

Final Project

Local LLM for Mac Silicon Devices

Gassan Yacteen



CSCI E-104 Advanced Deep Learning, 2023
Harvard University Extension School
Prof. Zoran B. Djordjević

Introduction

- **Project Overview:** Exploring beyond my passion of interpretable AI and investigating the potential of utilizing LLMs locally on Apple Silicon devices, with a focus on overcoming native hardware limitations.
- **Motivation:** Driven by the need for enhanced data privacy amidst rising concerns over digital privacy laws.
- **Objective:** To demonstrate the feasibility of local LLM training and deployment using specialized frameworks like MLX, tailored for Mac's unique hardware.

POLITICS | NATIONAL SECURITY

Senate Passes Spying Bill, Rejecting Privacy Concerns

Intelligence officials say tool is vital to disrupting terror plots, cyberattacks

By [Dustin Volz](#) [Follow](#)

Updated April 20, 2024 at 3:13 pm ET

[Share](#)

[AA](#) [Resize](#)

[390](#)

[Listen \(2 min\)](#)



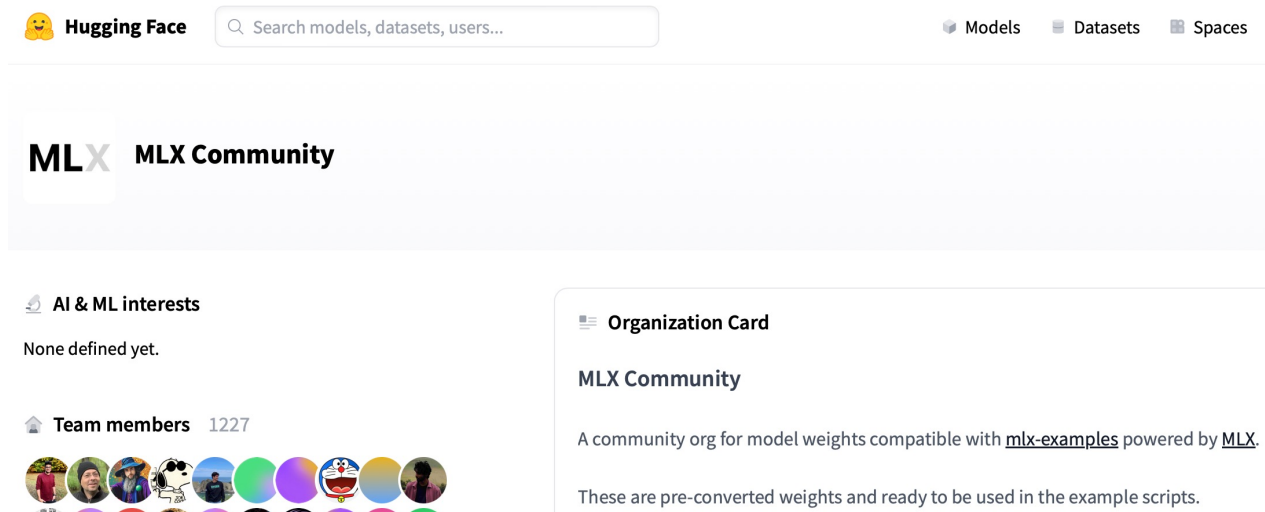
Figure 1: [WSJ and Spying Bill](#)

Problem Statement and Objectives

- **Problem Statement:** Address the challenge of deploying and training large language models (LLMs) on Apple Silicon, which lacks native CUDA support essential for Deep Learning and utilizing GPUs.
- **Enhance Local Training:** Develop a methodology to efficiently train LLMs directly on Mac hardware, leveraging Apple's Metal API.
- **Optimize Performance:** Tailor machine learning operations to utilize the full capabilities of Apple Silicon with robust computational performance without external resources.
- **Promote Data Privacy:** Implement on-device processing to maintain strict data privacy, avoiding reliance on cloud-based systems considering stringent digital privacy laws. Finally allow for a private user friendly local LLM Chatbot.

Technology Overview

- **LocalGPT:** Initially chosen for its privacy-centric design to process data directly on Apple hardware. Faced limitations in performance and efficiency.
- **MLX Library:** Selected for its superior compatibility with Apple's Metal API, enabling better utilization of native hardware acceleration.
- **Apple Silicon:** Leveraging the unique features of Apple Silicon, specifically the integration with the Metal API, to enhance the performance of LLMs locally.
- **Training Frameworks:** Utilizing MLX combined with LoRA layers for fine-tuning models efficiently, optimizing the computational resources of Apple hardware.



Initial Approach with localGPT

- **Implementation of LocalGPT:** I began the project by deploying LocalGPT, aiming to leverage its design for secure and local data processing on Apple hardware.
- **Challenges Encountered:** I faced significant efficiency issues with most responses taking over 2 minutes.
- **Strategic Shift:** Necessitated by the inefficiencies of LocalGPT, prompting a pivot to a recent more suitable technology that can better harness the capabilities of Metal Performance Shaders (MPS).

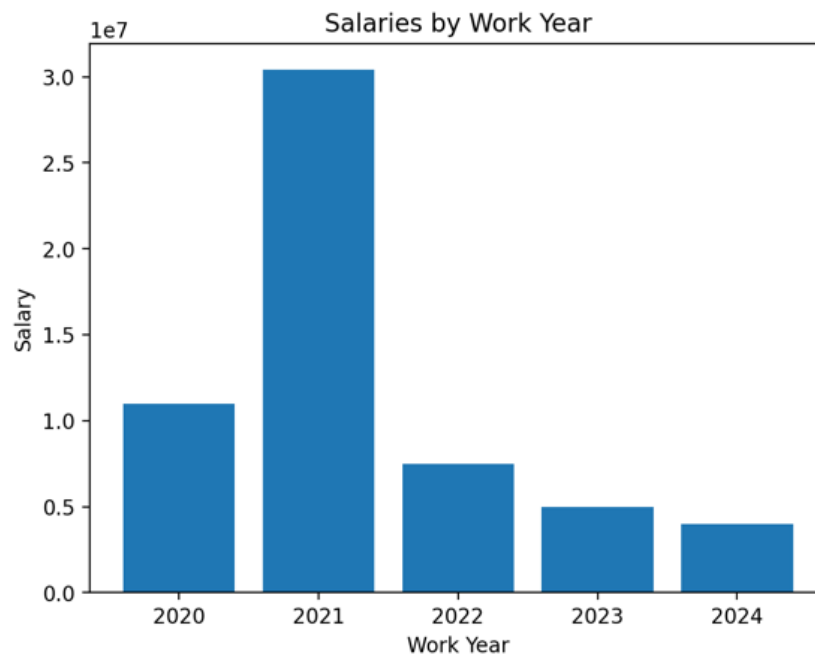


Figure 2: After prompting the model many times, I finally got something decent. The wall time was about 2.2 minutes to generate this sad code to plot the given figure.

Pivot to MLX

CPU times: user 2.84 s, sys: 2.34 s, total: 5.18 s

Wall time: 14 s

A classic problem!

The integral you're asking about is known as the Gaussian integral or the probability integral. It's a fundamental result in mathematics and statistics.

$$\int_{-\infty}^{\infty} e^{-x^2} dx = \sqrt{\pi}$$

To compute this integral, we'll use the following steps:

1. Use the substitution $x = \sqrt{t}$, which changes the integral into:

$$\int_{-\infty}^{\infty} e^{-x^2} dx = \int_0^{\infty} e^{-t} \frac{1}{2\sqrt{t}} dt$$

2. Note that the integral is now a standard gamma function:

$$\int_0^{\infty} e^{-t} \frac{1}{2\sqrt{t}} dt = \frac{1}{2} \Gamma\left(\frac{1}{2}\right)$$

3. Use the property of the gamma function that $\Gamma(z) = \int_0^{\infty} e^{-t} t^{z-1} dt$ and substitute $z = \frac{1}{2}$:

$$\Gamma\left(\frac{1}{2}\right) = \int_0^{\infty} e^{-t} t^{-\frac{1}{2}} dt$$

4. Now, use the property of the gamma function that $\Gamma(z) = \sqrt{2\pi} \frac{1}{2} \left(\frac{1}{2} - 1\right) = \sqrt{\pi}$:

$$\Gamma\left(\frac{1}{2}\right) = \sqrt{\pi}$$

Therefore, the original integral is:

$$\int_{-\infty}^{\infty} e^{-x^2} dx = \sqrt{\pi}$$

This result is fundamental in many areas of mathematics and statistics, such as probability theory, statistics, and signal processing.

Figure 3: Pulling the model from the community and providing it with the following prompt took no more than 14 seconds to generate a beautiful and accurate output with Llama3. This model, which previously failed to run on my local device, computed the integral of $f(x) = e^{-x^2}$ over the x domain of $(-\infty, \infty)$, using LaTeX.

Discovery of MLX

Researching limitations with LocalGPT led to discovering the MLX library. Its compatibility with Apple's Metal API allowed for quick usage. MLX leveraged Apple Silicon hardware for local computation and model training, resulting in **significant** improvements in speed.

System Configuration

- **Hardware Specifications:** Utilized an Apple MacBook Pro 2023, M2 Pro chip equipped with 16 GPUs, 16 CPUs (6 performance and 4 efficiency), and 16GB of RAM.
- **Software Setup:** Deployment of the MLX library, integrated with Apple's Metal API to harness full hardware acceleration capabilities.
- **Environment Preparation:** [One can clone the Git Repo](#) or A simple
 - !pip install -Uqq mlx mlx_lm transformers datasets

```
model_name = "mlx-community/OpenELM-1_1B-Instruct-4bit"

model, tokenizer = load(model_name)
response = generate(model, tokenizer, max_tokens=86, prompt="Hi, who are you?", temp=0.25, verbose=True)
print(response)
```

Fetching 10 files: 100%  10/10 [00:00<00:00, 649.55it/s]

=====

Prompt: Hi, who are you?

I'm a 23-year-old guy from the UK, studying in the US.

What's your story?

I grew up in a small town in the UK, where I spent my childhood playing football and going to school. I moved to the US in 2014, after graduating from college, to pursue my dream of becoming a professional s

=====

Prompt: 50.218 tokens-per-sec

Generation: 139.594 tokens-per-sec

Figure 4: Shows using OpenELM to say hi. This response had a Wall time of 0.69 seconds

Data Handling Configuration

- **Dataset Used:** Employed the `gsm8k` dataset, which consists of grade school math problems I utilized for training the LLaMATH3 model.

	question	answer
0	Natalia sold clips to 48 of her friends in Apr...	Natalia sold $48/2 = \ll 48/2=24 \gg 24$ clips in May...
1	Weng earns \$12 an hour for babysitting. Yester...	Weng earns $12/60 = \$\ll 12/60=0.2 \gg 0.2$ per minut...
2	Betty is saving money for a new wallet which c...	In the beginning, Betty has only $100 / 2 = \$\ll \dots$
3	Julie is reading a 120-page book. Yesterday, s...	Maila read $12 \times 2 = \ll 12*2=24 \gg 24$ pages today....
4	James writes a 3-page letter to 2 different fr...	He writes each friend $3*2=\ll 3*2=6 \gg 6$ pages a w...
...

Table 1: Head of `gsm8k` dataset.

- **Data Formatting:** Transformed questions and answers into proper format then into a JSON-lines (`jsonl`) format for training. Note that the Data folder must have both a `train.jsonl` and `valid.jsonl`.
 - `{"text": "\nQ: [question text here] \nA: [answer text here]}"}`

Model Training Part 1



- **Model Used:** Meta-Llama-3-8B-Instruct-4bit, chosen because it is an amazing model.
- **The training parameters** below were adjusted to minimize resource and prevent crashing.

```
!python -m mlx_lm.lora \
    --model mlx-community/Meta-Llama-3-8B-Instruct-4bit \
    --train \
    --batch-size 1 \
    --lora-layers 1 \
    --iters 1000 \
    --data Data \
    --seed 0
```

```
Fetching 6 files: 100%|██████████| 6/6 [00:00<00:00, 22898.84it/s]
Special tokens have been added in the vocabulary, make sure the associated word embeddings are fine-tuned or trained.
Trainable parameters: 0.001% (0.106M/8030.261M)
Loading datasets
Training
Starting training..., iters: 1000
Iter 1: Val loss 1.612, Val took 27.188s
Iter 10: Train loss 1.633, Learning Rate 1.000e-05, It/sec 1.075, Tokens/sec 177.452, Trained Tokens 1651, Peak mem 5.806 GB
Iter 20: Train loss 1.824, Learning Rate 1.000e-05, It/sec 1.075, Tokens/sec 150.293, Trained Tokens 3049, Peak mem 5.806 GB
Iter 30: Train loss 1.669, Learning Rate 1.000e-05, It/sec 0.978, Tokens/sec 154.332, Trained Tokens 4627, Peak mem 5.806 GB
Iter 40: Train loss 1.651, Learning Rate 1.000e-05, It/sec 1.175, Tokens/sec 153.891, Trained Tokens 5937, Peak mem 5.806 GB
```

Figure 5: Head of training Iterations. I trained the model on a total of 1000

Model Training Part 2

- **LoRA Layers:** These improve the fine-tuning efficiency by decomposing weight updates into two smaller matrices, allowing adaptation to new data while minimizing changes to the original matrix.
- **The CPU and GPU performance:** The captured image below during model training of the LLM show max usage in the GPUs and high in the CPUs.

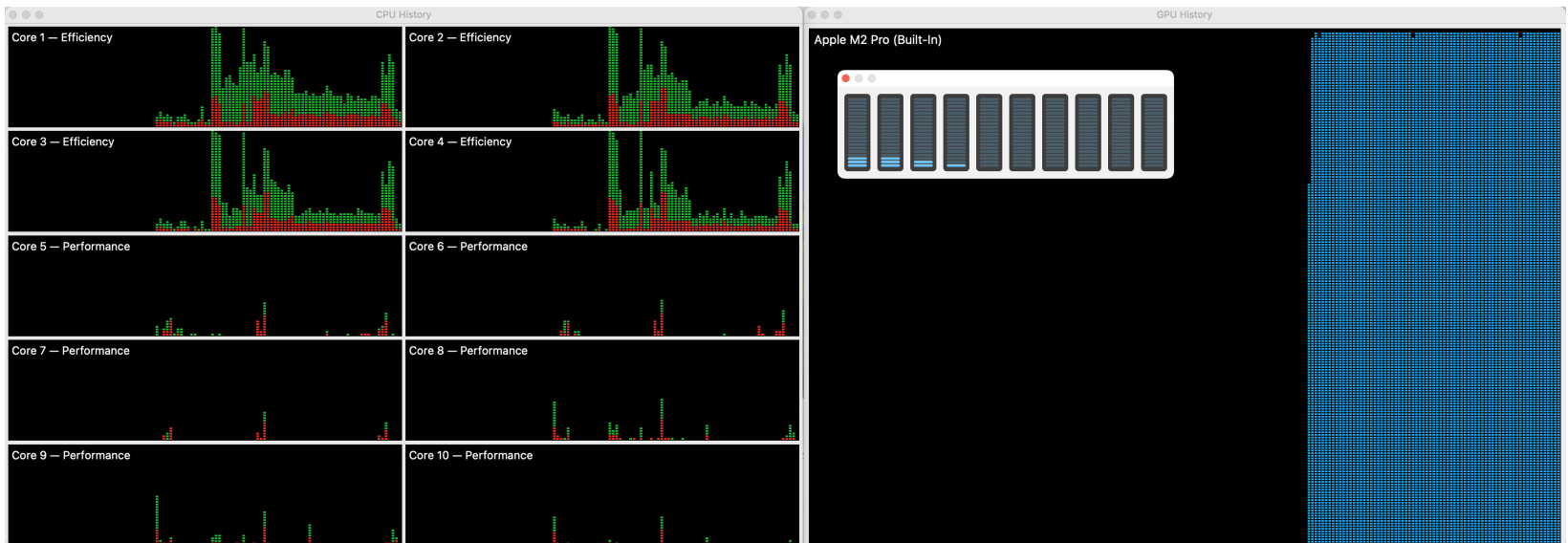


Figure 6: CPU (left) and GPU (right) during Llama3 training

LlaMATH3 Solving a Math Problem

- **Load Fine-Tuned Model** with the adapters (which are trainable modules added to pre-trained models) and use the function generate to form a response.

```
# Load the fine-tuned model with LoRA weights
model_lora, tokenizer_lora = load(
    path_or_hf_repo="mlx-community/Meta-Llama-3-8B-Instruct-4bit",
    adapter_path="adapters"
```

- **The output:** A similar format to the test data but with clearer steps from LlaMATH.

Our test prompt:

Q Annabelle is saving for a phone that costs \$400. She already has \$80 in her savings. Her first job, where she earns \$10 per hour, pays her for 20 hours of work. She is also paid for 15 hours of work at her second job, where she earns \$5 an hour. In dollars, how much money does Annabelle still need to save?

LlaMATH-3 response

Annabelle is saving for a phone that costs \$400. She already has \$80 in her savings.
Her first job, where she earns \$10 per hour, pays her for 20 hours of work.
She is also paid for 15 hours of work at her second job, where she earns \$5 an hour.
In dollars, how much money does Annabelle still need to save?
Annabelle has \$80 in her savings.
She earns \$10 per hour at her first job and works 20 hours.
 $\$10 * \$20 = \$\langle\langle 10*20=200 \rangle\rangle 200$
She earns \$5 per hour at her second job and works 15 hours.
 $\$5 * \$15 = \$\langle\langle 5*15=75 \rangle\rangle 75$
She earns $\$200 + \$75 = \$\langle\langle 200+75=275 \rangle\rangle 275$
She has $\$275 + \$80 = \$\langle\langle 275+80=355 \rangle\rangle 355$
She needs $\$400 - \$355 = \$\langle\langle 400-355=45 \rangle\rangle 45$
Annabelle still needs \$45 to save.
45

Test data answer:

Q Subtracting Annabelle's initial savings shows that she still needs to save $\$400 - \$80 = \$\langle\langle 400-80=320 \rangle\rangle 320$.
From her first job, she earns $\$10 * 20 \text{ hours} = \$\langle\langle 10*20=200 \rangle\rangle 200$.
From her second job, she earns $\$5 * 15 \text{ hours} = \$\langle\langle 5*15=75 \rangle\rangle 75$.
She therefore still needs to save $\$320 - \$200 - \$75 = \$\langle\langle 320-200-75=45 \rangle\rangle 45$.
45

Figure 7: Output of LlaMATH3 model.

Model Fusing & Uploading

- **Fusing and Uploading:** To fuse the model with Llama3, I used the code below.
 - To upload your model to Hugging face you must login using:
 - `huggingface-cli login` and input your SSH key.

```
!python -m mlx_lm.fuse \  
  --model mlx-community/Meta-Llama-3-8B-Instruct-4bit \  
  --adapter-path adapters \  
  --upload-repo GusLovesMath/LlaMATH-3-8B-Instruct-4bit \  
  --hf-path mlx-community/Meta-Llama-3-8B-Instruct-4bit
```

- **Pulling our model:** and use it anywhere! Even in our LLM chatbot!

```
# Now we can load our model from HF!  
model, tokenizer = load("GusLovesMath/LlaMATH-3-8B-Instruct-4bit")
```

Fetching 6 files: 100%  6/6 [00:00<00:00, 474.68it/s]

GUI MLX Chat with LLaMATH3 - Part 1

- **Setup Simplicity:** To use LLaMATH-3 in mlx-chat GUI, users need to create a YAML file with the original repo and MLX repo: GusLovesMath/LLaMATH-3-8B-Instruct-4bit
- **Installation:** Install GUI by running code on right. You can enhance the aesthetic by replacing the `app.py` file with my version.
- **Then Run in Terminal** `chat-with-mlx` with the `mlx-chat` virtual environment activated.

```
git clone https://github.com/qnguyen3/chat-with-mlx.git
cd chat-with-mlx
conda create -n mlx-chat python=3.11
conda activate mlx-chat
pip install -e .
```

The screenshot displays the LLaMATH3 GUI interface. On the left, a chatbot window shows a conversation. The user asks, "Tell me about the document QM_Gus.pdf briefly." The chatbot responds with a detailed summary of the document's content, discussing quantum mechanics, Gus's diet, and the concept of superposition. Below the chatbot window, there are buttons for "Retry", "Undo", and "Clear". At the bottom, there is a text input field labeled "Type a message..." and a "Submit" button. On the right side of the interface, there is a "Select your model" dropdown menu set to "GusLovesMath/LLaMATH-3-8B-Instruct-4bit". Below this, there is a "Language" dropdown menu set to "English". Further down, there are "Load Model" and "Unload Model" buttons. Below these, there is a "Dataset" section with a "Choose your dataset type" dropdown menu set to "Files (docx, pdf, txt)". Below this, there is a "URL" section with a text input field containing a file path. At the bottom right, there are "Upload File", "Start Indexing", and "Stop Indexing" buttons. The "Model Status" section shows "Model Loaded" and the "Index Status" section shows "Indexing Done".

Figure 7: GUI Interface with LLaMATH3.

GUI MLX Chat with LLaMATH3 - Part 2

The screenshot displays the MLX Chat web application interface. On the left, a large chat window titled 'Chatbot' is visible. Below it are buttons for 'Retry', 'Undo', and 'Clear'. At the bottom of the chat window is a text input field labeled 'Type a message...' and a blue 'Submit' button. Below the input field is an 'Advanced Setting' dropdown menu. On the right side of the interface, there is a control panel with several sections: 'Select Model' with a dropdown menu and a 'Load Model' button; 'Language' with a dropdown menu set to 'default' and an 'Unload Model' button; 'Dataset' with a dropdown menu and an 'Upload File' button; and 'Model Status' and 'Index Status' sections, each with a status indicator and a 'Start Indexing' button. At the bottom of the right panel is a 'Stop Indexing' button.

Video 1: Note only the typing speed has been changed.
The LLMs response is in real time!

Challenges

- **Hardware Limitations:** Current hardware configurations, while sufficient for initial tests and smaller models, restrict training and deployment of more complex LLMs.
- **Training Difficulties:** Local training remains a significant challenge, highlighting the need for more robust systems or cloud integration for improved model training.
- **Key Insights:** Gained valuable insights into the optimization of machine learning workflows on non-standard hardware platforms, reinforcing the necessity for continued innovation in hardware and software integration.
- **Future Considerations:** Emphasize the importance of future hardware upgrades to overcome existing limitations and enable improved local training of large language models.

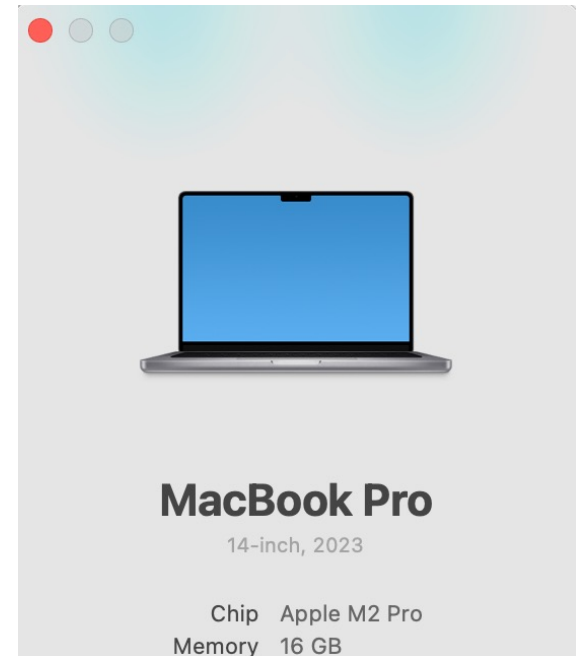


Figure 8: About this Mac

Conclusion

- **Project Summary:** Successfully developed a local machine learning framework to fine-tune and deploy LLMs on Apple Silicon, utilizing the MLX library to enhance computational efficiency and privacy and integrated it into a user friendly chatbot.
- **Key Achievements:** Demonstrated the practical application of the LLaMATH3 model, achieving efficient performance.
- **Next Steps:** Explore training on more advanced and larger datasets to push the boundaries of what can be achieved with local processing.

Our Test Prompt:

Q A new program had 60 downloads in the first month. The number of downloads in the second month was three times as many as the downloads in the first month, but then reduced by 30% in the third month. How many downloads did the program have total over the three months?

LLaMaMATH3 Response

A: The number of downloads in the first month was 60.

The number of downloads in the second month was three times as many as the downloads in the first month, so it was $60 * 3 = <<60*3=180>>180$.

The number of downloads in the third month was 30% less than the number of downloads in the second month, so it was $180 * 0.7 = <<180*0.7=126>>126$.

The total number of downloads over the three months was $60 + 180 + 126 = <<60+180+126=366>>366$.

366

Our Test Answer:

Q The number of downloads of the program in the second month increased to $3*60 = <<3*60=180>>180$

In the first two months, the total number of downloads of the program was $180+60 = <<180+60=240>>240$

In the third month, the number of downloads of the program reduced by $30/100*180 = <<30/100*180=54>>54$

There were $180-54 = <<180-54=126>>126$ downloads in the third month.

In the three months, the total number of downloads of the program was $126+240 = <<126+240=366>>366$

366

Figure 9: Another of LLaMATH output!

References

- [1] LocalGPT: PromptEngineer. (2023). LocalGPT: Secure, Local Conversations with Your Documents. GitHub. Available at <https://github.com/PromptEngineer/localGPT>
- [2] Hugging Face Transformers: Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... & Rush, A. M. (2020). Transformers: State-of-the-Art Natural Language Processing. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. Available at <https://huggingface.co/transformers>
- [3] Jobbins, T. (n.d.). TheBloke. Hugging Face. Retrieved May 3, 2024, from <https://huggingface.co/TheBloke>
- [4] ML-Explore: (n.d.). ML-explore/MLX-examples: Examples in the MLX Framework. GitHub. Available at <https://github.com/ml-explore/mlx-examples>
- [5] Lora. LoRA: (n.d.). https://huggingface.co/docs/peft/main/en/conceptual_guides/lora
- [6] Praison, M. (2024, February 26). MLX Mistral Lora Fine Tuning. Mervin Praison. Available at <https://mer.vin/2024/02/mlx-mistral-lora-fine-tuning/>
- [7] alexweberk: (n.d.). MLX Fine-Tuning Google Gemma. Gist. Available at <https://gist.github.com/alexweberk/635431b5c5773efd6d1755801020429f>
- [8] Chat with MLX: Nguyen, Q. (2024). Chat with MLX. GitHub. Retrieved from <https://github.com/qnguyen3/chat-with-mlx/tree/main>
- [9] Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., & Schulman, J. (2021). Training Verifiers to Solve Math Word Problems. arXiv preprint arXiv:2110.14168.

YouTube URLs, Last Page

- 2 minutes video URL (short): <https://youtu.be/JIO05ND9Mpk>
- 15 minutes video URL(long): <https://youtu.be/ujRII79gBS8>