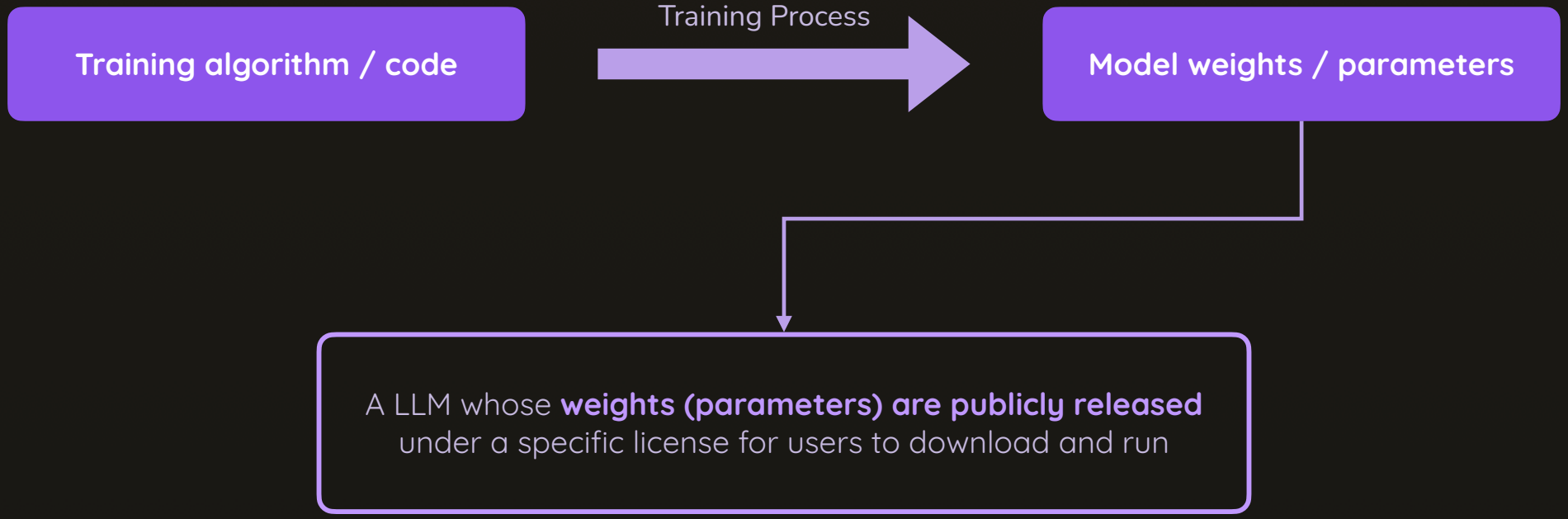
What Are “Open LLMs”?

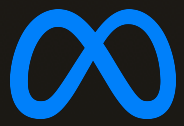
What Are “Open LLMs”?



Open vs Proprietary LLMs

Open LLM	Proprietary LLM
Free to use (must respect license, though)	Paid (Usage-based or Subscription)
Can be run locally or on your servers	Hosted by provider
100% privacy	Privacy depends on provider
No vendor lock-in, full control	Possible vendor lock-in, only little control
Offline-first, low / no latency	Internet connection required
Great for many (local) use-cases like (internal) tools, text summarization, or few-shot prompting	Necessary for use-cases where you need best-in-class performance

Examples for Open LLMs



Meta's Llama Models

Google's Gemma Models

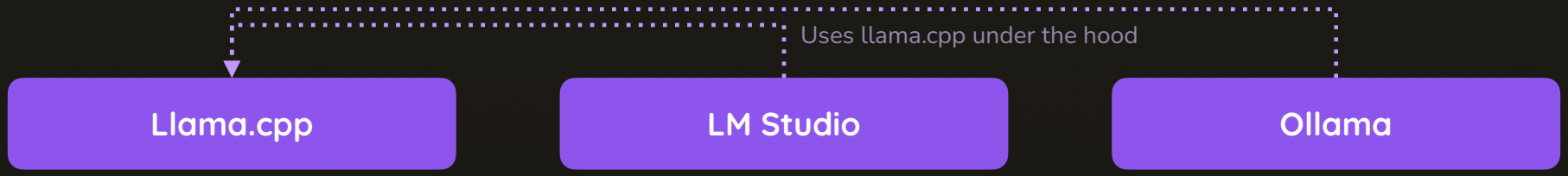
DeepSeek Models

Some Mistral Models

Where Do You Find Open Models?

Open vs Proprietary LLMs

Options For Running Open LLMs Locally



Llama.cpp	LM Studio	Ollama
Very low-level solution	User-friendly graphical UI	User-friendly CLI
Aimed at (experienced) developers, requires technical knowledge	Aimed at “normal users”, no technical expertise required	Aimed at “advanced users”, only limited technical expertise required
Offers CLI and server mode	Offers UI and server mode	Offers CLI and server mode
Plenty of configuration options, allows for building your own “customized version”	Works out of the box but does offer advanced configuration options	Works out of the box but does offer advanced configuration options

Also: **On-demand usage** via **services like Groq**

Important: Licenses!

MIT / Apache 2.0	Llama	Gemma
Very permissive licenses	Somewhat permissive licenses	Somewhat permissive licenses
Allow private & commercial use without limitations	Allows private & commercial use **with** limitations	Allows private & commercial use **with** special requirements
No or only little attribution required	Attribution required	Attribution required
	Prohibits usage for certain use-cases	Prohibits usage for certain use-cases

In general: Check the license & terms of the model you plan to use!

Beyond LLMs & Text Generation

We’ll Run Models Locally

Not remotely

Hardware Requirements & Quantization

Running Large Models On “Small” Machines

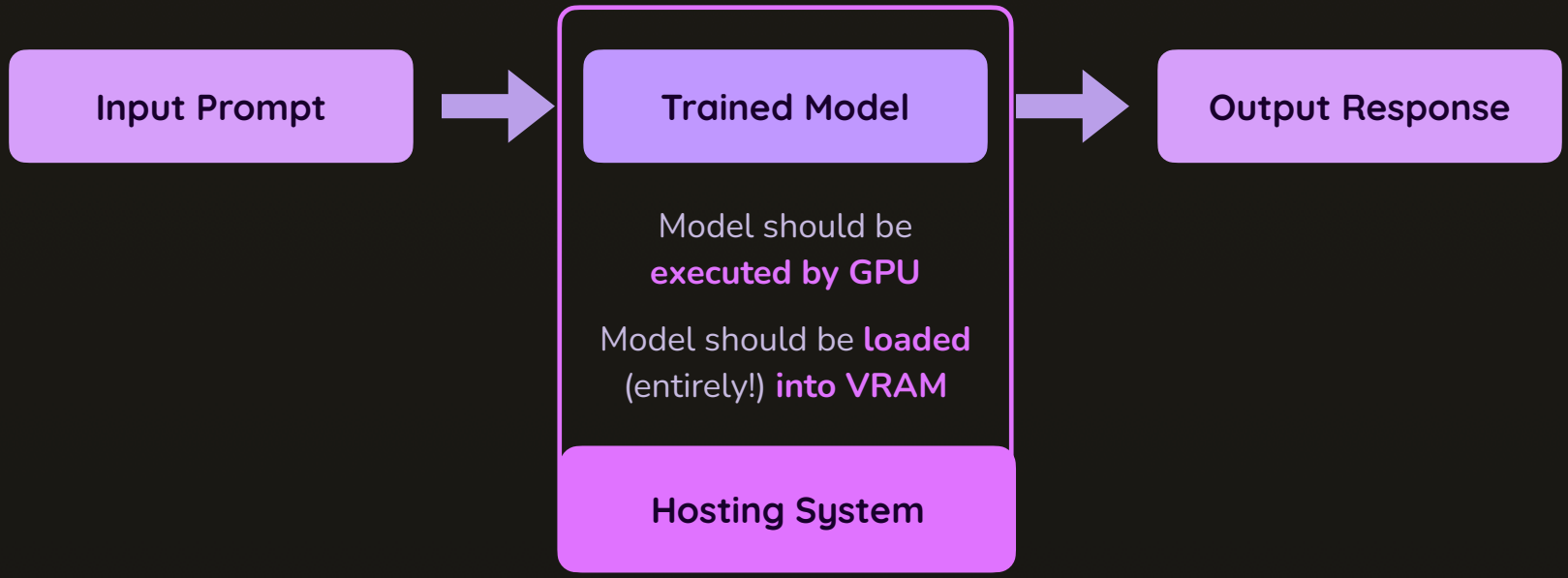
Understanding Model Parameters & Sizes

Hardware Requirements

Exploring Quantization

Running LLMs For Inference

Inference is the process of using a (open or proprietary) LLM to generate output

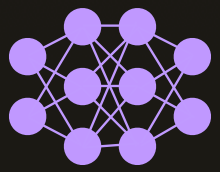


Understanding Model Parameters & Sizes

Input Prompt

Prompt is deconstructed into **tokens (token IDs)**

Trained Model



Every connection has a **weight (parameter) associated** to it

Every parameter is a **float32 or float16 value** (e.g., 0.512913)

Every value takes up 4 or 2 bytes **in memory**

A 2bn model takes up ~4-8 GB of memory

Output Response

Output tokens (token IDs) are **translated to human-readable text**

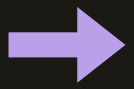
Does It Run On My Machine?

Understanding Quantization

In order to make LLMs **consume less (V)RAM** (and potentially also increase inference speed), the original models are **typically quantized (= compressed)**

Original Trained Model

Not quantized, would take up lots of memory



Quantized Model

Parameter values have been **transformed** from float16/32 to Int4/8

They therefore only take up **0.5-1 byte per parameter**

You don't need to perform this quantization yourself!

Models shared on Huggingface, especially when usable via LM Studio or Ollama, are **available as quantized versions**

LM Studio

A Convenient Chat Interface For Running Open Models Locally

Installing & Using LM Studio

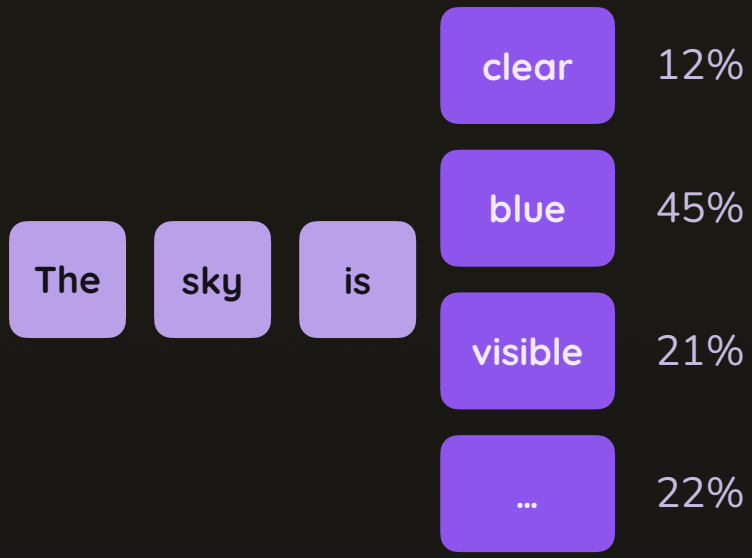
Advanced Configuration Options

Example Usage & Use-cases

Programmatic Usage

Tweaking “temperature”, “top_k”, “top_p” and “min_p”

LLMs generate token candidates



You can configure, which tokens are considered for sampling

Tweaking “temperature”, “top_k”, “top_p” and “min_p”

Temperature

Modifies probabilities

Low temperature:
Exaggerates differences

High temperature:
Flattens differences

top_k

Limits number of candidates

k = 5:
Choose from 5 most likely

k = 1:
Always choose the most likely

top_p

Limits candidates based on combined probabilities

p = 0.5:
All candidates that combined have > 50%

p = 0.9:
All candidates that combined have > 90%

min_p

Discard candidates that don’t meet min-probability threshold

p = 0.05:
Remove candidates with probability < 5%

p = 0.5:
Remove candidates with probability < 50%

Ollama

A Minimal Solution For Advanced Users

Installing & Using Ollama

Advanced Configuration Options

Example Usage & Use-cases

Programmatic Usage

Llama.cpp

No Convenience, Full Control

Understanding llama.cpp

Installing & Using llama.cpp

Example Usage & Use-cases

Programmatic Usage