Project Progress Report

By: Jeremy Flagg

Spring Semester 2025

1. Study Overview

This study focuses on:

- Training Merjek AI models on a GPU cluster.

2. Early Steps & Prompt Generation

The initial phase involved testing different LLM models for prompt generation and analyzing their outputs after database insertion.

3. Models Tested

Several models were tested for effectiveness and performance:

- Open-source models (e.g., LLama 3.1 8B, DeepSeek R1 1.5B, Mistral 7B v0.3)

4. GPU Cluster Specifications

Cluster Quota specifications:

- Max Jobs: 6
- Max Nodes: 3
- Max GPUs per Job: 4
- Max Runtime per Job: 48 hours

Training Progress:(1/24)

- **Initial meeting**

Training Progress: (2/7)

- Installation of Ollama and different open-source LLM models.
- Prompt generation and insertion into MySQL Workbench.

Training Progress:(2/14-2/28)

- Initial training/test practice, locally and in GPU Cluster,with Human Trafficking and Campus csv files.
- Migration to MongoDB Atlas/Compass

Training Progress: (3/7/25)

- Dataset: 2,000 documents (subset of 10K)
- Split: 80% train, 20% test
- Tested on 2 GPUs (1 node)
- Estimated training time: ~52 minutes for 1 epoch

Training Progress: (3/14/25)

- Created Merjek Github
- Prompt generation 5 hours per 1000 documents
- Meeting at library helping Md with MongoDB setup and prompt generation

- Continue generating ~8K prompts for the entire dataset of ~10K documents.
- Mistral 7B v0.3 is the model used for prompt generation. (LM Studio on my Windows setup)

- After generation, iterated through MongoDB collection to add prompts into arrays.

- Edit Slurm training script before executing within GPU cluster.

- Scaled training from 2,000 docs at 1 epoch to 10,000 docs at 3 epochs.

✅ Loaded 305835 valid prompts from the first 10,000 documents.

Training samples: 244668

Validation samples: 61167

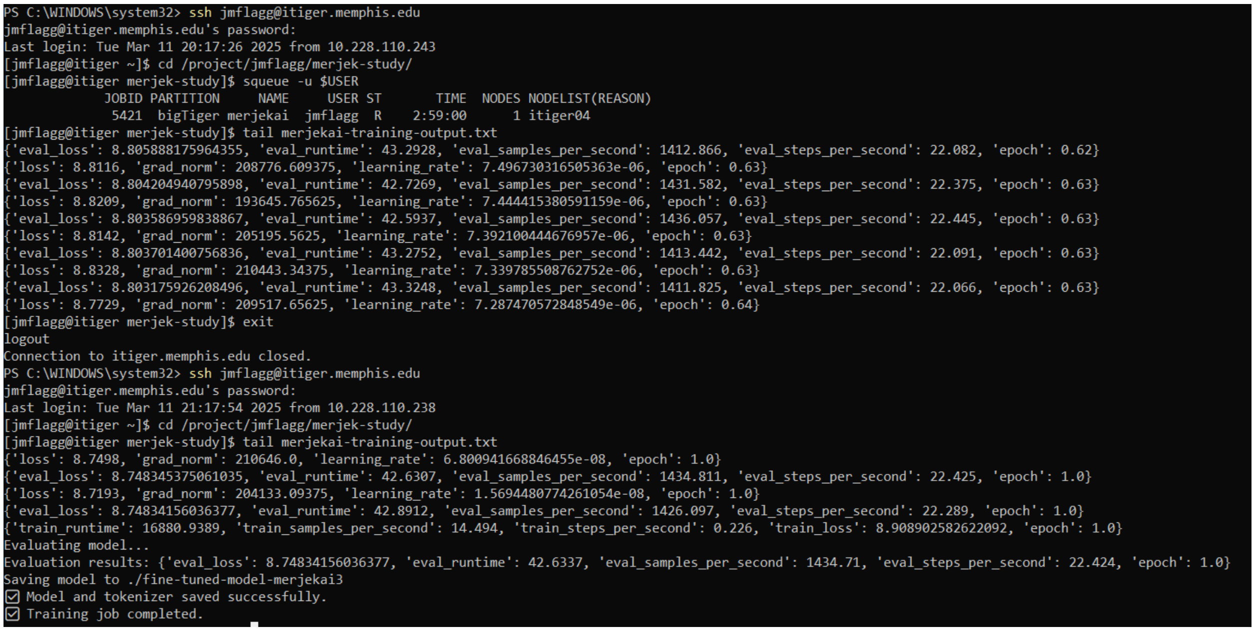
🖥️ Using device: cuda, Batch size: 16

GPU #: 4

Estimated train time for 1 epoch: 4 hours 41 minutes

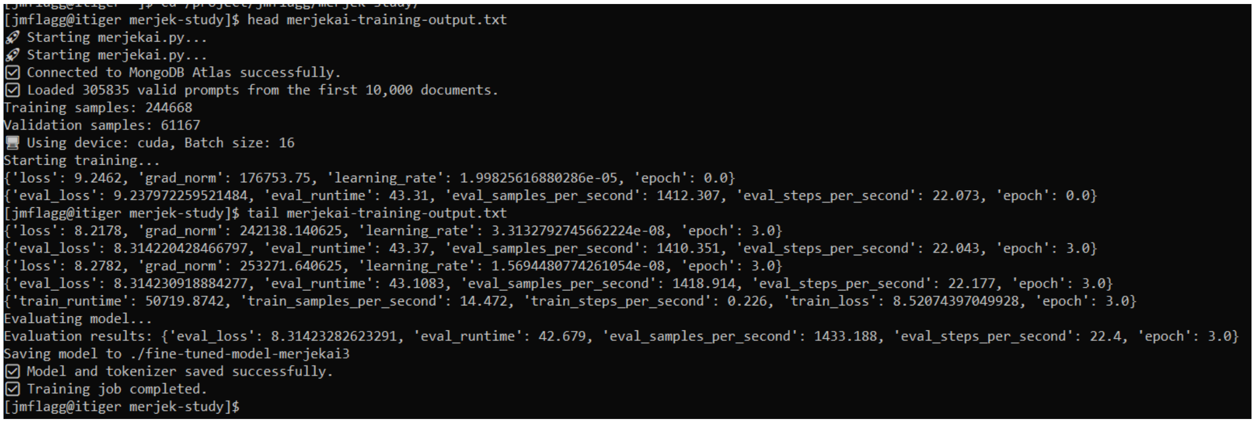
Estimated train time for 3 epochs: 14 hours 4 minutes

View inside cluster after 1 epoch for 10K docs:



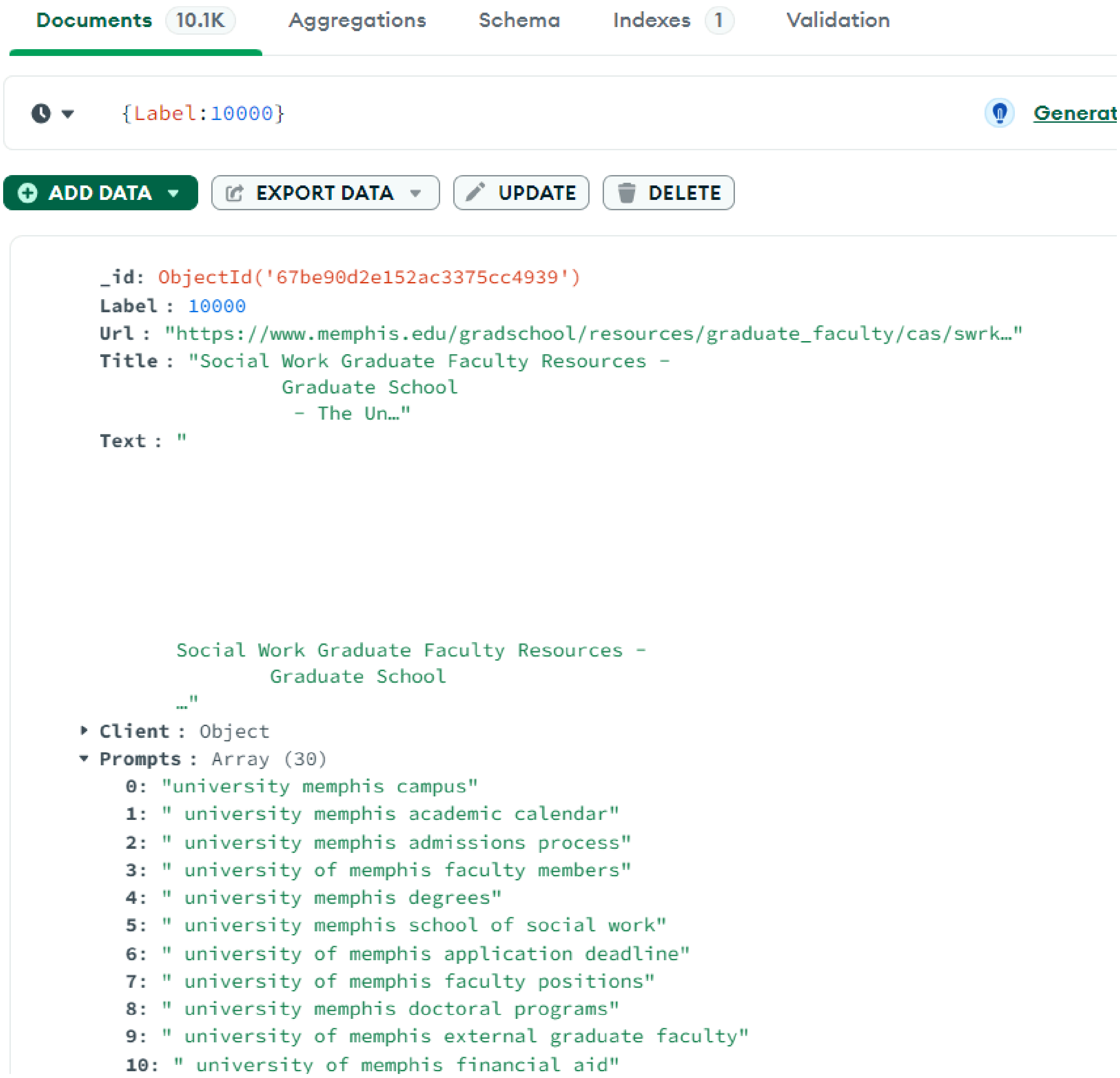
```
PS C:\WINDOWS\system32> ssh jmflagg@itiger.memphis.edu
jmflagg@itiger.memphis.edu's password:
Last login: Tue Mar 11 20:17:26 2025 from 10.228.110.243
[jmflagg@itiger ~]$ cd /project/jmflagg/merjek-study/
[jmflagg@itiger merjek-study]$ squeue -u $USER
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
              5421  bigTiger merjekai  jmflagg  R    2:59:00      1 itiger04
[jmflagg@itiger merjek-study]$ tail merjekai-training-output.txt
{'eval_loss': 8.805888175964355, 'eval_runtime': 43.2928, 'eval_samples_per_second': 1412.866, 'eval_steps_per_second': 22.082, 'epoch': 0.62}
{'loss': 8.8116, 'grad_norm': 208776.609375, 'learning_rate': 7.496730316505363e-06, 'epoch': 0.63}
{'eval_loss': 8.804204940795898, 'eval_runtime': 42.7269, 'eval_samples_per_second': 1431.582, 'eval_steps_per_second': 22.375, 'epoch': 0.63}
{'loss': 8.8209, 'grad_norm': 193645.765625, 'learning_rate': 7.444415380591159e-06, 'epoch': 0.63}
{'eval_loss': 8.803586959838867, 'eval_runtime': 42.5937, 'eval_samples_per_second': 1436.057, 'eval_steps_per_second': 22.445, 'epoch': 0.63}
{'loss': 8.8142, 'grad_norm': 205195.5625, 'learning_rate': 7.392100444676957e-06, 'epoch': 0.63}
{'eval_loss': 8.803701400756836, 'eval_runtime': 43.2752, 'eval_samples_per_second': 1413.442, 'eval_steps_per_second': 22.091, 'epoch': 0.63}
{'loss': 8.8328, 'grad_norm': 210443.34375, 'learning_rate': 7.339785508762752e-06, 'epoch': 0.63}
{'eval_loss': 8.803175926208496, 'eval_runtime': 43.3248, 'eval_samples_per_second': 1411.825, 'eval_steps_per_second': 22.066, 'epoch': 0.63}
{'loss': 8.7729, 'grad_norm': 209517.65625, 'learning_rate': 7.287470572848549e-06, 'epoch': 0.64}
[jmflagg@itiger merjek-study]$ exit
logout
Connection to itiger.memphis.edu closed.
PS C:\WINDOWS\system32> ssh jmflagg@itiger.memphis.edu
jmflagg@itiger.memphis.edu's password:
Last login: Tue Mar 11 21:17:54 2025 from 10.228.110.238
[jmflagg@itiger ~]$ cd /project/jmflagg/merjek-study/
[jmflagg@itiger merjek-study]$ tail merjekai-training-output.txt
{'loss': 8.7498, 'grad_norm': 210646.0, 'learning_rate': 6.800941668846455e-08, 'epoch': 1.0}
{'eval_loss': 8.748345375061035, 'eval_runtime': 42.6307, 'eval_samples_per_second': 1434.811, 'eval_steps_per_second': 22.425, 'epoch': 1.0}
{'loss': 8.7193, 'grad_norm': 204133.09375, 'learning_rate': 1.5694480774261054e-08, 'epoch': 1.0}
{'eval_loss': 8.74834156036377, 'eval_runtime': 42.8912, 'eval_samples_per_second': 1426.097, 'eval_steps_per_second': 22.289, 'epoch': 1.0}
{'train_runtime': 16880.9389, 'train_samples_per_second': 14.494, 'train_steps_per_second': 0.226, 'train_loss': 8.908902582622092, 'epoch': 1.0}
Evaluating model...
Evaluation results: {'eval_loss': 8.74834156036377, 'eval_runtime': 42.6337, 'eval_samples_per_second': 1434.71, 'eval_steps_per_second': 22.424, 'epoch': 1.0}
Saving model to ./fine-tuned-model-merjekai3
☑ Model and tokenizer saved successfully.
☑ Training job completed.
```

View inside cluster after 3 epochs for 10K:



```
[jmflagg@itiger ~]$ cd /project/jmflagg/merjek-study/
[jmflagg@itiger merjek-study]$ head merjekai-training-output.txt
🚀 Starting merjekai.py...
🚀 Starting merjekai.py...
☑ Connected to MongoDB Atlas successfully.
☑ Loaded 305835 valid prompts from the first 10,000 documents.
Training samples: 244668
Validation samples: 61167
💻 Using device: cuda, Batch size: 16
Starting training...
{'loss': 9.2462, 'grad_norm': 176753.75, 'learning_rate': 1.99825616880286e-05, 'epoch': 0.0}
{'eval_loss': 9.237972259521484, 'eval_runtime': 43.31, 'eval_samples_per_second': 1412.307, 'eval_steps_per_second': 22.073, 'epoch': 0.0}
[jmflagg@itiger merjek-study]$ tail merjekai-training-output.txt
{'loss': 8.2178, 'grad_norm': 242138.140625, 'learning_rate': 3.3132792745662224e-08, 'epoch': 3.0}
{'eval_loss': 8.314220428466797, 'eval_runtime': 43.37, 'eval_samples_per_second': 1410.351, 'eval_steps_per_second': 22.043, 'epoch': 3.0}
{'loss': 8.2782, 'grad_norm': 253271.640625, 'learning_rate': 1.5694480774261054e-08, 'epoch': 3.0}
{'eval_loss': 8.314230918884277, 'eval_runtime': 43.1083, 'eval_samples_per_second': 1418.914, 'eval_steps_per_second': 22.177, 'epoch': 3.0}
{'train_runtime': 50719.8742, 'train_samples_per_second': 14.472, 'train_steps_per_second': 0.226, 'train_loss': 8.52074397049928, 'epoch': 3.0}
Evaluating model...
Evaluation results: {'eval_loss': 8.31423282623291, 'eval_runtime': 42.679, 'eval_samples_per_second': 1433.188, 'eval_steps_per_second': 22.4, 'epoch': 3.0}
Saving model to ./fine-tuned-model-merjekai3
☑ Model and tokenizer saved successfully.
☑ Training job completed.
[jmflagg@itiger merjek-study]$
```

View inside MongoDB Compass:



```
{Label:10000}
```

```
_id: ObjectId('67be90d2e152ac3375cc4939')
Label : 10000
Url : "https://www.memphis.edu/gradschool/resources/graduate_faculty/cas/swrk…"
Title : "Social Work Graduate Faculty Resources -
                Graduate School
                 - The Un…"
Text : "






       Social Work Graduate Faculty Resources -
               Graduate School
       …"
▸ Client : Object
▾ Prompts : Array (30)
    0: "university memphis campus"
    1: " university memphis academic calendar"
    2: " university memphis admissions process"
    3: " university of memphis faculty members"
    4: " university memphis degrees"
    5: " university memphis school of social work"
    6: " university of memphis application deadline"
    7: " university of memphis faculty positions"
    8: " university memphis doctoral programs"
    9: " university of memphis external graduate faculty"
    10: " university of memphis financial aid"
```

Training Progress: (3/21/25 - 3/28/25)

Goal: Find a way to utilize the GPU cluster for prompt generation

1. Initial Attempt with vLLM

- **Tried using vLLM for running LLaMA 3.1 8B.**
- **Faced challenges and decided to move on to other methods.**

---

2. Transformers Library Approach

- **Attempted to use the Transformers library and download LLaMA 3.1 8B from HuggingFace.**
- **Performance was poor, comparable to 1B models.**
- **Assumed Ollama optimizes models behind the scenes for better performance.**

---

3. Dockerized Ollama Installation

- **Installed a Dockerized version of Ollama on the cluster using Podman (compatible with Docker).**
- **Downloaded and tested the LLaMA 3.2B model.**
- **The model worked but encountered two major issues:**
    - **No External Access: Unable to access Ollama from outside the container, even though ports were open and listening.**
    - **CPU-Only Inference: Without external access, couldn’t create a Slurm script for GPU usage, resulting in CPU-only inference.**

---

4. Native Ollama Installation

- **Installed Ollama natively by downloading and extracting the binary to a directory.**

- **Still faced CPU-only inference since GPUs on the cluster are only accessible through Slurm.**

---

5. Understanding Cluster Architecture

- **Gained better insight into the cluster's architecture:**
  - **Head Node: Where users log in, but no GPUs are available.**
  - **Worker Nodes: GPUs are only available on these nodes through Slurm jobs.**
- **Confirmed that Ollama needs to run on worker nodes for GPU access.**

---

6. Issues with Docker GPU Pass-Through

- **Directly using Docker containers for Ollama with GPU pass-through was unsuccessful.**
- **The cluster's GPU access is restricted, and Docker doesn't support direct GPU usage in this environment.**

---

7. Solution: Using Apptainer

- **Identified Apptainer (formerly Singularity) as the only functional solution for GPU access on the cluster.**
- **Apptainer is installed on the cluster and supports GPU pass-through.**
- **This method allows Ollama to run with GPU acceleration via Slurm jobs.**

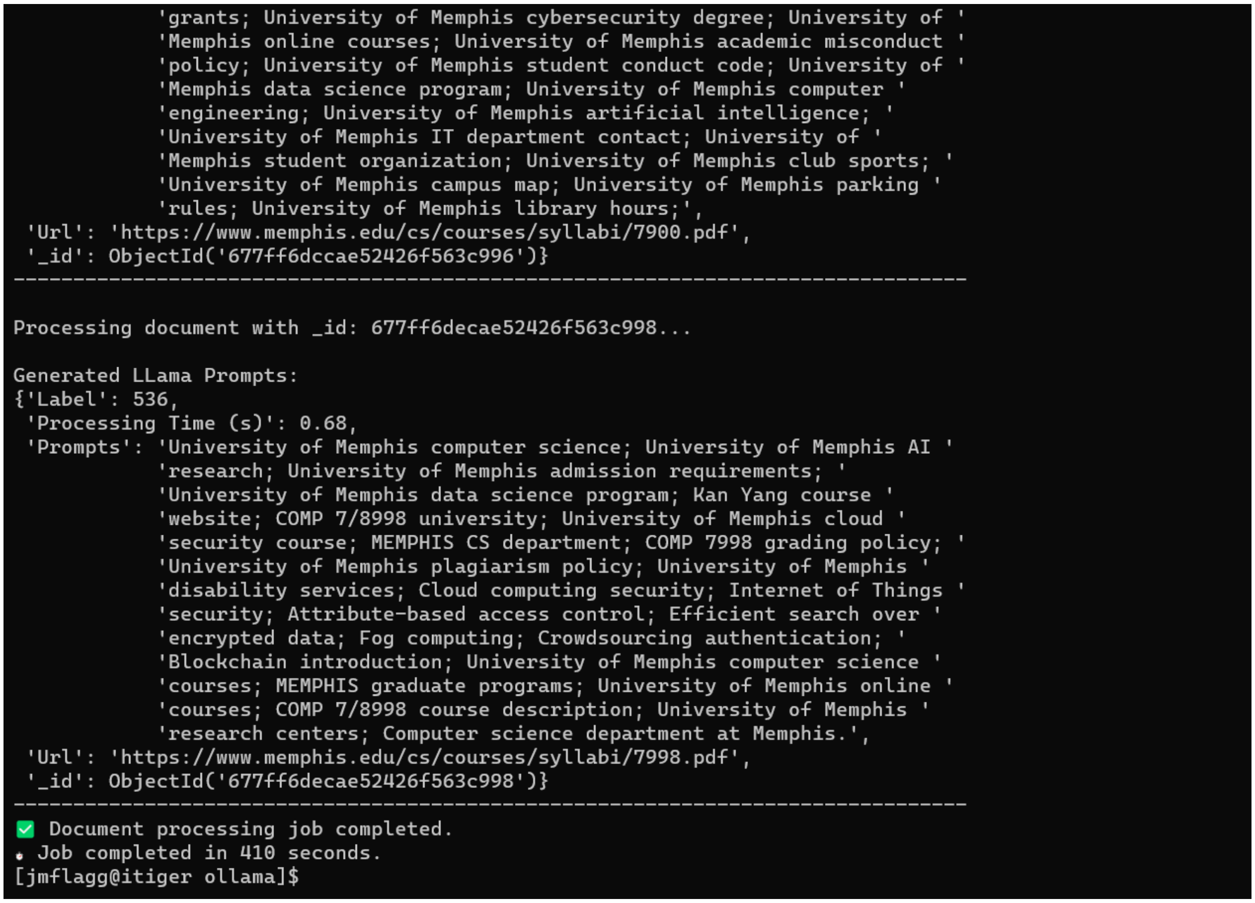
Prompt Generation Analysis

📂 Database: `crawled_cs_pages2`

📄 Total Documents: ~548 docs

⏳ Total Time: 410 seconds | 6m 50s

Model Tested: Llama 3.2:3b



```
             'grants; University of Memphis cybersecurity degree; University of '
             'Memphis online courses; University of Memphis academic misconduct '
             'policy; University of Memphis student conduct code; University of '
             'Memphis data science program; University of Memphis computer '
             'engineering; University of Memphis artificial intelligence; '
             'University of Memphis IT department contact; University of '
             'Memphis student organization; University of Memphis club sports; '
             'University of Memphis campus map; University of Memphis parking '
             'rules; University of Memphis library hours;',
 'Url': 'https://www.memphis.edu/cs/courses/syllabi/7900.pdf',
 '_id': ObjectId('677ff6dccae52426f563c996')}
--------------------------------------------------------------------------------

Processing document with _id: 677ff6decae52426f563c998...

Generated LLama Prompts:
{'Label': 536,
 'Processing Time (s)': 0.68,
 'Prompts': 'University of Memphis computer science; University of Memphis AI '
             'research; University of Memphis admission requirements; '
             'University of Memphis data science program; Kan Yang course '
             'website; COMP 7/8998 university; University of Memphis cloud '
             'security course; MEMPHIS CS department; COMP 7998 grading policy; '
             'University of Memphis plagiarism policy; University of Memphis '
             'disability services; Cloud computing security; Internet of Things '
             'security; Attribute-based access control; Efficient search over '
             'encrypted data; Fog computing; Crowdsourcing authentication; '
             'Blockchain introduction; University of Memphis computer science '
             'courses; MEMPHIS graduate programs; University of Memphis online '
             'courses; COMP 7/8998 course description; University of Memphis '
             'research centers; Computer science department at Memphis.',
 'Url': 'https://www.memphis.edu/cs/courses/syllabi/7998.pdf',
 '_id': ObjectId('677ff6decae52426f563c998')}
--------------------------------------------------------------------------------
✅ Document processing job completed.
⏱ Job completed in 410 seconds.
[jmflagg@itiger ollama]$
```

🚀 Ollama Setup and GPU Usage on Cluster

🔎 Check for Apptainer Installation

```
which apptainer
/usr/bin/apptainer
```

📥 Download and Convert Ollama Docker Image to SIF

```
apptainer pull docker://ollama/ollama:latest
```

- Converts Docker image to SIF format for Apptainer compatibility.

📂 Verify SIF File

```
ls -lh *.sif
# Example Output:
# -rwxr-xr-x 1 jmflagg users 1.7G Mar 26 18:44 ollama_latest.sif
```

📊 Cluster Management

🖥️ Check Node Status

```
sinfo
```

- Provides general info about available nodes.

📋 Detailed Node Information

```
sinfo -N -o "%N %P %C %G %T %M %E"
```

- Displays node-specific details like CPU, GPU availability, state, and errors.

🏁 Running Jobs on Specific Nodes

🚪 SSH into Cluster

```
ssh jmflagg@itiger.memphis.edu
```

🏢 Navigate to Project Directory

```
cd /project/jmflagg/ollama
```

🚀 Submit a Job to Specific Node

```
srun --partition=bigTiger --nodelist=itiger03 --gres=gpu:1 --mem=64G --time=1:00:00 --pty bash
```

- Requests a GPU node for one hour with 64GB memory.

📜 Check Running Jobs

```
squeue -u $USER
```

🖼️ Monitor GPU Status

```
nvidia-smi
```

- Displays real-time GPU usage and memory allocation.

📤 Submit Job with SBATCH to specific node

```
sbatch --nodelist=itiger03 app-job.sh
```

🛠️ Install Ollama Natively

📥 Download Ollama Binary

```
curl -L https://ollama.com/download/ollama-linux-amd64.tgz -o
ollama-linux-amd64.tgz
```

📁 Extract and Install

```
mkdir -p ~/ollama
```

```
tar -xzf ollama-linux-amd64.tgz -C ~/ollama
```

🛣️ Add Ollama to Path

```
export PATH=$HOME/ollama/bin:$PATH
```

```
# Confirm installation

ollama --version
```