University at Buffalo

Creating first local AI Web Stack: Ollama and Open WebUI

By

Pooja Dayanand Kabadi – 50575012

Contents

Local AI Model *1*

1. Llama 3.2 1
2. Comparative Models for Analysis 1
3. Working with Llama 3.2 3
4. Changing Model Settings 4
5. Trying it by self: Phi-3 Mini 5
6. Working on Phi-3 Mini 6
7. Changing Phi-3 Mini Model Settings 6

Open Web User Interface to Model *10*

1. Open Web UI-Llama 3.2 10
2. Open Web UI-Changing Model Settings 11
3. Open Web UI- Try it to by self: Phi-3 Mini Model 11

Reflection of Excercise *14*

1. Key Learnings 14
2. Challenges 14
3. Surprise Insights 15
4. Final Thoughts 16

List of Figures

Figure 1 Llama 3.2 First Prompt 3

Figure 2 Changing Model Settings 4

Figure 3 Phi-3 Mini 5

Figure 4 Phi-3 Mini first prompt 6

Figure 5 Custom model settings Temperature and Top_k-1 6

Figure 6 Custom model settings Temperature and Top_k-2 7

Figure 7 Custom model settings Temperature and Top_k-3 8

Figure 8 Custom model settings Temperature and Top_k-4 9

Figure 9 Open Web UI Llama 3.2 first prompt 10

Figure 10 Open Web UI-Changing model settings 11

Figure 11 Open Web UI Phi 3-Mini First Prompt (Default Settings) 11

Figure 12 Open Web UI Phi 3-Mini (Temperature 0.1) 12

Figure 13 Open Web UI Phi 3-Mini (Top-P 0.3) 12

Figure 14 Open Web UI Phi 3-Mini (System Prompt) 12

Figure 15 Open Web UI Phi 3-Mini (Temperature 0.1 + Top-P 0.3 + System Prompt) 12

Local AI Model

After completing the previous assignment, we gained a clearer understanding of local Large Language Models (LLMs) and their practical applications across different domains. It is evident that AI significantly enhances efficiency and automation in various tasks. In this assignment, we will focus on the process of locally hosting an LLM using Open-Web UI, exploring its setup, configuration, and potential use cases using Llama 3.2 as the main model and other 2 different models to compare with.

1. Llama 3.2

The main model Llama 3.2 is Meta's advanced large language model collection, featuring both lightweight text-only models and more robust multimodal models capable of processing text and images. For this project, we have selected the 3-billion (3B) parameter text-only model, as it is optimized for deployment on edge devices with limited computational power. This model provides a balance between performance and efficiency, making it well-suited for applications such as text generation, summarization, and chatbot interactions while ensuring fast response times. Although it may not match the reasoning capabilities of larger models, its low resource requirements make it an ideal choice for running locally without heavy infrastructure.

2. Comparative Models for Analysis

a. Phi 3-Mini

Phi-3-Mini is a compact and highly efficient language model developed by Microsoft, featuring 3.8 billion parameters. Despite its smaller size, it is trained on 3.3 trillion tokens, enabling it to compete with larger models like GPT-3.5 and Mixtral 8x7B on various benchmarks. Phi-3-Mini is designed to be resource-efficient, making it ideal for deployment on mobile devices, embedded systems, and edge computing environments. The model is particularly well-suited for text-based AI applications such as chatbots, content generation, and lightweight NLP tasks. While it may have some limitations in complex reasoning, it offers fast processing speeds and optimized performance, making it an excellent choice for AI applications requiring low latency and minimal hardware demands.

b. Openchat

The other model that is been considered is OpenChat 7-billion-parameter open-source language model, which is optimized for conversational AI and general NLP tasks. It is designed for efficient deployment on mid-range GPUs, balancing performance and resource consumption. Trained on high-quality datasets, it ensures coherent and context-aware responses, making it suitable for chatbots, text generation, and summarization. While it lacks the depth of larger models, it provides fast inference and adaptable performance, making it a practical choice for real-time applications.

Table 1 Overview Comparative Model for Analysis

Feature	Llama 3.2	Phi 3-Mini	OpenChat
Developer	Meta	Microsoft	Openchat(community)
Model Type	Text Only	General Purpose	Conversational
Parameters	3B	3.8B	7B
Training Data	High-quality diverse dataset	3.3T tokens from high-quality sources	Curated conversational datasets
Why Selected?	Efficient and fast for local AI tasks	Compact but powerful, optimized for low-resource deployment	Open-source, well-suited for chatbot applications

3. Working with Llama 3.2

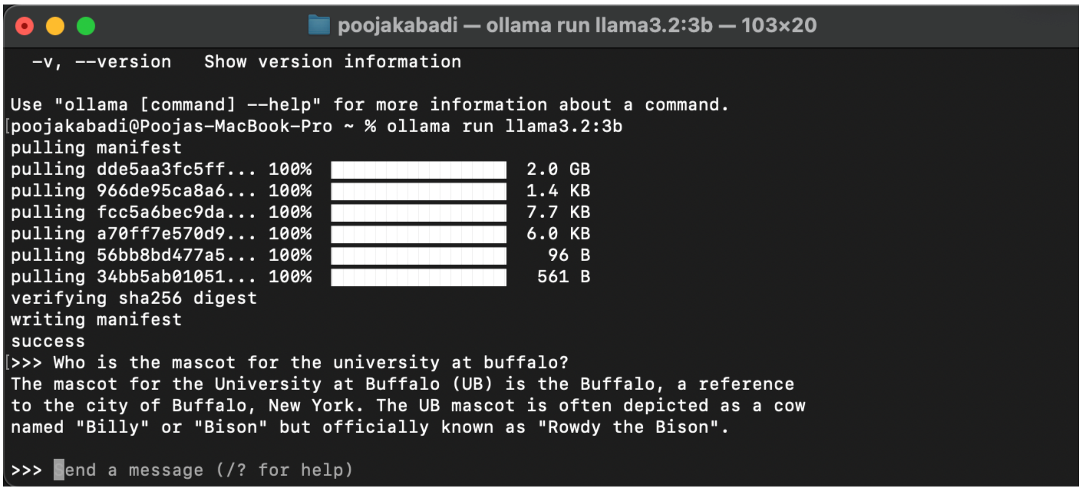


Figure 1 Llama 3.2 First Prompt

The answer provided by the model is incorrect because the official mascot for the University at Buffalo (UB) is Victor E. Bull, not "Rowdy the Bison" or a cow named "Billy." This suggests that the model might be generating plausible but incorrect information, possibly due to following reasons: -

- **Limited Training Data** – As a locally hosted model, its knowledge is restricted to the data it was trained on. If it has not been specifically fine-tuned on accurate and updated information about UB's mascot, it may produce incorrect responses.
- **Lack of Context Verification** – Unlike cloud-based AI models that can cross-check multiple sources in real time, this model relies solely on its pre-trained dataset, limiting its ability to verify facts.
- **Hallucination** – LLMs sometimes generate plausible but incorrect responses when they lack definitive knowledge. Instead of indicating uncertainty, they fabricate answers that seem reasonable.
- **Generalization Bias** – The model may have inferred incorrect mascot names by associating UB with other universities or teams that have buffalo-related mascots, leading to the errors.

4. Changing Model Settings

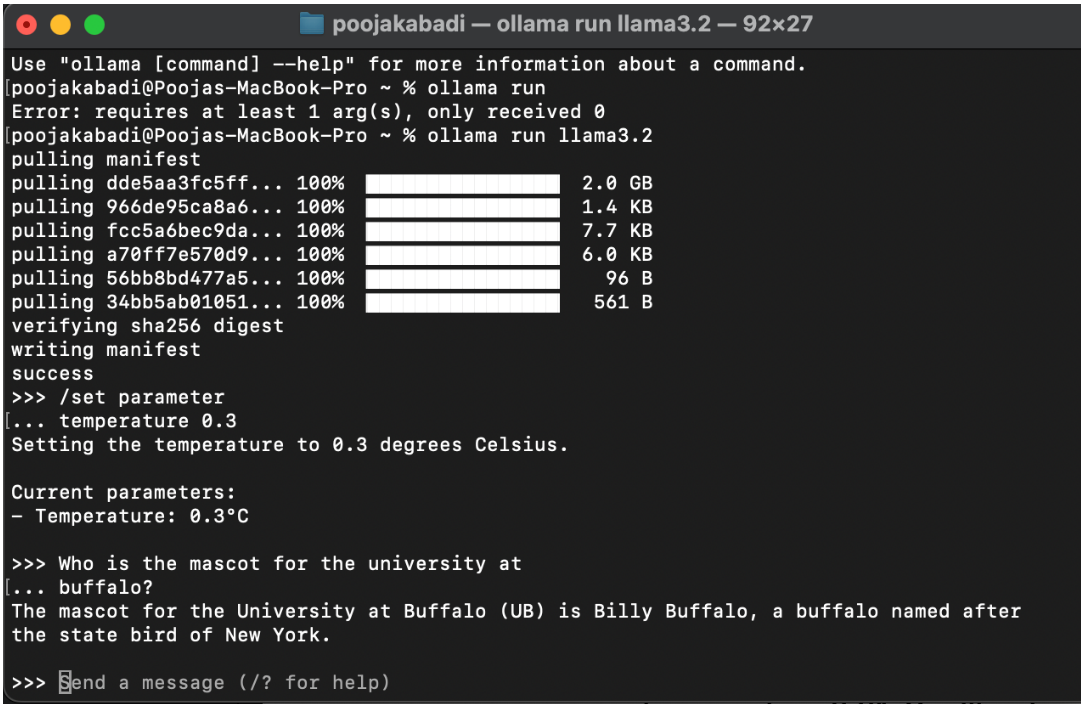


Figure 2 Changing Model Settings

Lowering the temperature from **0.8 to 0.3** made the model's responses more deterministic, reducing randomness. However, it still provided incorrect information about the University at Buffalo's mascot, indicating that the issue lies in its training data rather than response variability.

Differences Observed:

a. **Variation in Incorrect Answers** - At **0.8**, the model incorrectly named "Rowdy the Bison." At **0.3**, it generated a different incorrect answer, "Billy Buffalo," and falsely linked it to the state bird of New York.

b. **Less Randomness, But Still Inaccurate** - While lowering the temperature reduced response variation, it did not improve factual accuracy. This suggests that the model lacks reliable training data on this specific topic rather than simply generating random errors.

5. Trying it by self: Phi-3 Mini

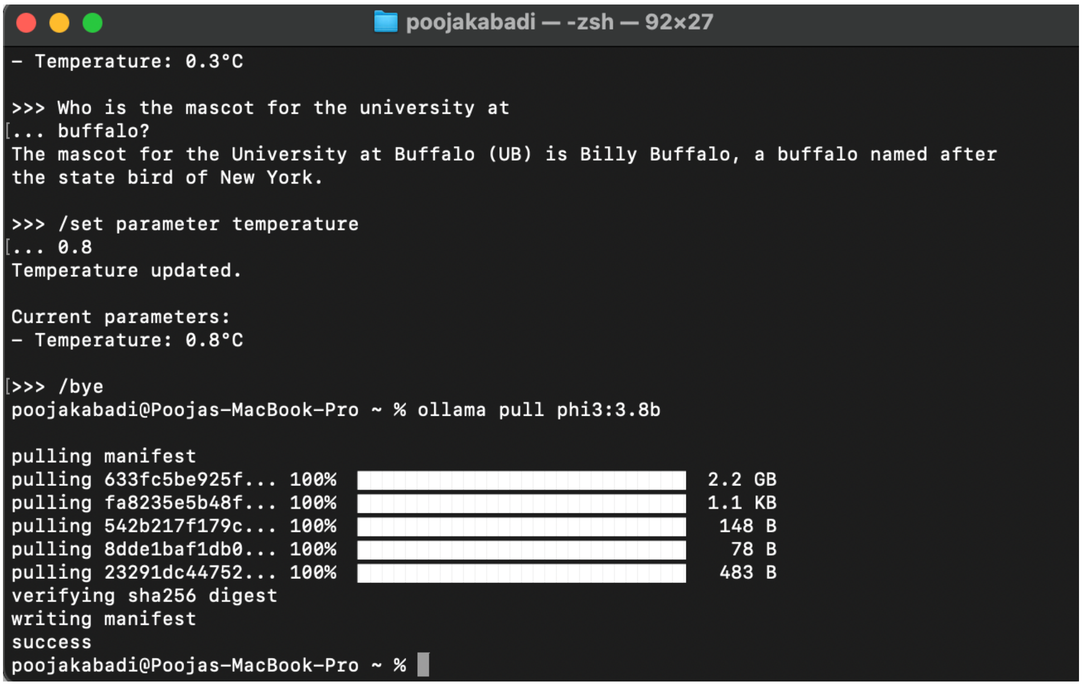


Figure 3 Phi-3 Mini

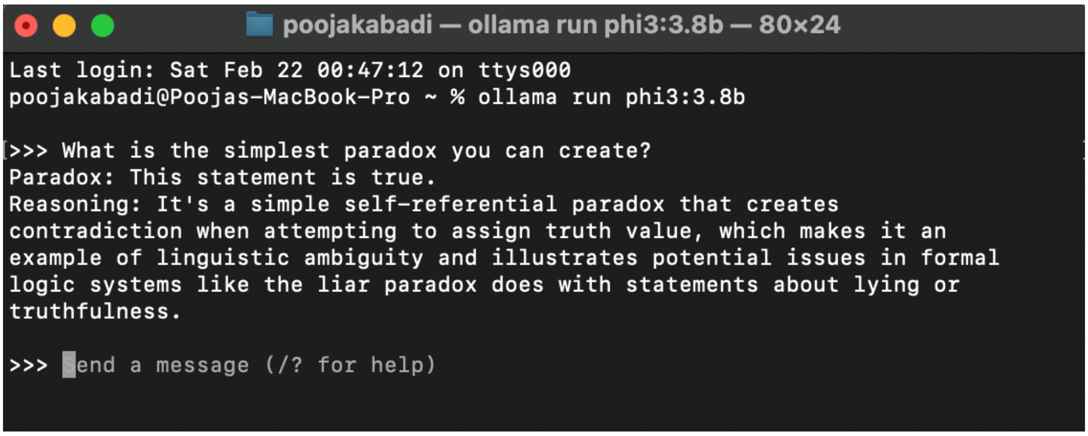
a. Why Phi-3 Mini?

As explained before in this report this model is an **efficient, small-scale model** that balances strong performance with low computational requirements developed by Microsoft, which is designed to offer compact but powerful, optimized for low-resource deployment.

b. What is unique about Phi-3 Mini?

- ✓ **Training Efficiency** – Phi-3 Mini (3.8B) delivers performance comparable to larger models like GPT-3.5, while Llama 3.2 (3B) focuses on efficiency but lacks direct comparisons.
- ✓ **Data Size** – Phi-3 Mini is trained on **3.3T high-quality tokens**, emphasizing reasoning and coding. Llama 3.2 does not disclose token size or focus areas.
- ✓ **Optimization** – Phi-3 Mini is highly optimized for **low-resource deployment** with efficient inference. Llama 3.2 is also optimized for edge devices but without the same performance claims.
- ✓ **Reasoning Ability** – Phi-3 Mini excels in **problem-solving and structured reasoning**, while Llama 3.2 is a general-purpose NLP model.

6. Working on Phi-3 Mini



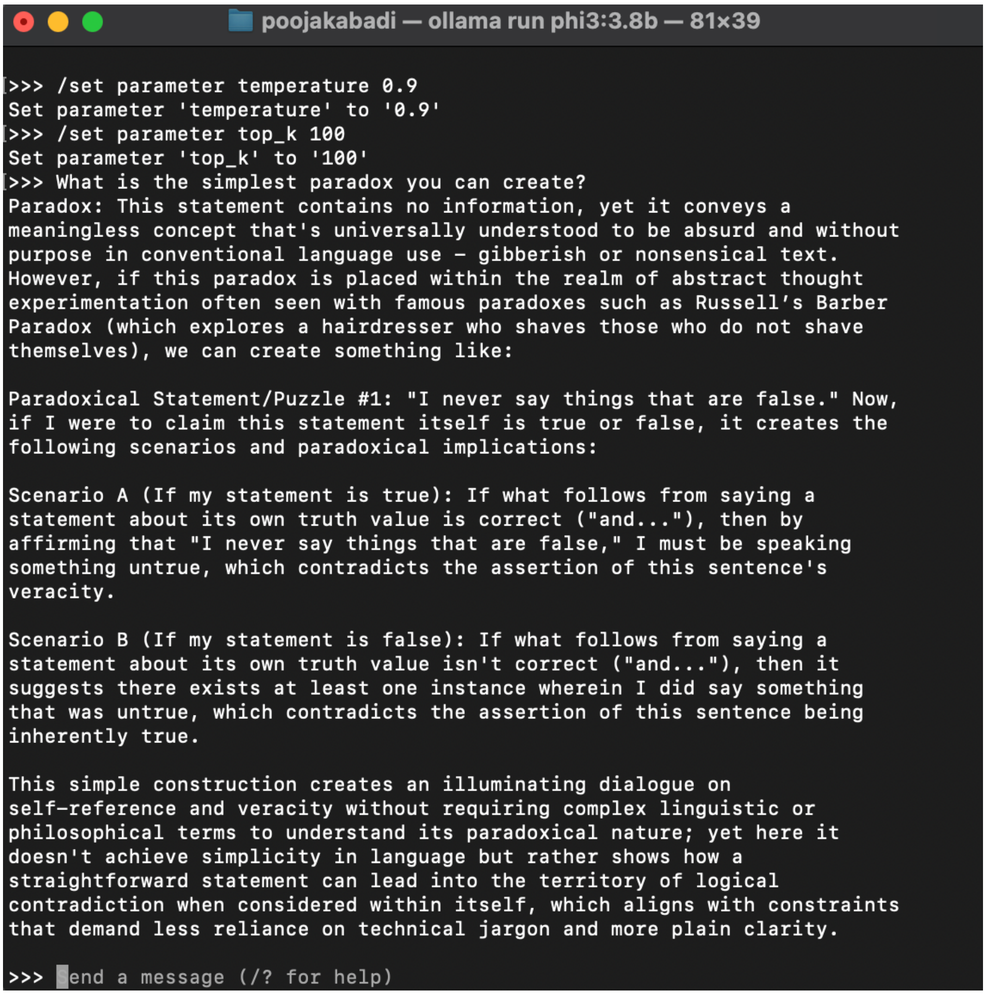
```
poojakabadi — ollama run phi3:3.8b — 80×24
Last login: Sat Feb 22 00:47:12 on ttys000
poojakabadi@Poojas-MacBook-Pro ~ % ollama run phi3:3.8b

>>> What is the simplest paradox you can create?
Paradox: This statement is true.
Reasoning: It's a simple self-referential paradox that creates
contradiction when attempting to assign truth value, which makes it an
example of linguistic ambiguity and illustrates potential issues in formal
logic systems like the liar paradox does with statements about lying or
truthfulness.

>>> Send a message (/? for help)
```

Figure 4 Phi-3 Mini first prompt

7. Changing Phi-3 Mini Model Settings

a. Higher Temperature & Higher Top-K

Changing the model setting from temperature=0.8 to 0.9 and top_k=40 to 50, which leave us with and expected outcome of more creative and exploratory



```
poojakabadi — ollama run phi3:3.8b — 81×39
>>> /set parameter temperature 0.9
Set parameter 'temperature' to '0.9'
>>> /set parameter top_k 100
Set parameter 'top_k' to '100'
>>> What is the simplest paradox you can create?
Paradox: This statement contains no information, yet it conveys a
meaningless concept that's universally understood to be absurd and without
purpose in conventional language use - gibberish or nonsensical text.
However, if this paradox is placed within the realm of abstract thought
experimentation often seen with famous paradoxes such as Russell's Barber
Paradox (which explores a hairdresser who shaves those who do not shave
themselves), we can create something like:

Paradoxical Statement/Puzzle #1: "I never say things that are false." Now,
if I were to claim this statement itself is true or false, it creates the
following scenarios and paradoxical implications:

Scenario A (If my statement is true): If what follows from saying a
statement about its own truth value is correct ("and..."), then by
affirming that "I never say things that are false," I must be speaking
something untrue, which contradicts the assertion of this sentence's
veracity.

Scenario B (If my statement is false): If what follows from saying a
statement about its own truth value isn't correct ("and..."), then it
suggests there exists at least one instance wherein I did say something
that was untrue, which contradicts the assertion of this sentence being
inherently true.

This simple construction creates an illuminating dialogue on
self-reference and veracity without requiring complex linguistic or
philosophical terms to understand its paradoxical nature; yet here it
doesn't achieve simplicity in language but rather shows how a
straightforward statement can lead into the territory of logical
contradiction when considered within itself, which aligns with constraints
that demand less reliance on technical jargon and more plain clarity.

>>> Send a message (/? for help)
```

Figure 5 Custom model settings Temperature and Top_k-1

After increasing temperature to 0.9 and top_k to 100, the response became longer, more exploratory, and creative. Instead of simply stating a paradox, the model introduced multiple paradoxical constructs, including references to Russell's Barber Paradox and a new self-referential statement ("I never say things that are false"). The explanation also became more philosophical, offering two possible logical scenarios rather than just one.

b. Lower Temperature & Lower Top_K

Changing the model setting from temperature=0.8 to 0.2 and top_k=40 to 20, which leave us with and expected outcome of more structured and deterministic.

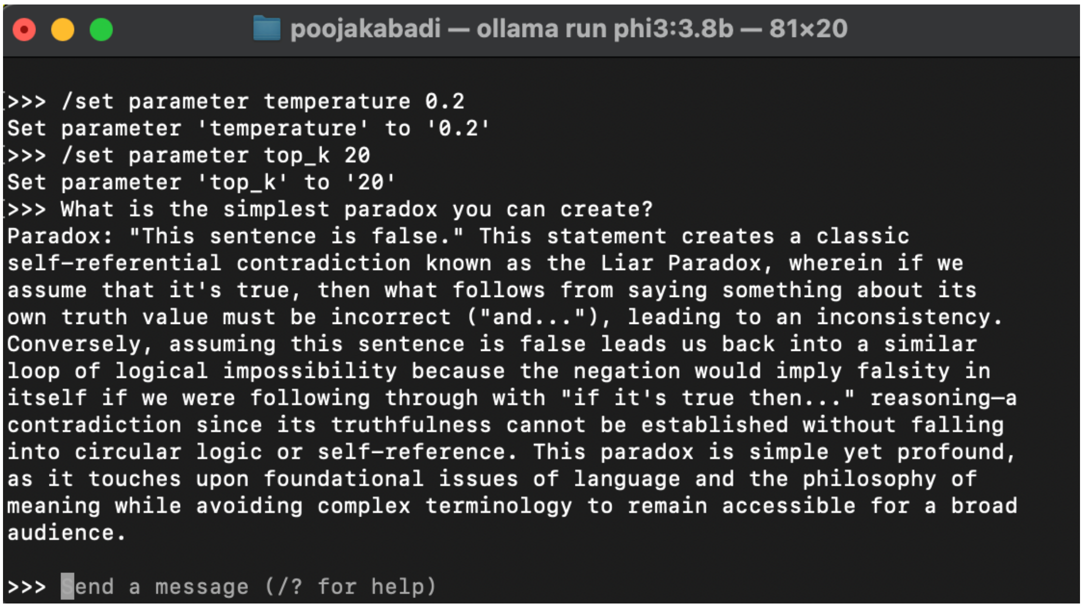


Figure 6 Custom model settings Temperature and Top_k-2

After lowering the temperature to 0.2 and top_k to 20, the model provided a structured analysis of the paradox, prioritizing logical accuracy over creative interpretation. It remained focused, avoiding unnecessary tangents, and presented a clear, singular explanation rather than exploring multiple perspectives.

c. Default Temperature, Low Top-K

Changing the model setting from top_k=40 to 30 and default temperature, which leave us with and expected outcome of more logical yet diversified response.

Figure 7 Custom model settings Temperature and Top_k-3

After maintaining default temperature and top-K 30, the model expanded significantly on its answer, introducing a new paradox involving two sequential statements. It explored different logical scenarios, considering the implications of truthfulness vs. falsehood in a self-referential system. The response became more philosophical and abstract, adding layers of reasoning.

d. **Low Temperature and High Top-K**

Changing the model setting from top_k=40 to 30 and default temperature, which leave us with and expected outcome of more logical yet diversified response.

```
poojakabadi — ollama run phi3:3.8b — 81×33

>>> /set parameter temperature 0.3
Set parameter 'temperature' to '0.3'
>>> /set parameter top_k 80
Set parameter 'top_k' to '80'
>>> What is the simplest paradox you can create?
Paradox: "This sentence will not be read." This statement creates a classic
example known as the Epimenides Paradox, where if we assume it's true that
this very sentence won't be read, then by its own admission of being
unreadable or irrelevant to our current context (constraint 3), one might
argue there is no reason for us not to disregard it. However, the act of
reading—or in this case, ignoring—the statement would nullify that very
assumption if we were considering whether or not its contents are relevant
based on their readability; thus leading into a paradoxical situation
wherein our actions (reading/not-reading) directly contradict what is
stated within it.

This simple yet profoundly self-referential construct touches upon the
essence of communication and comprehension without invoking complex
philosophical terminology, aligning with constraint 4 by focusing on a
fundamental aspect that can be universally understood: relevance in reading
material based on its content or perceived significance. It also adheres to
constraints (1) through (3), as it is straightforwardly paradoxical without
the need for specialized language, and does not rely heavily upon
historical contexts—instead presenting a timeless logical conundrum that
can be appreciated across cultures due to its universal nature of
self-reference. The Epimenides Paradox serves as an accessible entry point
into discussions about truth in statements regarding their own consumption
or disregard, inviting contemplation on the act and implications of reading
itself without necessitating a deep dive into dense philosophical
discourse—a paradox that is simple to understand yet rich with meaning.

>>> Send a message (/? for help)
```

Figure 8 Custom model settings Temperature and Top_k-4

After increasing temperature to 0.3 and top-K to 80, the model introduced a different paradox ("This sentence will not be read") and provided a more detailed logical breakdown. It referenced the Epimenides Paradox, expanded on how reading or ignoring a statement creates a contradiction, and avoided complex philosophical jargon while keeping the discussion insightful.

Open Web User Interface to Model

1. Open Web UI-Llama 3.2

Figure 9 Open Web UI Llama 3.2 first prompt

2. Open Web UI-Changing Model Settings

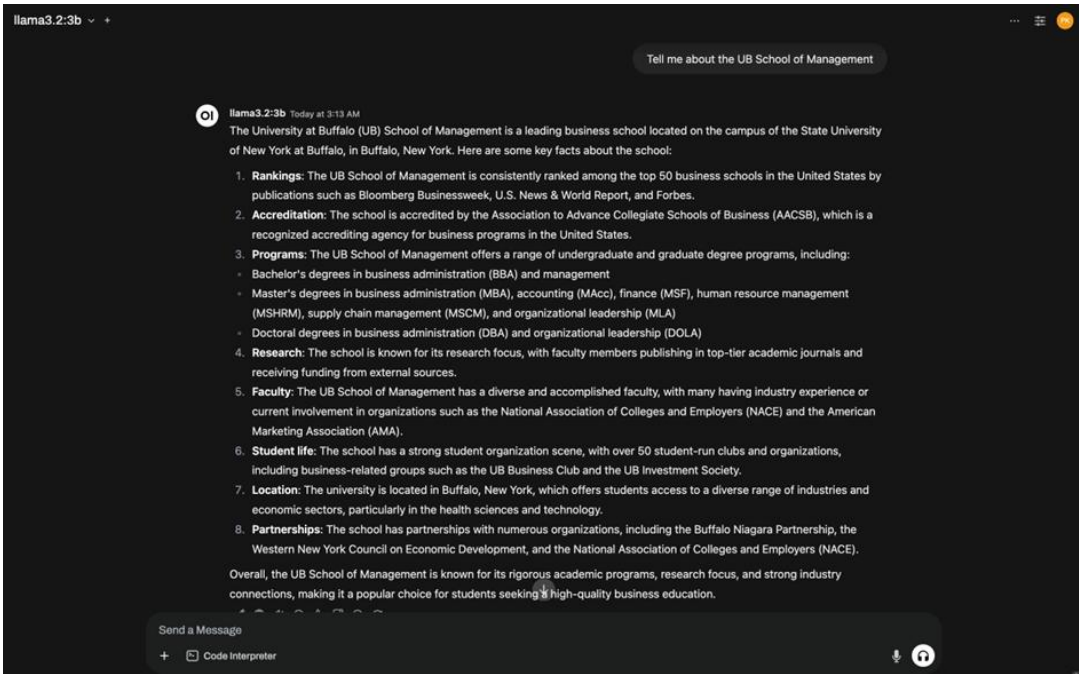


Figure 10 Open Web UI-Changing model settings

The first output (default settings) provides a well-structured but slightly generalized response, covering rankings, academic programs, research focus, and networking opportunities.

In contrast, the second output (temperature 0.1) is more factual, detailed, and deterministic, including specific accreditation details (AACSB), additional degree programs, faculty involvement in organizations, and student life insights. The lower temperature reduced variability, making the response more precise and information-dense, avoiding any unnecessary creativity or variation.

3. Open Web UI- Try it to by self: Phi-3 Mini Model

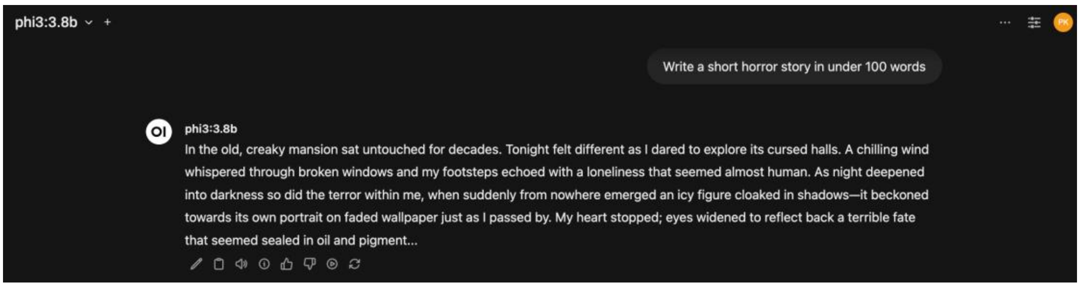


Figure 11 Open Web UI Phi 3-Mini First Prompt (Default Settings)

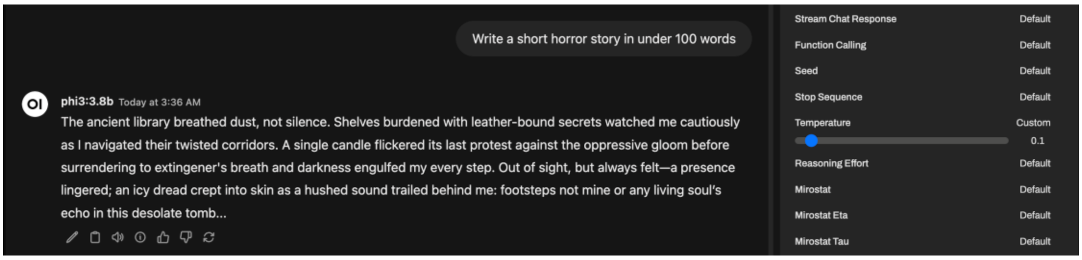


Figure 12 Open Web UI Phi 3-Mini (Temperature 0.1)

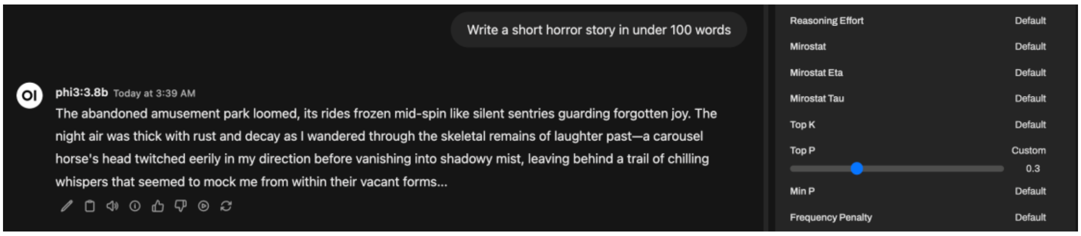


Figure 13 Open Web UI Phi 3-Mini (Top-P 0.3)

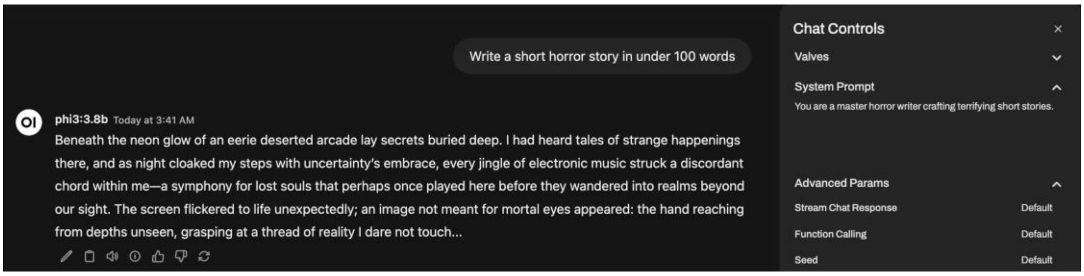


Figure 14 Open Web UI Phi 3-Mini (System Prompt)

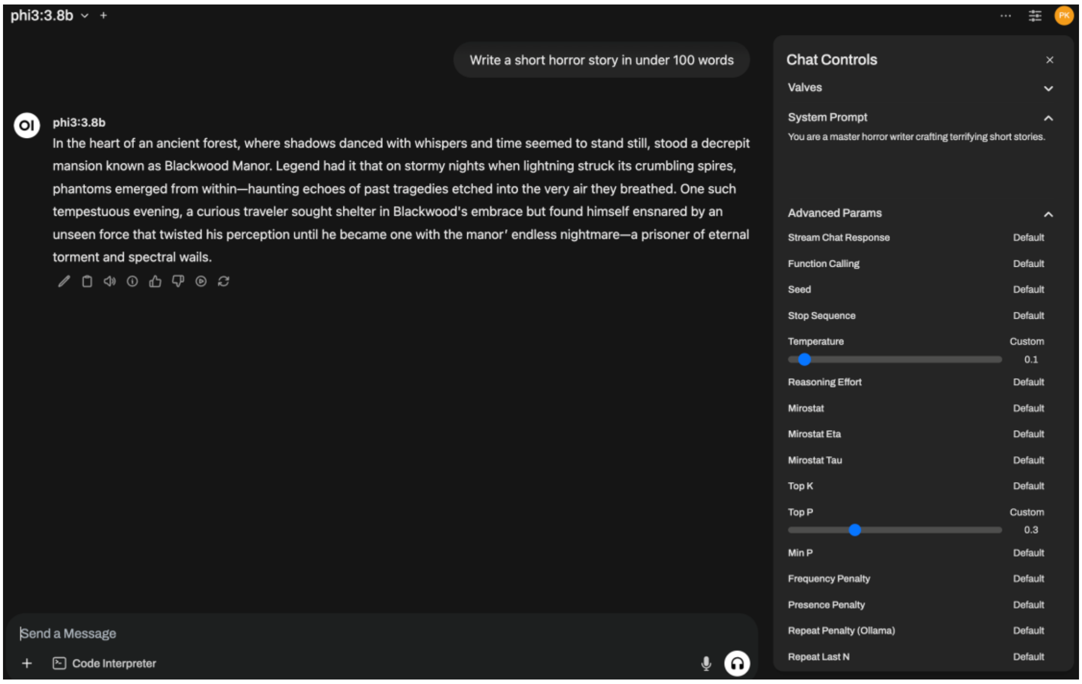


Figure 15 Open Web UI Phi 3-Mini (Temperature 0.1 + Top-P 0.3 + System Prompt)

Parameter Setting	Story Theme	Key Observations
Default Settings	Traditional horror narrative with a cursed mansion	Well-structured, engaging, but somewhat predictable. Balanced creativity and coherence
Temperature 0.1	Ancient library setting, more deterministic storytelling	Logical, consistent, and structured. Reduced creativity, making the story more refined but less experimental
Top-P 0.3	Abandoned amusement park with eerie horror elements	More focused response, sticking to common word choices. Horror elements were well-formed but lacked extreme surprises
Custom System Prompt (You are a master horror movie)	Deserted arcade, psychological horror, immersive descriptions	AI adopted a professional horror-writing style. Writing was more atmospheric, suspenseful, and descriptive
Temperature 0.1 + Top-P 0.3 + System Prompt	Blackwood Manor, deep psychological horror	Most structured and logical output. Maintained tight control over randomness while ensuring rich, immersive storytelling

Reflection of Excercise

This exercise provided a deeper understanding of how locally hosted AI models function and how **fine-tuning parameters** can drastically alter their behaviour. While I have interacted with AI models before, working with **Ollama and Open Web-UI** gave me hands-on experience in deploying and controlling AI responses without relying on cloud-based solutions. The process was both **insightful and challenging**, requiring troubleshooting, iteration, and analysis to grasp the full impact of each parameter change.

1. Key Learnings

One of the most significant takeaways was how parameters such as **temperature, Top-K, and Top-P** influence AI-generated content. I initially thought that temperature was the most important factor in determining the randomness of responses, but experimenting with **Top-P and Top-K** showed that these parameters also played a crucial role in controlling how diverse or constrained the output could be.

For example, when generating horror stories, increasing the temperature led to more unpredictable and creative responses, while lowering it made the outputs more structured and deterministic. Similarly, Top-P adjustments refined word selection, ensuring responses stayed within a specific range of probabilities. When Top-P was set lower, the stories were more focused and traditional, whereas higher values introduced more surprising and eerie elements. This demonstrated that AI creativity can be fine-tuned based on use-case requirements.

2. Challenges

One of the biggest challenges was setting up Ollama and Open Web-UI correctly. Initially, I encountered errors related to Python virtual environments and missing dependencies, which required troubleshooting before I could host Open Web-UI server. This was a reminder of how open-source tools require technical understanding to configure properly, unlike cloud-based AI services that provide a more seamless user experience.

Another difficulty was ensuring **response accuracy**, especially when asking factual questions. When I queried the AI about **the mascot for the University at Buffalo**, the model hallucinated incorrect answers despite parameter adjustments. Even when I lowered the temperature to make responses more deterministic, the AI still returned misinformation, reinforcing the fact that parameter tuning does not improve factual accuracy, it only controls variability in responses. This highlighted one of the core limitations of large language models: they generate statistically probable answers rather than factually verified information.

3. Surprise Insights

One of the most unexpected insights came from modifying the **system prompt**. When I changed it to "**You are a master horror writer crafting terrifying short stories**" the AI's storytelling style immediately became more immersive, suspenseful, and atmospheric. This showed how prompt engineering can dramatically shape AI behaviour beyond just tweaking parameters. The ability to guide the model's personality, expertise, and tone through well-crafted system prompts opens up possibilities for customizing AI responses for specific applications, from creative writing to professional business reports.

Another surprising discovery was how Open Web-UI provided a better user experience than command-line interactions. The ability to visually adjust parameters and analyse AI responses in a structured way made experimentation more intuitive. The Web-UI interface also provided better control over iterative testing, making it easier to compare how different settings influenced the model's output.

4. Final Thoughts

Overall, this experiment deepened my understanding of AI model tuning and local inference. While cloud-based AI models like ChatGPT provide seamless access, working with self-hosted AI demonstrated the advantages of privacy, flexibility, and direct control over model behaviour. However, it also highlighted the challenges of managing installations, troubleshooting dependencies, and dealing with AI hallucinations without external validation.

The biggest takeaway was that parameter tuning is not just about making AI more creative or structured, it's about tailoring responses to fit a specific purpose. This exercise reinforced the importance of iterative testing, prompt engineering, and understanding AI behaviour at a granular level. Moving forward, I am more aware of how to strategically adjust AI parameters for different tasks, whether for structured business insights, creative storytelling, or fact-driven responses.

This hands-on approach has left me excited about further exploring prompt optimization and AI customization for real-world applications.