Infomaid: An Offline AI Text-Generative Tool to Support Ethical Learning with Textual Assistance

Oliver Bonham-Carter, Ph.D.

https://github.com/developmentAC/infomaid

Dept of Computer and Information Science, Meadville, PA

ALLEGHENY COLLEGE

https://www.oliverbonhamcarter.com

obonhamcarter@allegheny.edu

Presented at PyCon 2025

Project Objectives

Online generative AI technologies are useful for summarizing and working with text. However, many are proprietary and raise serious privacy concerns for users.

- User prompts may contain sensitive data
- Online AI solutions often function without transparency

Our Solution: *Infomaid*

- An *offline* AI assistant, integrating Ollama models, coded in Python and Poetry
- Ideal for teaching AI applications in education, research, and private areas
- Prioritizing AI education, data privacy and security

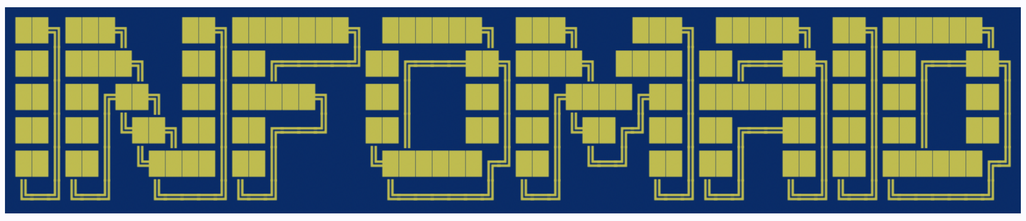

Figure: 1 A teaching and learning tool for generative AI.

Features

- **Offline**: Run fully disconnected from the Internet
- **Markdown Output**: Easy to archive, read, and collaborate
- **Simple Construction**: The code can be modified for new implementations
- **Simple CLI**: Framed in Poetry for command line interaction
- **No Telemetry**: All data from usage remains locally

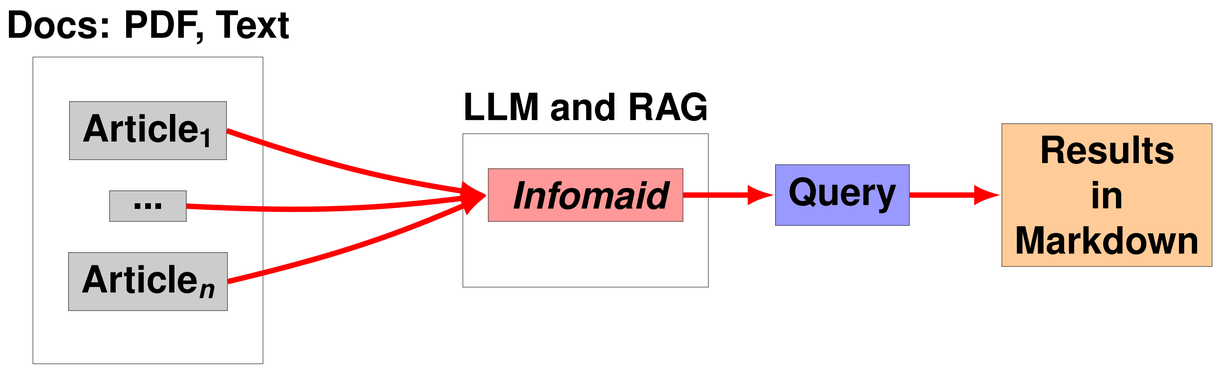

Figure: 4 RAG support to be able to query local documents.

Motivations

Safe usage of AI text generative technologies: Novices can learn to harness AI without the data security concerns associated with online services

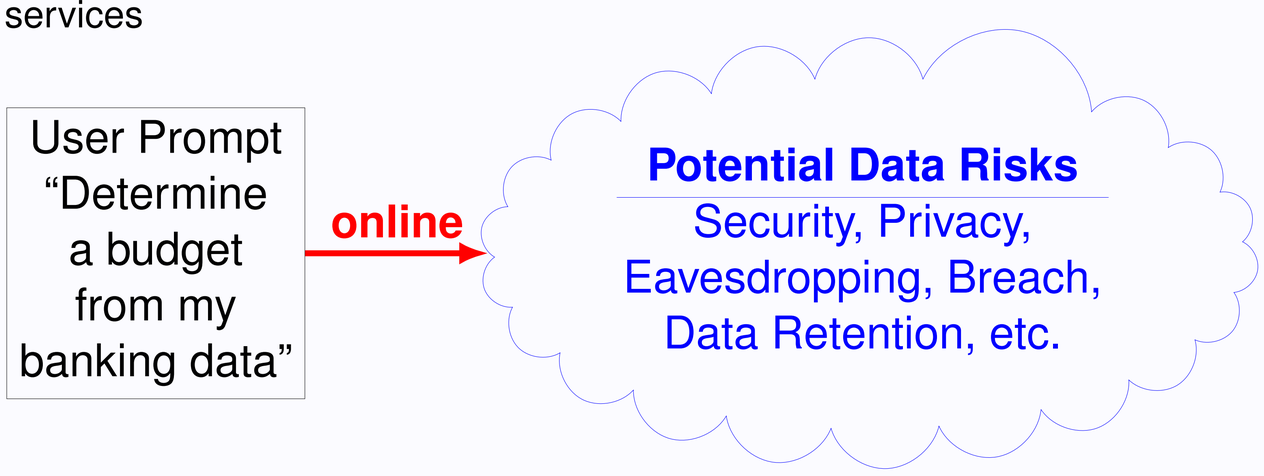

Figure: 2 Online tools are inconvenient for teaching and learning about AI.

- Online services (*e.g.,* ChatGPT, etc) tend to store user data and create privacy and data security concerns
- Novices may submit personal data unknowingly when learning prompt engineering

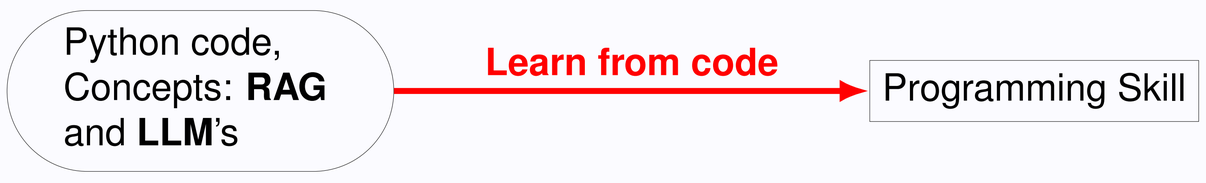

Figure: 3 Code may be studied to learn how to integrate AI technologies.

- The project has been documented for beginners
- The code is clearly written, and would be simple to modify
- Different Ollama models may be used for specific tasks
- Accessible computing: no network necessary to execute
- *Infomaid* trains users in prompt engineering
- Results are in Markdown formatting and may be archived

Query Support

- Supports prompt-based interaction for pipelines and workflows
- Run Ollama models on seemingly any hardware
- RAG support: Create custom models from local documents (*e.g.,* PDF, TXT, XML)
- *Infomaid* uses open source packages and costs nothing to use
- Local models are stored as databases and are processed by user-selected Ollama models.

Using Infomaid

The commands to setup and access Infomaid's help screen:

- Initialize project: **`poetry install`**
- Online help is available to assist user;
  - **`poetry run infomaid --bighelp`**

```
+ Ask a silly question of generative AI!
      ->  poetry run infomaid --prompt "name four shapes"
+ Use general chat, give me two results to consider, not using pdf data
      ->  poetry run infomaid --count 2 --prompt "describe four breeds of dogs"
+ Reset and build own model trained model with local data,
       use --usepdf or ---usexml options.
      ->  poetry run infomaid --resetdb
  * Reset and build own model trained model with PDF files.
      ->  poetry run infomaid --resetdb --usepdf
  * Reset and build own model trained model with XML files.
      ->  poetry run infomaid --resetdb --usexml
  * Reset and build own model trained model with TXT files.
      ->  poetry run infomaid --resetdb --usetxt
  * Reset and build own model trained model with CSV files.
      ->  poetry run infomaid --resetdb --usecsv
+ Use own model trained model as data source. Ask me for the prompt.
      ->  poetry run infomaid --useowndata
+ Query own model trained model with supplied prompt and provide output.
      ->  poetry run infomaid --useowndata --prompt "Whose name is on the included CV?"
+ Use the prompt details of the supplied file for generative AI results
      ->  poetry run infomaid --promptfile promptFiles/tell_me_a_joke.txt
```

Figure: 5 To facilitate the user experience, online help is available.

Running a Simple Query

- For example, in Figure 6, a query command is executed to provide a report of dog breeds.
- **`poetry run infomaid`**
  `--prompt "describe four breeds of dogs"`
- The default model is **`mistral`**, but it can be changed in the code. All output, including the prompt, is saved in Markdown format.

```
Code prompt:
describe four breeds of dogs
Model: mistral
Number of stories to create: 1
       Creating :0_out/
Number 0 written to the newly created file.
The current number by the file is :0
Saving the story: --> ./0_out/myAI_mistral_0.md
```

Figure: 6 Running a simple query using command line parameters. Results are called *stories*.

References

- GitHub: https://github.com/developmentAC/infomaid
- Ollama: https://ollama.com/
- Model: Nomic-Embed-Text, https://ollama.com/library/nomic-embed-text