Lab 7. Intro to Ollama

Feb 26, 2026

Outline

Outline

Introduction

- This session provides a hands-on introduction to **Ollama**, which helps us running LLMs locally.

What is Ollama?

- Running LLMs locally can be challenging — setup, dependencies, and GPU configurations can be complex.
- *Ollama* simplifies local development with open-source LLMs.
- Think of it as Docker for LLMs — each model (with weights + config) is packaged into a single Model file.
- You can easily pull, run, and customize models locally.

Step 0: Set Up a Python Virtual Environment

- Before writing code with the `ollama` Python library, it's good practice to work inside a virtual environment.
- This keeps dependencies isolated from your global Python installation.

Step 1: Install Ollama

- Ollama supports macOS, Windows, and Linux.
- Download from the official site:
  `https://ollama.com/download`
- Installation auto-detects your GPU drivers (NVIDIA/AMD).
- CPU mode works fine but is slower.

Step 2: Install a model

- Visit the model library at ollama.com/library. Each model family has multiple sizes and variants - so please take a look first!
- For example, if you choose gemma2 $\rightarrow$ `ollama run gemma:2b`
- The model (1.7B parameters) will download and cache locally.
- If you enter ollama list in the terminal, you can see which models are installed.

Step 3: Run the model

There are several functions:

- 1. Chat with the model.

```
>>> Why is the sky blue?
The sky appears blue because of light scattering...
>>> /bye (if you want to exit)
```

- The model will respond interactively
- Try a few simple questions!

Step 3: Run the model

There are several functions:

- 2. Multiline input
- If you want to enter a long message, enclose it with triple quotation marks (""").

```
>>> """
Hello!
Hope you have a great day!
"""
```

Step 3: Run the model

There are several functions:

- 3. Some multimodal models also support image input.

```
>>> What's in this image? /path/to/sunflower.png
The image shows a sunflower...
```

Step 4: Use Ollama with Python

- Ollama can also be used in applications (e.g., Python library)
- First, install the Python package:

```
$ pip install ollama
```

Example: Python script

```
[language=Python]
import ollama

response = ollama.generate(
    model='gemma:2b',
    prompt='What is a qubit?'
)
print(response['response'])
```

- The output will contain a locally generated LLM response.
- You can use this setup to build custom apps and chat interfaces.
- Please check here: `https://ollama.com/blog/python-javascript-libraries`

Outline

Assignment: Run and customize a local LLM

- Goal: Use `Ollama` to run a local LLM and explore how system prompts affect model behavior.
- You will write a short Python script that:
  - Generates text from a locally run LLM
  - Demonstrates a simple, clearly defined use case
  - Prints and saves the model's output

Step 1: Choose a model

- Select any open model available through Ollama.
- In approximately 100 words, explain why you chose this model.
- Your explanation should demonstrate understanding of the model's size, strengths, or intended use.

Step 2: Write your script

- Define a clear use case. For example:
  - Summarize a paragraph
  - Generate a short creative story
  - Translate a sentence
  - Explain a concept in simple terms
  - Extract keywords from text

Step 2: Write your script

- Your script must:
  - Load the selected model
  - Include a system prompt that defines tone or behavior
  - Generate text based on your task
  - Save the output to `output.txt`
- You may generate multiple outputs if desired.

Step 3: Save and submit

- Submit the following files:
  - `task.py` – Python script
  - `output.txt` – Model-generated output
  - `README.txt` – Brief reflection
    - Model used
    - System prompt
    - Short observation about the response
- Zip all files as: `lastname_firstname_ollama_lab7.zip`

Evaluation criteria

- 3 pts – Python Script: Runs without errors and correctly uses Ollama.
- 2 pts – Output File: Output is generated locally and appropriate for the defined task.
- 1 pt – Reflection (README): Clear and concise explanation of model choice and prompt effects.

Presentation

- Suruthi - Smith (2020). *Contextual Word Representation: A Contextual Introduction*
- Billy - Huang et al. (2018). *Music Transformer*

Outline

Next week

Tuesday

- Project group meeting (Work in a group, Check-in with me)

Thursday

- Project proposal presentations (13 groups total; `https://hksung.github.io/Spring26_PSYC681/project/`)
- 5-minute informal update on your progress so far

Upcoming Deadline

- Submit the full project proposal by 3/13
- Guidelines: `https://hksung.github.io/Spring26_PSYC681/assignments/2_project%20proposal`