access
- **OpenAI,ChatPDF**

# Query Your Docs Locally with Llama2

Framework for RAG Systems

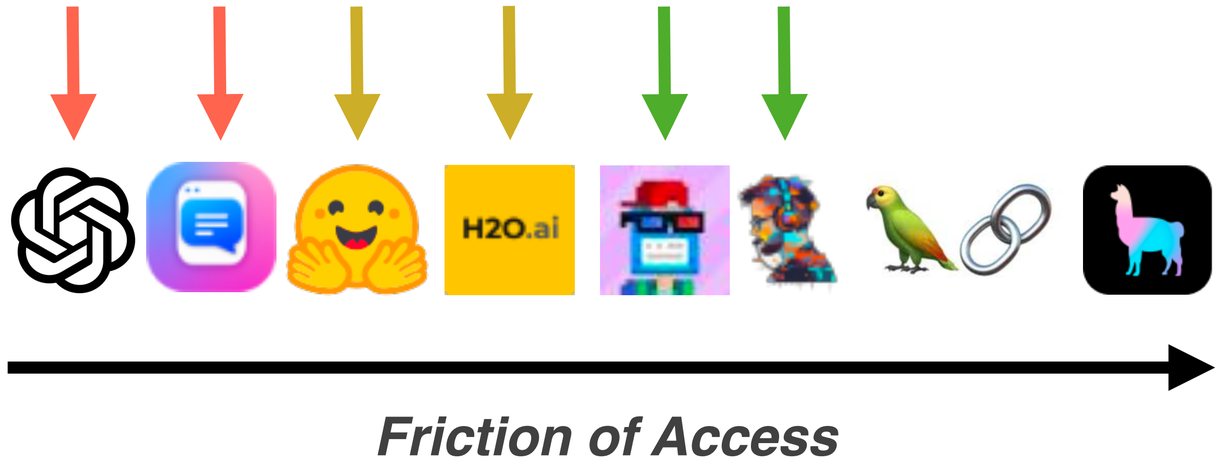
- Q&A Tech Friction of Access
- **OpenAI,ChatPDF**, Hugging Face, h20GPT



# Query Your Docs Locally with Llama2

Framework for RAG Systems

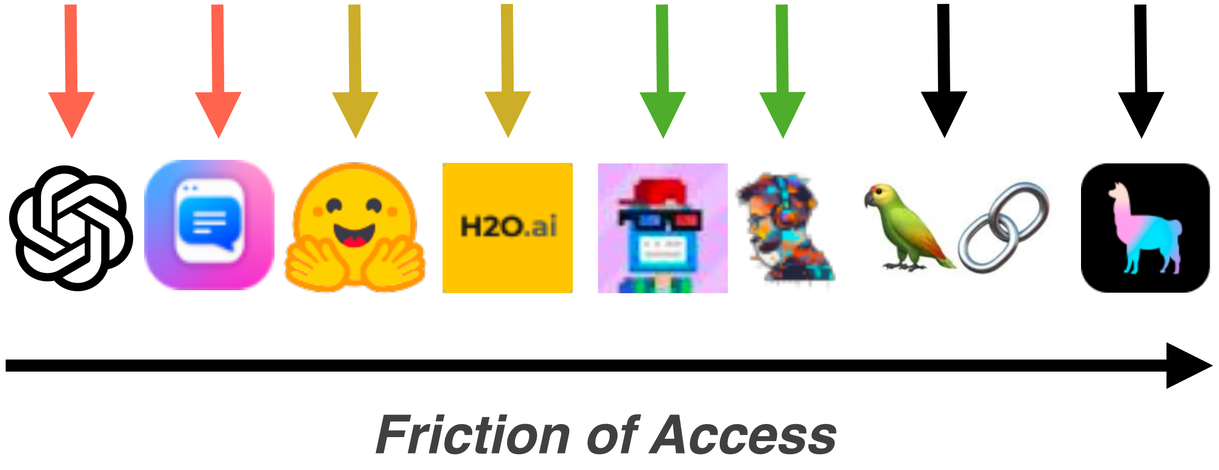
- Q&A Tech Friction of Access
- **OpenAI,ChatPDF**, Hugging Face, h20GPT, PrivateGPT, LocalGPT



# Query Your Docs Locally with Llama2

Framework for RAG Systems

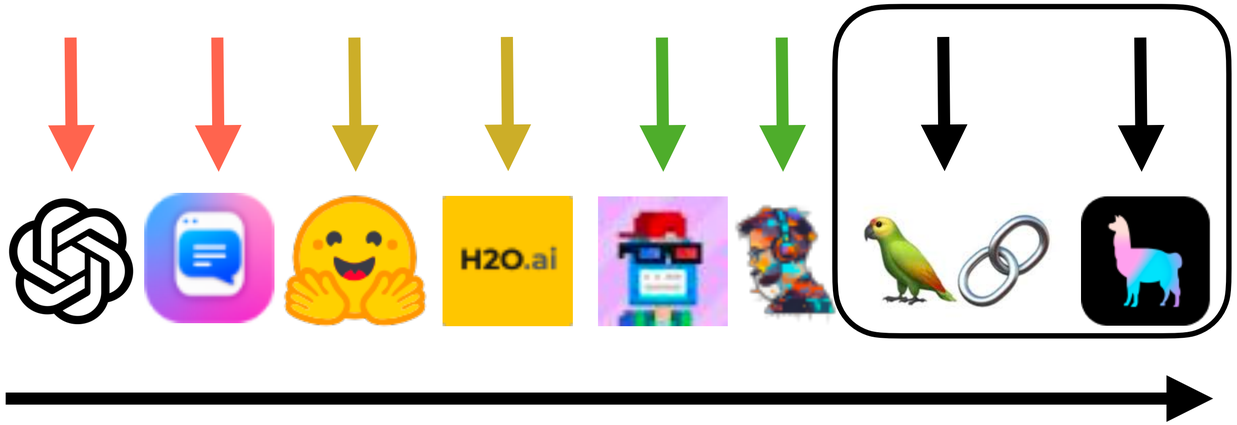
- Q&A Tech Friction of Access
- **OpenAI,ChatPDF**, Hugging Face, h20GPT, PrivateGPT, LocalGPT, **Langchain, Llama-index**



# Query Your Docs Locally with Llama2

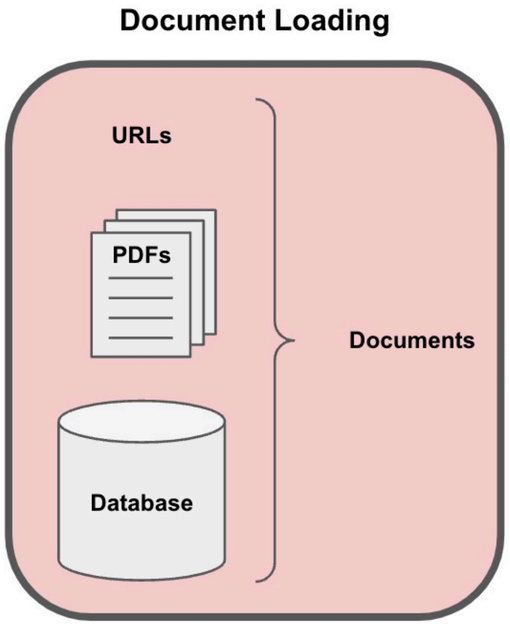
Framework for RAG Systems

- Q&A Tech Friction of Access
- **OpenAI,ChatPDF**, Hugging Face, h20GPT, PrivateGPT, LocalGPT, **Langchain, Llama-index**



# Query Your Docs Locally with Llama2

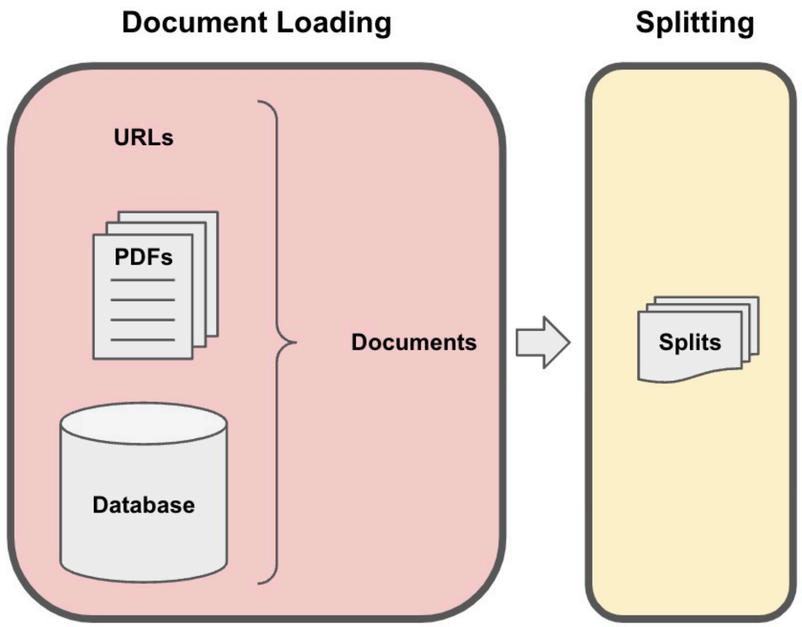
Framework for RAG Systems




https://python.langchain.com/docs/use_cases/question_answering/

# Query Your Docs Locally with Llama2

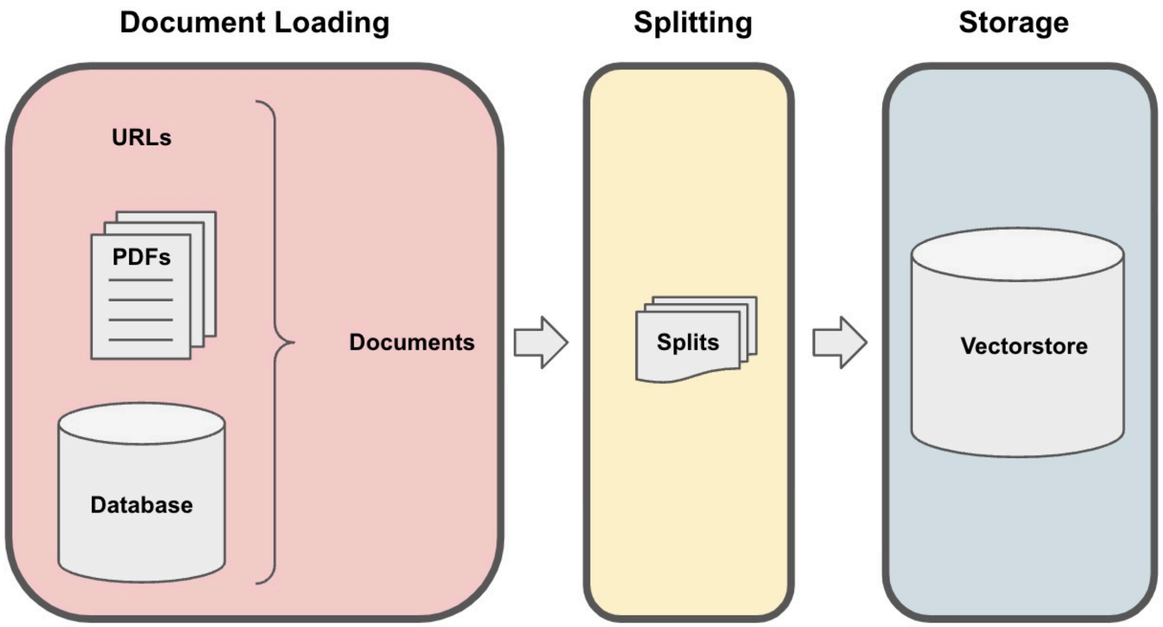
Framework for RAG Systems




https://python.langchain.com/docs/use_cases/question_answering/

# Query Your Docs Locally with Llama2

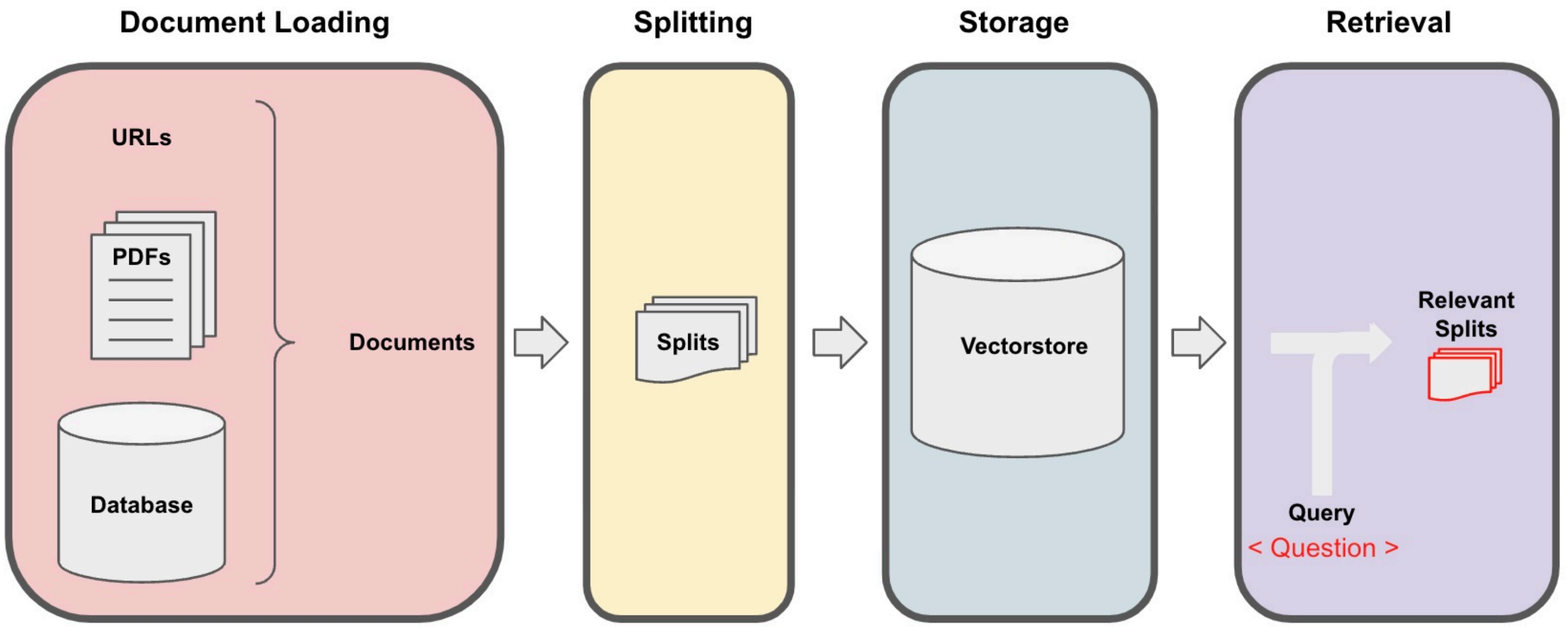
Framework for RAG Systems




https://python.langchain.com/docs/use_cases/question_answering/

# Query Your Docs Locally with Llama2

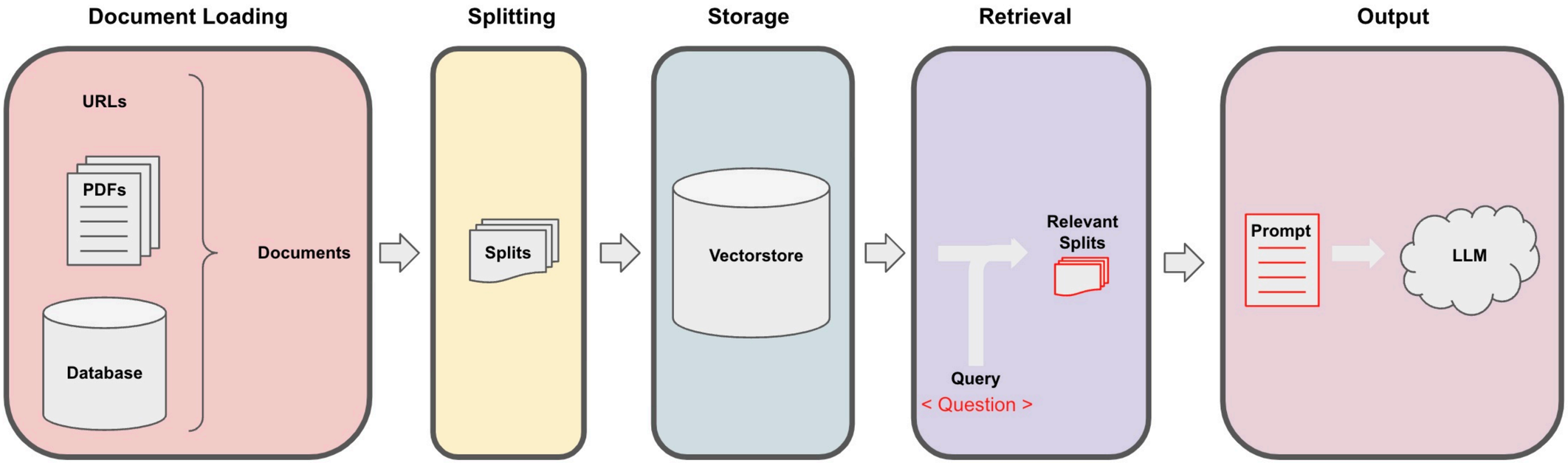
Framework for RAG Systems




https://python.langchain.com/docs/use_cases/question_answering/

# Query Your Docs Locally with Llama2

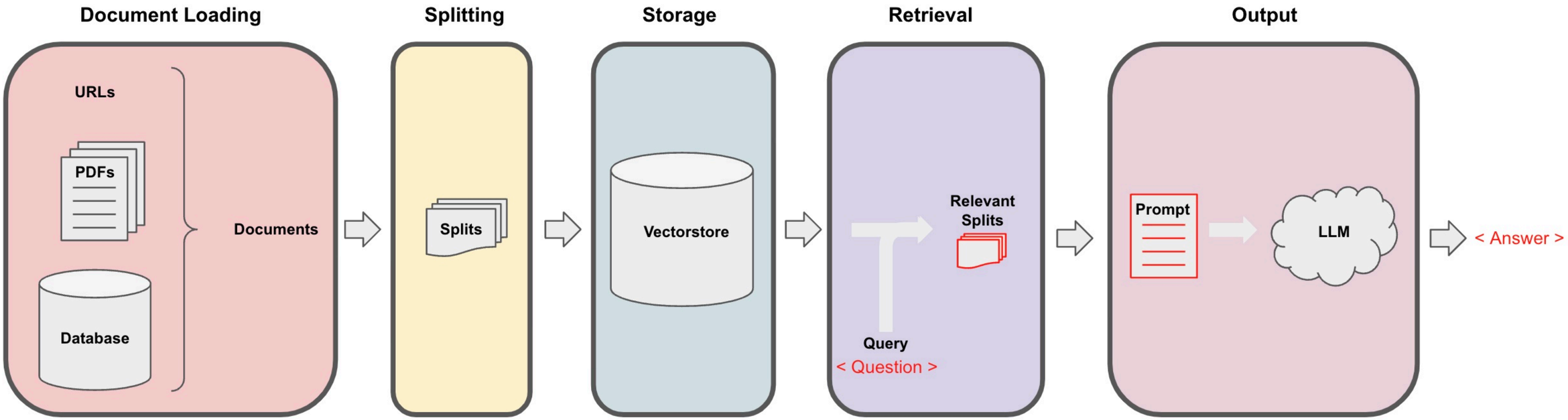
Framework for RAG Systems




https://python.langchain.com/docs/use_cases/question_answering/

# Query Your Docs Locally with Llama2

Framework for RAG Systems




https://python.langchain.com/docs/use_cases/question_answering/

# Query Your Docs Locally with Llama2

Framework for RAG Systems

- Load Document
- Chunk, Split, Process
- Embed
- Vectorise, index
- Query Index
- Process response

# Query Your Docs Locally with Llama2

Framework for RAG Systems

- Load Document
- Chunk, Split, Process
- Embed

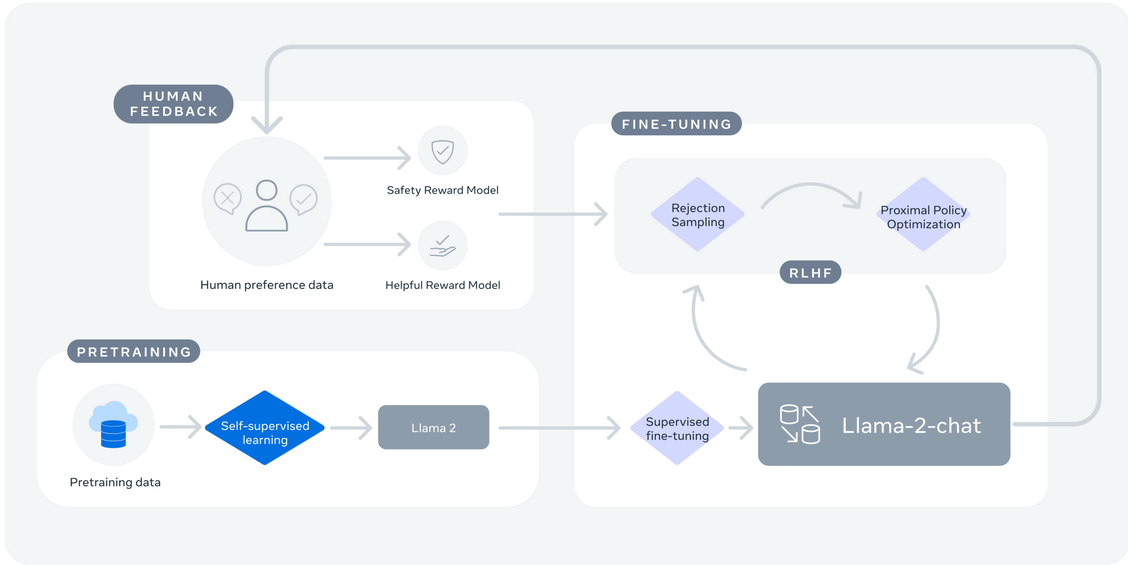
- Vectorise, index
- Query Index
- Process response

Notebook demo

# Fine Tuning Llama2

What, why & how.

## What is Fine Tuning?

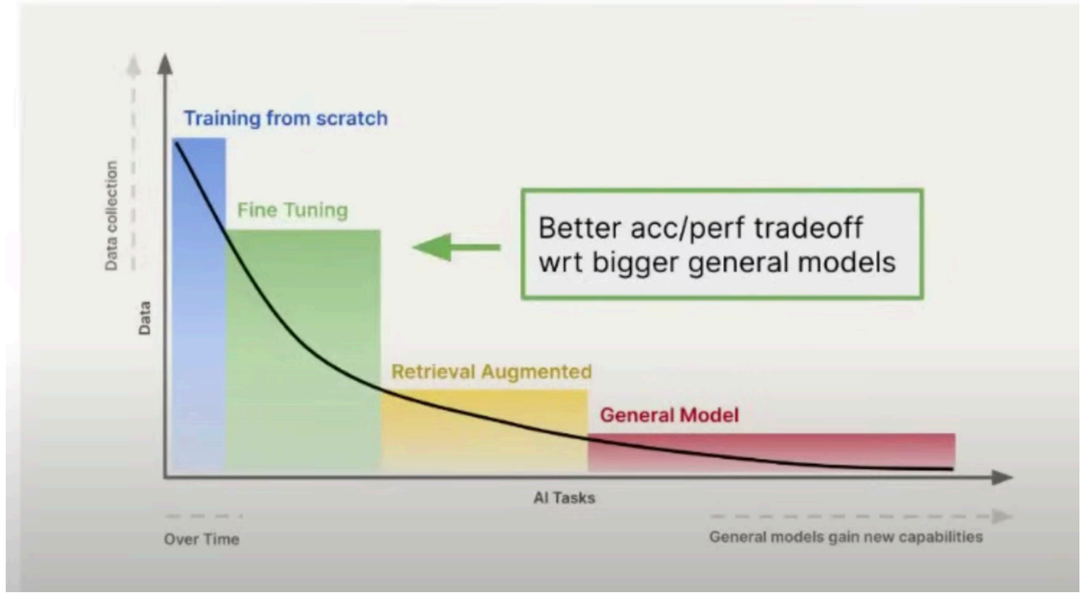



https://arxiv.org/pdf/2307.09288.pdf

# Fine Tuning Llama2

What, why & how.

- What is Fine Tuning?
- Why Fine Tune?




https://www.youtube.com/watch?v=g68qlo9lzf0&t=2935s

# Fine Tuning Llama2

What, why & how.

- What is Fine Tuning?
- Why Fine Tune?
- Memory cost of LLMs: parameters, gradients, optimiser states

## The Memory Bottleneck: GPU comparison

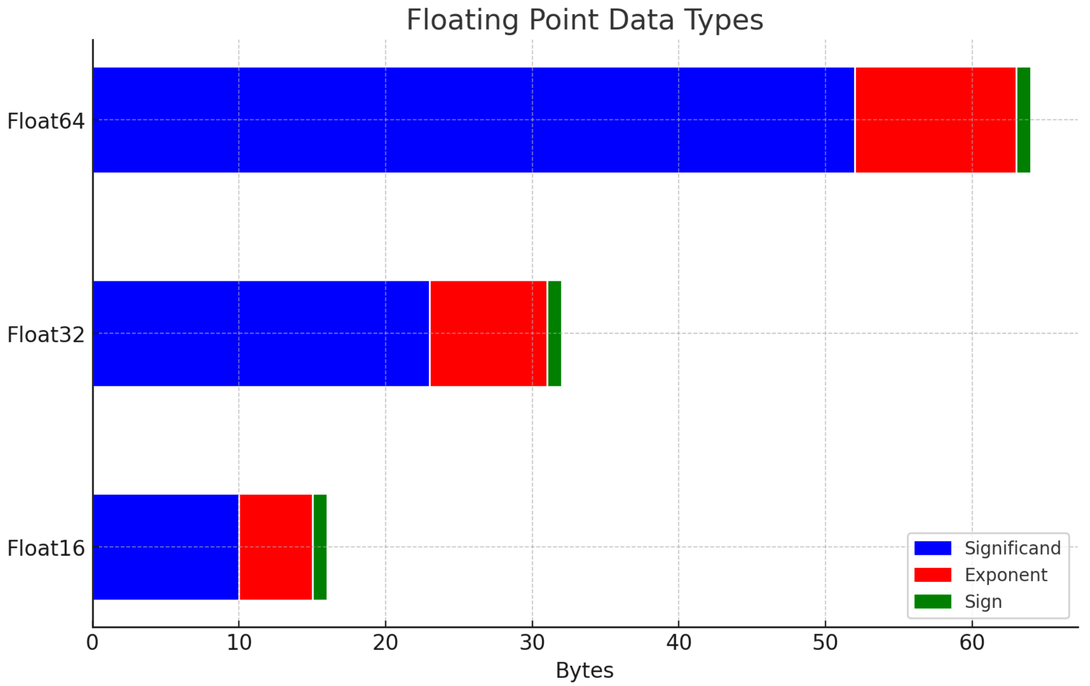
| GPU | Tier | $ / hr (AWS) | VRAM (GiB) |
|---|---|---|---|
| H100 | Enterprise | 12.29 | 80 |
| A100 | Enterprise | 5.12 | 80 |
| V100 | Enterprise | 3.90 | 32 |
| A10G | Enterprise | 2.03 | 24 |
| T4 | Enterprise | 0.98 | 16 |
| RTX 4080 | Consumer | N/A | 16 |

# Fine Tuning Llama2

What, why & how.

Problem - Loading Params

Solution - Half Precision

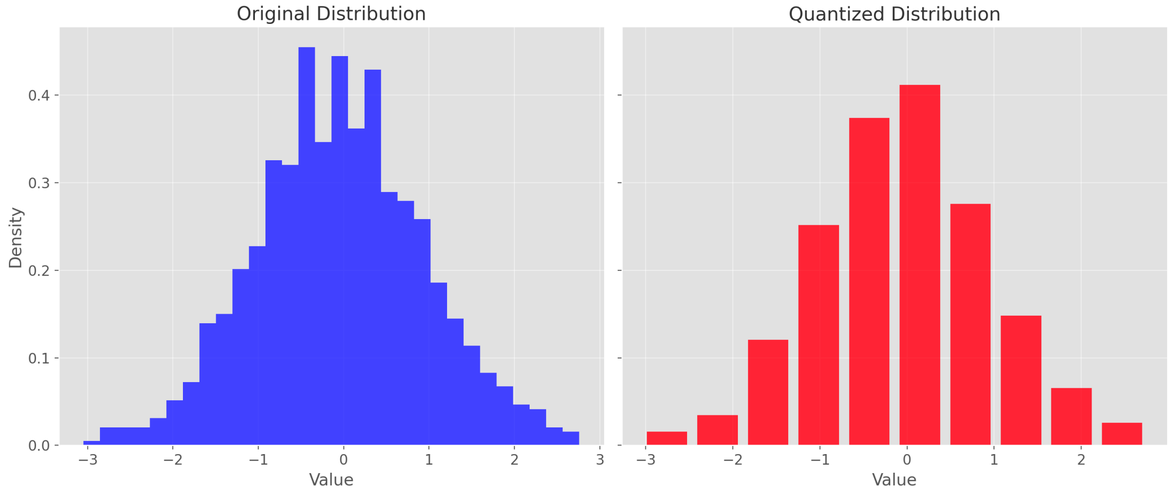



https://www.youtube.com/watch?v=g68qlo9lzf0&t=2935s

# Fine Tuning Llama2

What, why & how.

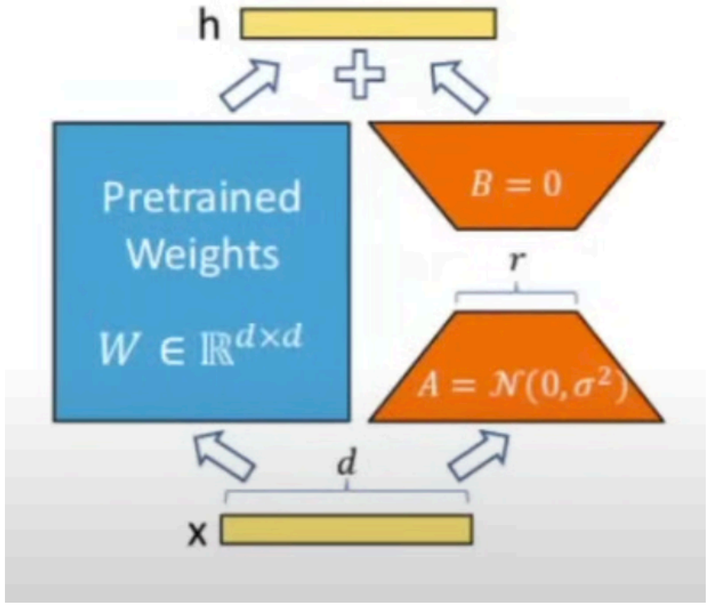
- Problem - Loading Params
  Solution - Half Precision
- Problem - Loading Gradients
  Solution - Quantization



# Fine Tuning Llama2

What, why & how.

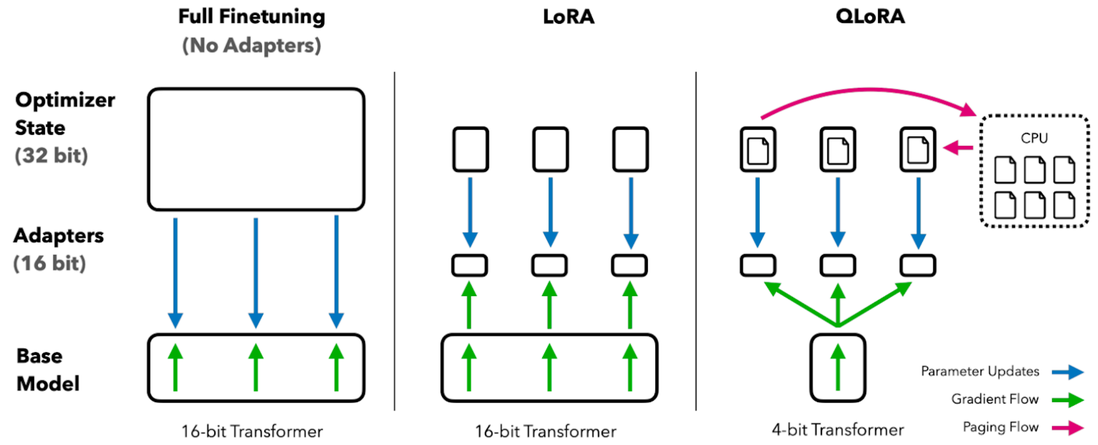
- Problem - Loading Params
  Solution - Half Precision
- Problem - Loading Gradients
  Solution - Quantization
- Problem - Loading Optimizer States
  Solution - LoRA, QLora



# Fine Tuning Llama2

What, why & how.

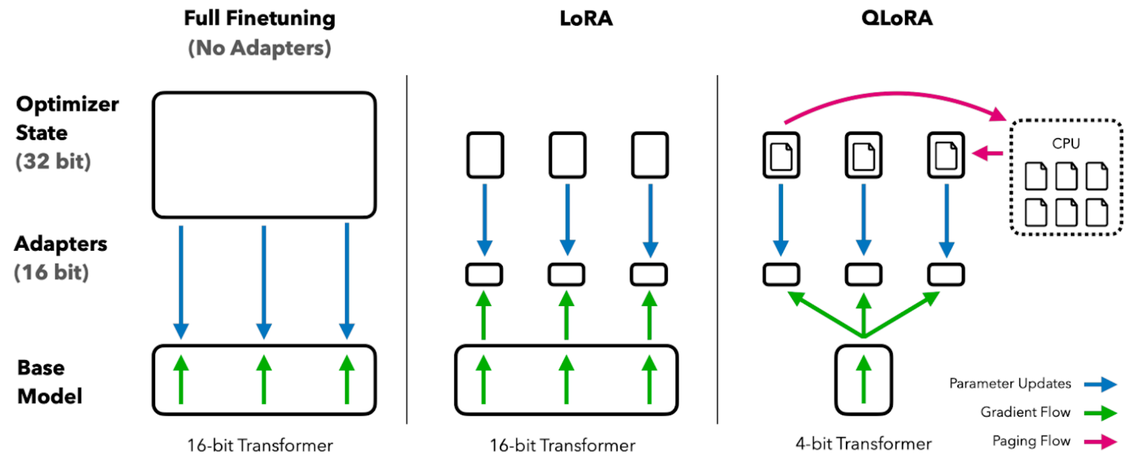
- Problem - Loading Params
  Solution - Half Precision
- Problem - Loading Gradients
  Solution - Quantization
- Problem - Loading Optimizer States
  Solution - LoRA, QLora



# Fine Tuning Llama2

What, why & how.

- Problem - Loading Params
  Solution - Half Precision
- Problem - Loading Gradients
  Solution - Quantization
- Problem - Loading Optimizer States
  Solution - LoRA, QLora




Notebook demo