Week 1 Required Task - Research Result

**GPU infra**

1. [AWS Sagemaker](https://aws.amazon.com/sagemaker/pricing/?nc=sn&loc=3&refid=ft_card)

Amazon offers various GPUs in its instances. Training and script will have background execution enabled. Free trial includes m4.xlarge and m5.xlarge with 4 virtual CPUs. Our targeted NVIDIA A100 is offered in ml.p4d.24xlarge and ml.p4de.24xlarge. However, these two products only offer 8GPUs option, which means it will be very expensive.

1. [Google Colab](https://colab.research.google.com/signup/pricing?utm_source=notebook_settings&utm_medium=link&utm_campaign=premium_gpu_selector)

Free version of Colab offers very limited access to GPUs. Usage limits are much lower than they are in paid versions of Colab. Paid versions of Colab (Pro, Pro+) offer premium GPUs subject to availability. The types of GPUs available will vary over time, so we may not be able to use NVIDIA A100 any time we want. In addition, background execution is disabled. We need to set the environment everyday and code execution including training is only supported for up to 24 hours.

1. [Columbia University - High Performance Computing (HPC)](https://www.cuit.columbia.edu/shared-research-computing-facility)

The HPC provides various clusters having lots of GPU resources. There are four options: purchase, rent, free, education. “Researchers, including graduate students may use the system on a low-priority, as-available basis.” We may send an application to request the HPC resources. The rate is to be determined for the other options, purchase and rent.

| Company | Pros | Cons | Pricing |
| --- | --- | --- | --- |
| AWS sagemaker | Background execution, various cluster configurations | Expensive, possibly redundant computing resources | Ml.p4d.24 xlarge:  $37 / hour |
| Google Colab | Cheap, easy to use | No background execution, unstable GPU availability | Pro: $9.99 / month  Pro+: $49.99 / month |
| CU HPC | Free | Application needed, unstable GPU availability | Free |

**Vector Database**

| Name | Pros | Cons | Notes |
| --- | --- | --- | --- |
| ChromaDB | - Easy application and integration with intuitive name in python coding  - hands-on experience  - embedding model can be customized | - ​​may struggle with large scale dataset  - limited customization for data modeling and querying(may not be the main concern) | comparing five models:  <https://medium.com/@3minutesnapshot/top-5-vector-databases-and-when-to-use-them-6c321e8ccc33>  Chroma with Langchain:  <https://python.langchain.com/docs/integrations/vectorstores/chroma> |
| Milvus | - optimized for similarity search with advanced indexing techniques to speed up search operations  - allow large datasets and high query loads  - can leverage GPU resources, leading to significant speed improvements  - specialized in similarity search and handle embedding vectors converted from unstructured data  (built to pair with FAISS) | - setting up and configuring Milvus can be complex  - may need to fine-tune Milvus, select appropriate indexing methods, and configure hardware resources like GPUs. This optimization process can be challenging and time-consuming.  - Data is often assumed to be static once fed into existing systems, complicating processing for dynamic data | Basic Introduction:  <https://milvus.io/blog/scalable-and-blazing-fast-similarity-search-with-milvus-vector-database.md>  Milvus with Langchain:  <https://milvus.io/docs/integrate_with_langchain.md> |
| pgvector | easy to integrate with PostgreSQL | - most cloud offerings of PostgreSQL have not yet integrated pgvector  - not that convenient if we are not going to use PostgreSQL | github link:  <https://github.com/pgvector/pgvector> |
| Qdrant | - fast and reliable even under high load  - supports metadata filtering like ChromaDB, and integrates into technologies like Cohere (embeddings), LangChain, and LlamaIndex | - open-source vector database written in Rust  - not entirely free | github link:  ​<https://github.com/qdrant/qdrant> |
| Pinecone | - offers blazing-fast search capabilities, allowing users to retrieve similar vectors in real-time, making it well-suited for content-based searching  - excellent choice to deal with vast amount of data due to its architecture design  - relatively easy to apply with automatically indexes vectors | - might lack some advanced querying capabilities that certain projects require  - **not open source**  **- has free version :**  search through roughly a million vectors in around 100ms, or through 100K vectors in around 20ms | pinecone documentation:  <https://docs.pinecone.io/> |

**FAISS Nearest Neighbor Algorithms**

FAISS (Facebook AI Similarity Search) is a library that allows developers to quickly search for embeddings of multimedia documents that are similar to each other. It also contains supporting code for evaluation and parameter tuning.

Documentation: <https://github.com/facebookresearch/faiss/wiki>

| Index | Introduction | Notes |
| --- | --- | --- |
| FLAT | If FLAT index is used, the vectors are stored in an array of float/binary data without any compression. during searching vectors, all indexed vectors are decoded sequentially and compared to the query vectors.  FLAT index provides 100% query recall rate. Compared to other indexes, it is the most efficient indexing method when the number of queries is small | <https://milvus.io/docs/v1.1.1/index.md#FLAT> |
| IVF  (Inverted File Index) | IVF (Inverted File) is an index type based on quantization. It divides the points in space into nlist units by clustering method. during searching vectors, it compares the distances between the target vector and the center of all the units, and then select the nprobe nearest unit. Then, it compares all the vectors in these selected cells to get the final result. | <https://milvus.io/docs/v1.1.1/index.md#:~:text=IVF%20(Inverted%20File)%20is%20an,select%20the%20nprobe%20nearest%20unit>. |
| HNSW  (Hierarchical Navigable Small World) | HNSW (Hierarchical Small World Graph) is a graph-based indexing algorithm. It builds a multi-layer navigation structure for an image according to certain rules. In this structure, the upper layers are more sparse and the distances between nodes are farther; the lower layers are denser and the distances between nodes are closer. The search starts from the uppermost layer, finds the node closest to the target in this layer, and then enters the next layer to begin another search. After multiple iterations, it can quickly approach the target position. | <https://supabase.com/blog/increase-performance-pgvector-hnsw#how-does-hnsw-work> |

## **Vector Libraries**

Vector libraries store vector embeddings in in-memory indexes, in order to perform similarity search. Most vector libraries share the following characteristics:

1. they store vectors only,
2. index data is immutable,
3. query during import limitation

## **Vector Databases**

One of the core features that set vector databases apart from libraries is the ability to store and update your data. Vector databases have full CRUD (create, read, update, and delete) support that solves the limitations of a vector library. Additionally, databases are more focused on enterprise-level [production deployments](https://www.youtube.com/watch?v=gXPuhyM11_k).

**LLM**

**Trending:**

| Name | Introduction | Pros | Cons | Notes |
| --- | --- | --- | --- | --- |
| **GPT**  GPT-4 |  | - Highest quality response | - Expensive | - Not Open Source |
| **T5**  FLAN-T5  FLAN-UL2 | The T5 series of models open-sourced by Google is available in various sizes of parameters.  A Sequence to Sequence model that follows an Encoder and Decoder architecture as opposed to the GPT family of models that are decoder models only. | - A great model to fine-tune at a relatively low cost  - Wide variety of model sizes  - Well suited for translation and summarization tasks | - Requires a large amount of computational power and memory to run  - May generate unreliable results when it is presented with new or unusual inputs  - Long training time | Hugging  Face:  <https://huggingface.co/google/flan-ul2>  <https://huggingface.co/google/flan-t5-large>  Others:  <https://exemplary.ai/blog/flan-t5> |
| **Falcon**  Falcon-7B  Falcon-180B | Falcon-7B is a 7B parameters causal decoder-only model built by TII and trained on 1,500B tokens of RefinedWeb enhanced with curated corpora. It is made available under the Apache 2.0 license. | - Strong conversational capabilities and performance optimization, well-suited for interactive and engaging conversational experiences | - Less parameters than GPT so less complexity and capacity of the models to capture and generate human-like text | Hugging  Face:  <https://huggingface.co/tiiuae/falcon-7b> |
| **LLAMA**  Llama-2-7b | Developed by Meta Llama 2 is an auto-regressive language model that uses an optimized transformer architecture. The tuned versions use supervised fine-tuning (SFT) and reinforcement learning with human feedback (RLHF) to align to human preferences for helpfulness and safety. | - LLaMA 2 can generate high-quality texts in a matter of seconds. It also uses less computational resources than other LLMs of similar size and complexity. | - GPT-4 has more parameters, more data, more context length, and more modalities than LLaMA 2. This gives it an edge in terms of accuracy, complexity, diversity, and generality of its outputs. | Hugging Face:  <https://huggingface.co/meta-llama/Llama-2-7b> |

**Financial Models on Hugging Face:**

| Name | Introduction | Notes |
| --- | --- | --- |
| ahmedrachid/FinancialBERT-Sentiment-Analysis | FinancialBERT is a BERT model pre-trained on a large corpora of financial texts. The purpose is to enhance financial NLP research and practice in financial domain, hoping that financial practitioners and researchers can benefit from this model without the necessity of the significant computational resources required to train the model.  The model was fine-tuned for Sentiment Analysis task on Financial PhraseBank dataset. Experiments show that this model outperforms the general BERT and other financial domain-specific models. | <https://huggingface.co/ahmedrachid/FinancialBERT-Sentiment-Analysis> |
| human-centered-summarization/financial-summarization-pegasus | This model was fine-tuned on a novel financial news dataset, which consists of 2K articles from Bloomberg, on topics such as stock, markets, currencies, rate and cryptocurrencies.  It is based on the PEGASUS model and in particular PEGASUS fine-tuned on the Extreme Summarization (XSum) dataset: google/pegasus-xsum model. | <https://huggingface.co/human-centered-summarization/financial-summarization-pegasus> |

**UI**

Web-app: python with flask

**KPls**

**LangChain**

[LangChain Q&A Bot-Comparative Study of Oil and Gas Company KPIs-.ipynb](https://colab.research.google.com/drive/17WzEftJczZCYfq-TJducj1H4PPPydPLP)

Reference:

<https://python.langchain.com/docs/>

<https://python.langchain.com/docs/use_cases/question_answering/>

<https://liaokong.gitbook.io/llm-kai-fa-jiao-cheng/#text-splitters-wen-ben-fen-ge>