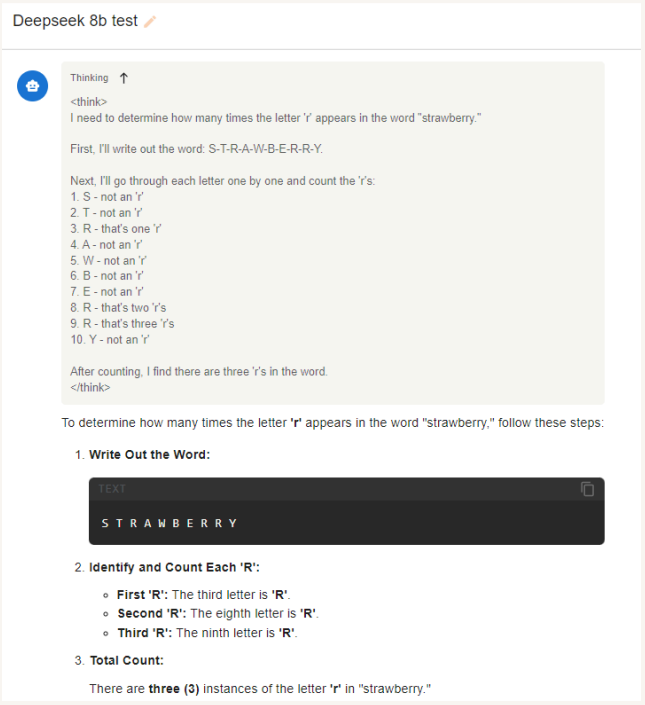
NextWork.org

How to Use DeepSeek

DI Dineshraj Dhanapathy



DI
Dineshraj Dhanapathy
NextWork Student
NextWork.org

Introducing Today's Project!

In this project, I will demonstrate testing a range of prompts with DeepSeek. I'm doing this project to learn how DeepSeek compares to ChatGPT, optimizing it locally using Ollama and Chatbox for best results.

Tools and concepts

Services I used were DeepSeek, Ollama, and Chatbox. Key concepts I learnt include local inference, model quantization, and temperature control. After reviewing DeepSeek and OpenAI, I personally preferred DeepSeek for privacy-focused offline use.

Project reflection

This project took me approximately 2-3 hours. The most challenging part was troubleshooting model compatibility in Ollama-Chatbox integration. It was most rewarding to achieve fully private, offline AI functionality.

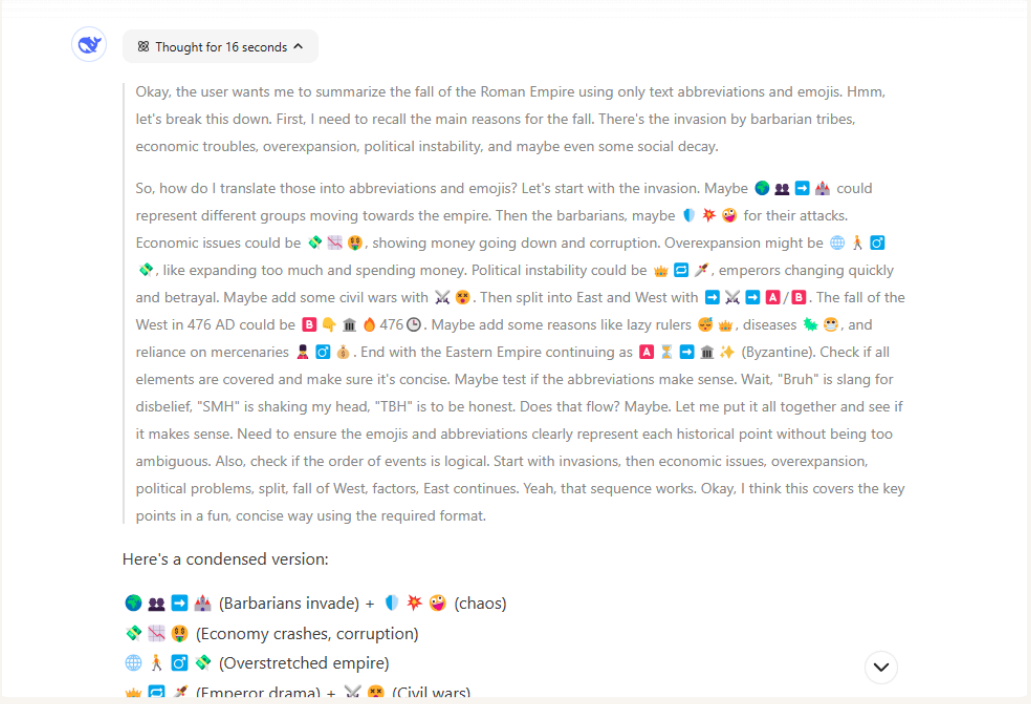
I pursued this project today to explore open-source AI alternatives addressing privacy/offline needs. It met goals: local DeepSeek deployment worked (despite 1.5B errors), proving viable for private use while highlighting model-size tradeoffs.

DI
Dineshraj Dhanapathy
NextWork Student

Exploring DeepSeek

DeepSeek is a Chinese AI company focused on AGI research. Their R1 model gained attention for outperforming top models like GPT-4 in reasoning benchmarks (MATH, GSM8K) and coding, while being open-source and cost-efficient for developers.

While you could access DeepSeek over the web app, some concerns are privacy/data control, latency, and API costs. To address these concerns, hosting R1 locally via Ollama ensures on-prem data processing, faster response times, and cost savings.



DI
Dineshraj Dhanapathy
NextWork Student

Ollama and DeepSeek R1

Ollama is an open-source tool for running large language models (LLMs) locally, enabling offline AI tasks with privacy-first data processing. It supports models like DeepSeek R1 and LLaMA through simple CLI commands.

You won't be able to find OpenAI models in Ollama because they're closed-source, requiring API access rather than local deployment. Ollama only supports open-weight models (like LLaMA, DeepSeek R1) that allow self-hosting and offline inference.

I tested using DeepSeek offline by off wifi & running ollama run deepseek-r1:8b. It responded successfully. i can also see 'think' tags in the terminal because Ollama handles inference locally; no API calls, ensure full offline functionality.

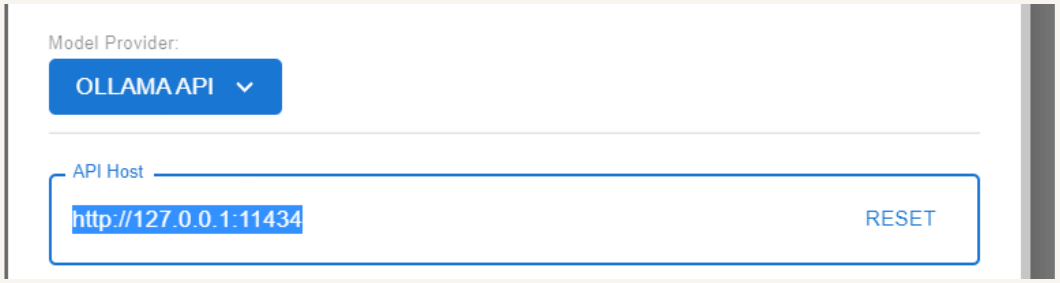
```
>>> Can I use DeepSeek while offline?
<think>
Okay, so I'm trying to figure out if I can use DeepSeek while being offline. First off, I'm not exactly sure what
DeepSeek is. From the context given, it seems like it's a tool or maybe an AI service that might require an
internet connection. The user mentioned wanting to know if they can use it without being online.

Alright, so let me break this down. If DeepSeek is a service that runs on the cloud, then it probably needs an
active internet connection to function properly. But wait, sometimes there are local installations of tools like
this where you can run them offline after initial setup. Maybe I should consider both possibilities: using it as a
web-based tool and having a local installation.

I think if DeepSeek is accessed through a browser, like via a website, then yes, without an internet connection,
the user wouldn't be able to use it because it's dependent on servers for processing or delivering content. But if
there's a downloadable version or a local setup where you can run it on your own computer offline, then maybe.

But how would that work exactly? If I download something like DeepSeek, does it become a standalone application
that doesn't require the internet after installation? That depends on what the tool is designed for. For example,
some AI tools might require updates or data from the internet, so if you're offline, they might not function as
expected.

Also, maybe there's a way to cache data or download necessary files beforehand. So perhaps the user can
pre-download whatever data or processing they need and then use it offline later. That would be helpful for
situations where you don't always have an internet connection available, like on a plane or in areas with poor
coverage.
```

DI
Dineshraj Dhanapathy
NextWork Student
NextWork.org

DeepSeek R1 Sizes

DeepSeek R1 has different model sizes, which mean varying levels of processing needs and accuracy. This is helpful for running locally because smaller/faster models suit low-power devices, while larger ones offer better reasoning for capable hardware

The R1 model you choose to run locally depends on hardware constraints (RAM/GPU) and speed vs accuracy needs. I chose the 8B model (i.e., 8 billion parameters) because its offers substantial improvement from 1.5b model.



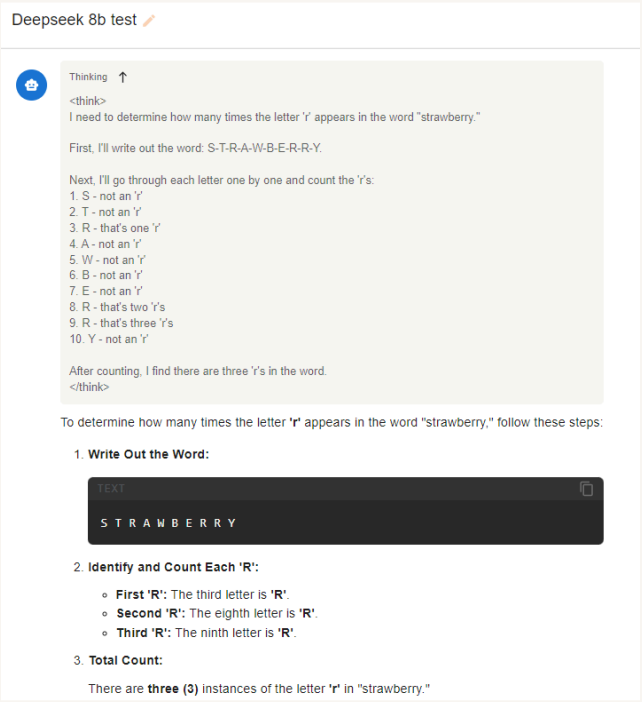
DI

Dineshraj Dhanapathy
NextWork Student

Chatbox

To complete my local setup, I installed Chatbox to replace OpenAI's web UI with a privacy-focused GUI. My Chatbox settings use Ollama's local API (http://localhost:11434) to interact directly with DeepSeek R1 models offline.

I tested two different R1 model sizes, which were 1. 5b and 8b, using the prompt of "How many r's are in strawberry?" The results made it clear that 1.5b was a lighter model. it responded with only one 'r' in "Stawberry," whereas 8b model said 3 'r'.



DI

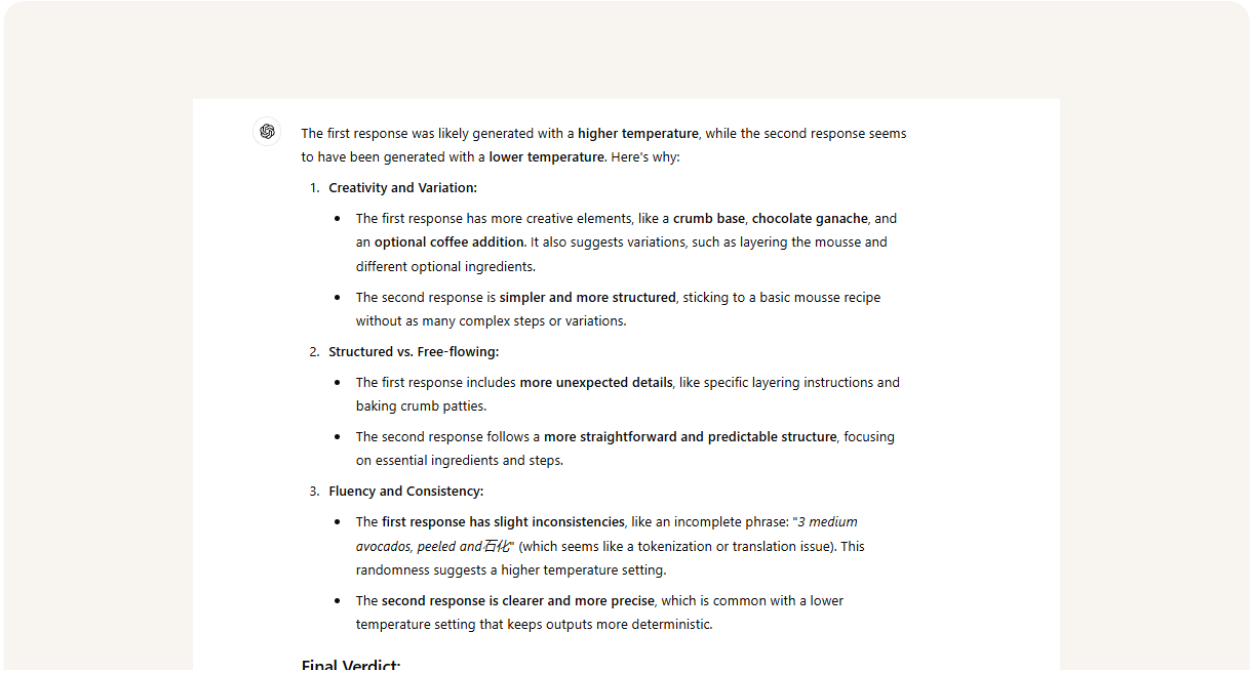
Dineshraj Dhanapathy
NextWork Student

Temperature Settings

The temperature setting in an LLM determines how creative and output randomness. To see this in action, i dialled up the temperature to the maximum (2) and observed formulaic vs. wildly varied responses.

I started a third chat with ChatGPT to act as judge for the responses. can chatGPT tell which one was generated with a high temperature? ChatGPT's analysis will also help us with understanding how to spot low vs high temperature-generated text.

ChatGPT quickly figured out which piece was generated with a high temperature because it had more creativity, unexpected details, and slight inconsistencies, while the low-temperature text was structured and direct.



The first response was likely generated with a **higher temperature**, while the second response seems to have been generated with a **lower temperature**. Here's why:

1. **Creativity and Variation:**
   - The first response has more creative elements, like a **crumb base**, **chocolate ganache**, and an **optional coffee addition**. It also suggests variations, such as layering the mousse and different optional ingredients.
   - The second response is **simpler and more structured**, sticking to a basic mousse recipe without as many complex steps or variations.
2. **Structured vs. Free-flowing:**
   - The first response includes **more unexpected details**, like specific layering instructions and baking crumb patties.
   - The second response follows a **more straightforward and predictable structure**, focusing on essential ingredients and steps.
3. **Fluency and Consistency:**
   - The **first response has slight inconsistencies**, like an incomplete phrase: "*3 medium avocados, peeled and石化*" (which seems like a tokenization or translation issue). This randomness suggests a higher temperature setting.
   - The **second response is clearer and more precise**, which is common with a lower temperature setting that keeps outputs more deterministic.

Final Verdict: