Instituto de Física de Cantabria

Deploy your LLM

Ignacio Heredia
Instituto de Física de Cantabria (CSIC-UC)

iheredia@ifca.unican.es
IgnacioHeredia

Master Data Science (UIMP-UC)
March 2025

Outline

Instituto de Física de Cantabria

- Motivation
- Docker mini-tutorial
- Deploy a chatbot with ollama
- Deploy a vision model with ollama
- Use ollama from Python
- Build a Retrieval Augmented Generation workflow

Motivation

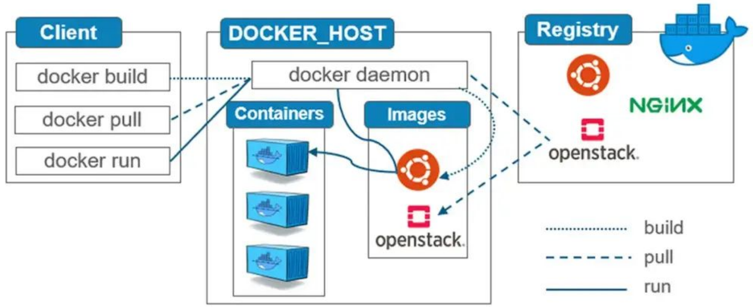
Why deploying models locally?

- 🔍 **privacy**: deploying locally allows you to use LLMs without sending your personal data to a private company that might sell it,
- 👑 **sovereignty**: you won't rely on an external private company.
- 🤑 **cost**: using local models is free!
- ⚡ **speed**: when using models without RAG, small local models reply faster than an large LLM hosted somewhere else
- 🌱 **energy**: using small models (usually) is more energy efficient

The main disadvantage is that small local model are <u>less powerful</u> than big ones. But for simple things, they do work surprisingly well.

Docker - Overview

Docker is an open-source platform that allows developers to build, deploy, run, and manage containerized applications. It uses OS-level virtualization to package software and its dependencies into standardized units called **containers**.



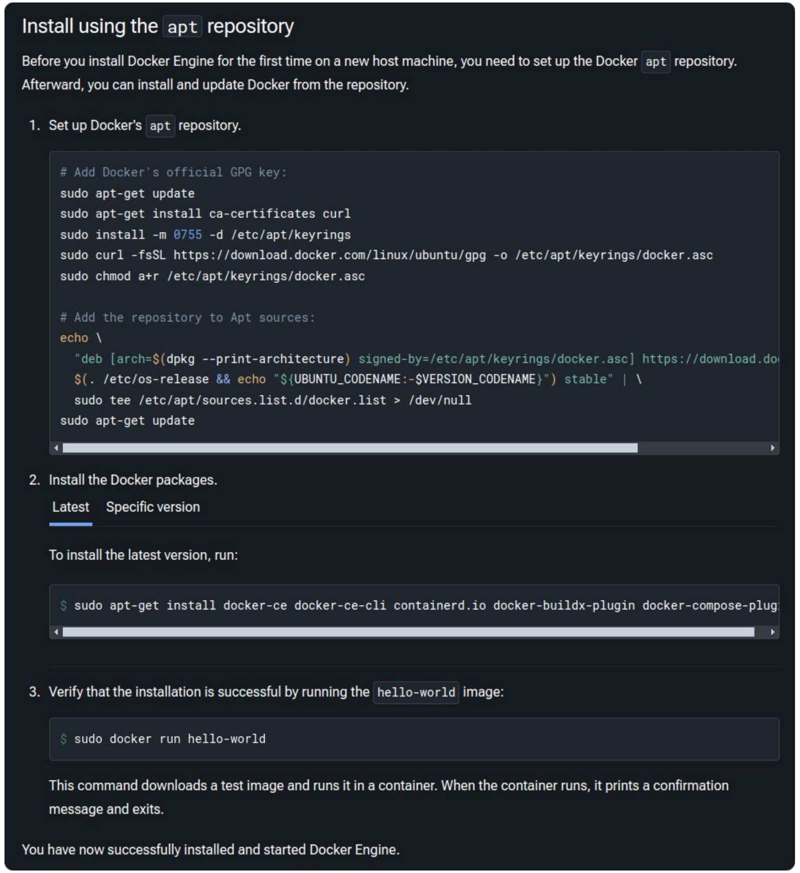
Docker allows developers to package their software and its dependencies in a single unit that can run anywhere.

The most popular registry is DockerHub.

Docker - Installation

- Docker Engine has installation guides for Linux.
- For Windows/Mac user, you can use it through Docker Desktop.
- Installation steps for Ubuntu users:

Go ahead and install it!



Install using the `apt` repository

Before you install Docker Engine for the first time on a new host machine, you need to set up the Docker `apt` repository. Afterward, you can install and update Docker from the repository.

1. Set up Docker's `apt` repository.

```
# Add Docker's official GPG key:
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.do
  $(. /etc/os-release && echo "${UBUNTU_CODENAME:-$VERSION_CODENAME}") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
```

2. Install the Docker packages.

Latest   Specific version

To install the latest version, run:

```
$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plug
```

3. Verify that the installation is successful by running the `hello-world` image:

```
$ sudo docker run hello-world
```

This command downloads a test image and runs it in a container. When the container runs, it prints a confirmation message and exits.

You have now successfully installed and started Docker Engine.

Docker - Basic commands

`docker pull <image>`	pull an image from a registry (eg. Dockerhub)
`docker images`	list your pulled images
`docker run <image>`	create a new container from an image
`docker ps`	list your deployed containers
`docker exec -it <container> <command>`	run a command in a container
`docker rm <container>`	delete a container

Deploy Llama 3.2 with Ollama

- We are going to deploy an LLM locally with ollama
- We can choose any model from the ollama Marketplace.
- You should have at least:
  - 8 GB of RAM available to run the 7B models,
  - 16 GB to run the 13B models,
  - 32 GB to run the 33B models.
- For a better latency experience, we are going to deploy the smaller models.
- Let's deploy Llama 3.2 (1B) from Meta

```
$ docker run -d -v ollama:/root/.ollama -p 11434:11434 --name ollama ollama/ollama
$ docker exec -it ollama /bin/bash
> ollama pull llama3.2:1b
> ollama list
> ollama run llama3.2:1b
> ollama stop llama3.2:1b
```

Deploy DeepSeek-R1 with Ollama

- Let's get fancier and deploy DeepSeek-R1 7B (Qwen distill)

```
> ollama run deepseek-r1:7b
```

- You can see that it is iteratively reflecting over it's own responses

```
>>> You're so cool
<think>
Alright, the user just said "You're so cool." That's pretty straightforward.
I need to respond in a friendly way. Maybe acknowledge their comment.

I should keep it simple and positive without overcomplicating things.
</think>

Thank you! I'm glad you like it. How can I assist you further? 😊
```

(⚠️) People with low resources can use **`deepseek-r1:1.5b`**

Deploy a Vision model

- We can also deploy a small vision model like IBM's granite3.2-vision:2b to test the capabilities to analyse images:

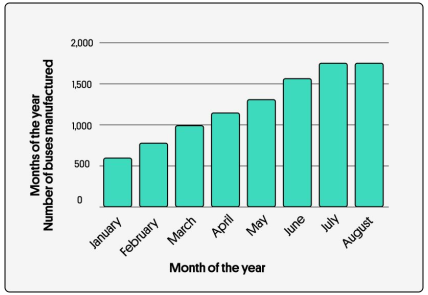
```
$ docker run -d -v ollama:/root/.ollama -p 11434:11434 --name ollama ollama/ollama
$ docker exec -it ollama /bin/bash
> apt update && apt-get install wget
> wget -O demo.jpg <image-url>
> ollama run granite3.2-vision:2b
>>> Describe the image in /demo.jpg
```

⚠️ Take into account that Vision models can be **really slow**, especially when running on large images using CPU only.

Deploy a Vision model

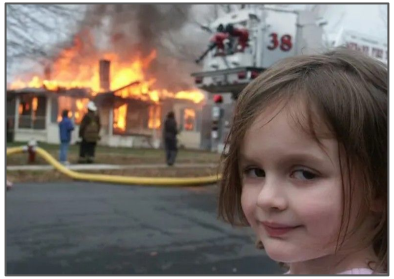
The image depicts a lion walking across a sandy terrain. The lion is positioned towards the left side of the frame, with its body oriented slightly to the right. It has a robust and muscular build, typical of a male lion. Its mane is long and thick, **[...]**

Correctly detects lion, as well as its surroundings



The image is a bar chart that displays data on the "Number of **businesses** manufactured" over different months of the year. The x-axis represents the months of the year, labeled as January, February, March, April, **[...]**

Confuses bus with businesses, rest is pretty fine



The humor in this image stems from the juxtaposition of a young child's innocent **curiosity** with the chaotic backdrop of a fire. The child's expression, which seems to be one of **wonder or mild concern**, contrasts sharply with the **[...]**

It's not able to pick up complex human expression

The model is able to perform (reasonably) well in a wide variety of tasks. We say the models are **zero-shot.**

Using Ollama with Python

- You can use the ollama Python library, to query your model from Python.
- For more advanced workflows, like Retrieval Augmented Generation, you can use the Llama Index library.

```
$ python -m venv --system-site-packages llama-index-env
$ source llama-index-env/bin/activate
$ pip install ollama
$ pip install llama-index
$ pip install llama-index-llms-ollama
$ deactivate
```

Data processing in Python

Instituto de Física de Cantabria

- For example, a simple usecase would be to use an LLM to *fix our data during preprocessing*.
- The processing is easy enough so we can go with a non-thinking model that usually performs faster. In addition, distilled deepseek model are not trained with RL so they can output weird data sometimes.
- Let's deploy Llama 3.2 (3B).

```
$ docker run -d -v ollama:/root/.ollama -p 11434:11434 --name ollama ollama/ollama
$ docker exec -it ollama /bin/bash
> ollama pull llama3.2:3b
> ollama run llama3.2:3b
```

Data processing in Python

- Using the ollama module:

```python
import ollama

faulty_data = "Ths si som slopppy dta tat willl hopefly gt fxed by sart lage languje mdel"
response = ollama.chat(model='llama3.2:3b', messages=[
 {
   "role": "user",
   "content": f"""
The following phrase has mistakes in it, please fix it.
Only reply with the revised text, nothing else:

{faulty_data}
"""
 },])
print(response['message']['content'])
# This is some sloppy data that will hopefully get fixed by smart large language model.
```

RAG with llama-index

- We can also use it to perform Retrieval Augmented Generation in your personal Knowledge Base (for example, create a 📂 **data** folder containing 📄 Guía docente - Machine Learning II):

```python
from pathlib import Path

from llama_index.core import VectorStoreIndex, SimpleDirectoryReader, Settings
from llama_index.embeddings.huggingface import HuggingFaceEmbedding
from llama_index.llms.ollama import Ollama

Settings.embed_model = HuggingFaceEmbedding(model_name="BAAI/bge-base-en-v1.5")
Settings.llm = Ollama(model="llama3.2:1b", request_timeout=360.0)

data_folder = Path(__file__).parent / "data"
documents = SimpleDirectoryReader(data_folder).load_data()
index = VectorStoreIndex.from_documents(documents)
query_engine = index.as_query_engine()

query = "Who are the lecturers in Machine Learning?"
response = query_engine.query(query)
print(response)
```

RAG with llama-index

- We can also use it to perform Retrieval Augmented Generation in your personal Knowledge Base (for example, create a 📂 **data** folder containing 📄 Guía docente - Machine Learning II):

```
The lecturers listed for the Machine Learning course 249 - AUTOMATIC LEARNING II
at the University Master's Degree in Data Science are:

1. Luis Ignacio Santamaría Caballero
2. Steven Johan Maria van Vaerenberg
3. Soto Herrera Garcia
4. Javiera Diez Sierra
5. MaiaLEN Iturbide Martinez De Albenez
```

```
(*) Correct answers

LUIS IGNACIO SANTAMARIA CABALLERO
STEVEN JOHAN MARIA VAN VAERENBERGH
SIXTO HERRERA GARCIA
JAVIER DIEZ SIERRA
MAIALEN ITURBIDE MARTINEZ DE ALBENIZ
```

- Errors could come both from PDF parsing and from model size.
- You could also tweak the previous script to change the temperature (ie. make the LLM more creative), to see the citations the model uses as reference, etc.

Conclusions

- LLMs are very easy to deploy locally, thus benefiting from additional privacy,
- Response latency for smaller models is pretty good, at least for pure-text models,
- Using +7B models locally tends to consume too many resources in standard workstations,
- Use cases:
  - Chat works pretty well, allowing to show the thought process
  - Vision models work fine in simple usecases,
  - Effective for “simple” data wrangling that would otherwise be difficult with standard Python modules,
  - Using RAG delivers mixed results depending on the database size,

Questions

Ignacio Heredia
Instituto de Física de Cantabria (CSIC-UC)

iheredia@ifca.unican.es

IgnacioHeredia

Image sources