DEVELOPMENT OF AN LLM FOR DUNE

ALEENA RAFIQUE
For the DUNE collaboration

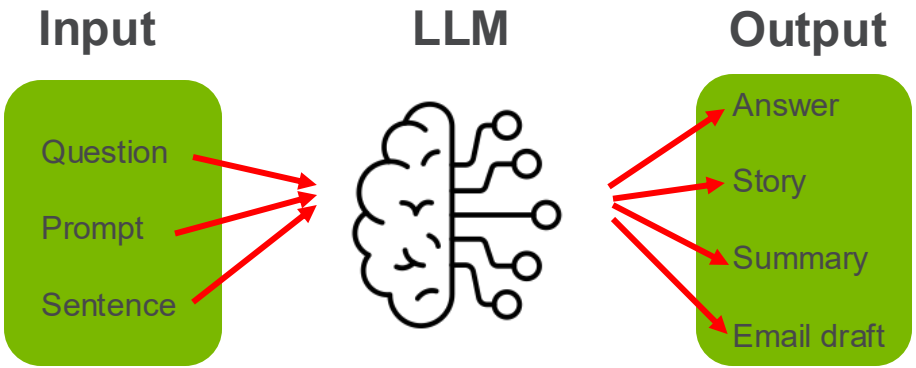
Lepton-Photon 2025

U.S. DEPARTMENT OF ENERGY Argonne National Laboratory is a U.S. Department of Energy laboratory managed by UChicago Argonne, LLC.

08/28/2025

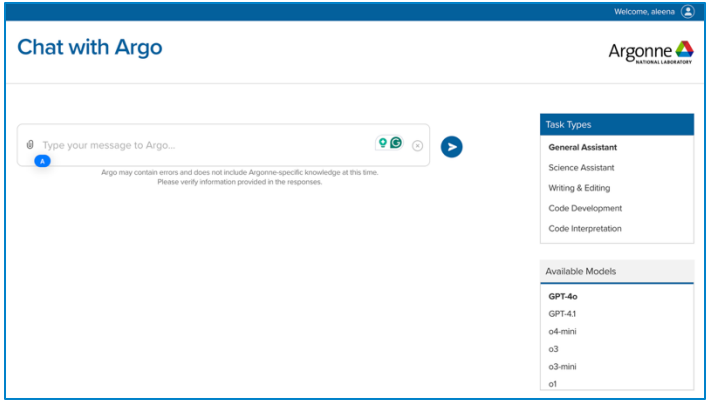
LARGE LANGUAGE MODELS (LLMS)

- **What is an LLM?**
  - A **Large Language Model** is an AI system trained to understand and generate human-like text
  - Examples: **ChatGPT (OpenAI), Claude (Anthropic), Gemini (Google)** etc
- **How do they work?**
  - Trained on **huge amounts of text data** (books, websites, conversations)
  - Use deep learning, especially **transformer models**, to predict the next word in a sentence
  - Can perform tasks like:
    - Answering questions
    - Summarizing text
    - Writing code
    - Translating languages
    - Generating creative content



LOCAL LLMS AT ANL AND FNAL

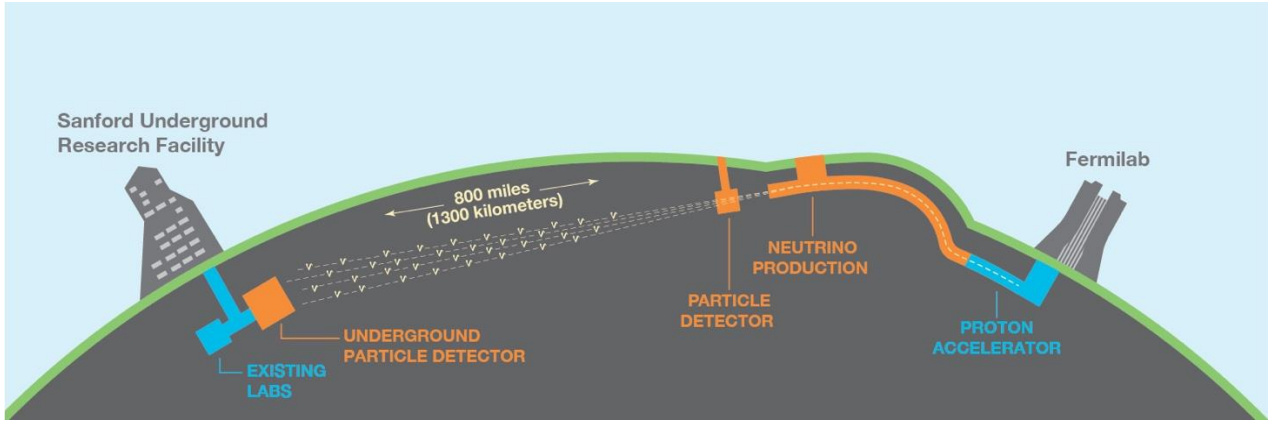
- **Argo at Argonne**
  - Internal **chatbot platform** hosted at Argonne
  - **Security-first**: does not store or share user data
  - **Text-only** capabilities (no image generation yet)
  - **Argo Gateway**: Custom API enabling direct LLM access
  - Supports multiple providers: such as OpenAI, Anthropic, Google



- **Ollama at Fermilab**
  - Runs locally on FNAL hardware (A100 GPU node)
  - Designed for **LLM development and experimentation**
  - **Data-secure**: no data is shared or stored externally
  - Uses **open-weight models**: Primarily **Qwen** and **LLaMA**
  - Not among one the latest state-of-the-art LLMs

DUNE

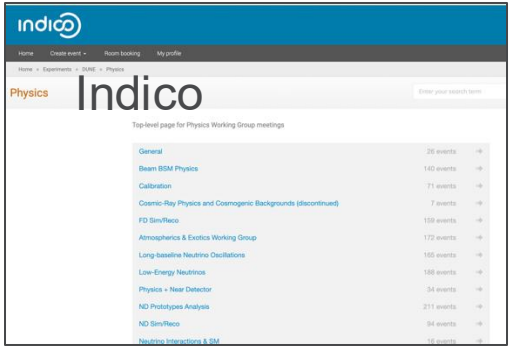
- The Deep Underground Neutrino Experiment (DUNE) is a future accelerator-based multi-detector long-baseline neutrino oscillations experiment.

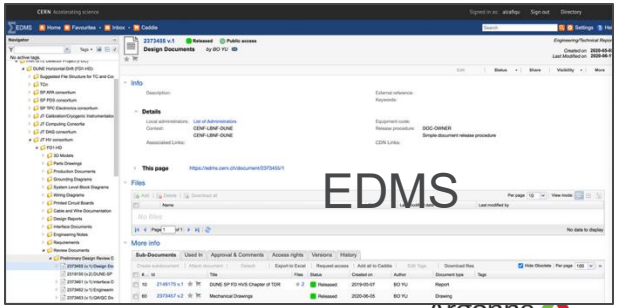


- Major DUNE goals:
  - Neutrino Oscillation Physics, Nucleon Decay, Supernova burst physics & astrophysics
- DUNE collaboration: 1830 people, 250 institutions, 39 countries
- The detector will collect data late 2020s – early 2030s

LLM FOR DUNE DOCUMENTATION

- DUNE relies on multiple documentation platforms, each of them already have tens of thousands of documents :
  - DocDB (Fermilab hosted)
  - EDMS (CERN Hosted)
  - Indico (hosted by various institutions)
- Extracting relevant information across these systems is **non-trivial**
  - Especially Indico, which lacks robust search capabilities
- Developing an LLM (called DUNE-GPT) for DUNE to combine these databases into one, in a form that when a collaborator queries the model, along with responses, it fetches the top matching references from these databases.
- Also, we are utilizing ANL-local resources/supercomputers such as Polaris and Aurora to perform this task.
- This tool can be used for any DUNE-specific information and could be expanded to perform data analyses efficiently in the future



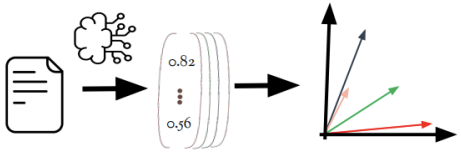


RETRIEVAL AUGMENTED GENERATION (RAG)

- Generic LLMs aren't trained on domain-specific internal documents
- We store internal documents into a vector space and search through them later as needed
- Upon querying, the retrieved documents are then fed into the LLM as context, enabling domain-aware responses.

01

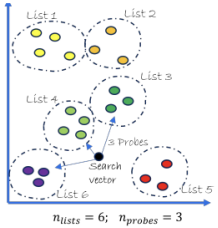
Embeddings



Transformer model converts **paragraphs** into **vectors** encoding semantic meaning.

02

Retrieval



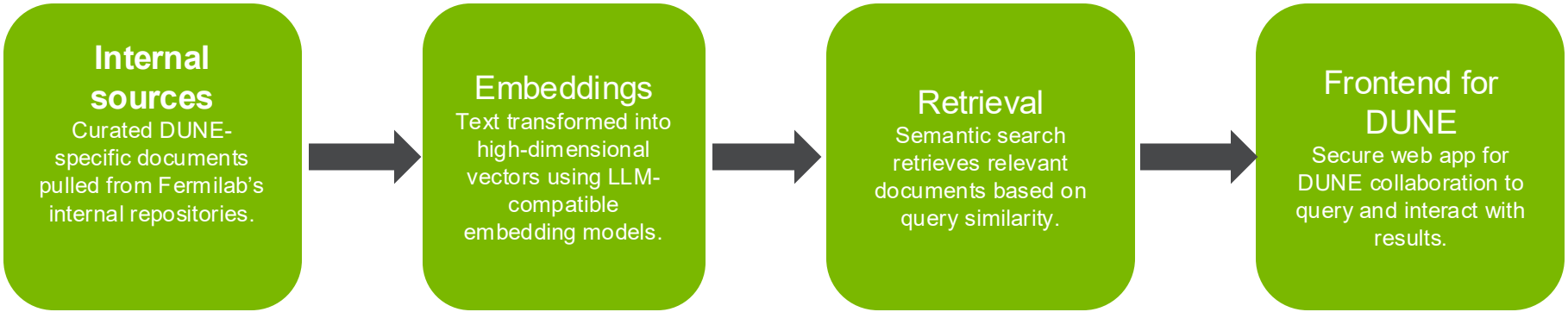
A **Cosine** Similarity Search via **Approximate** Nearest Neighbors finding the closest matching vectors.

03

Generation

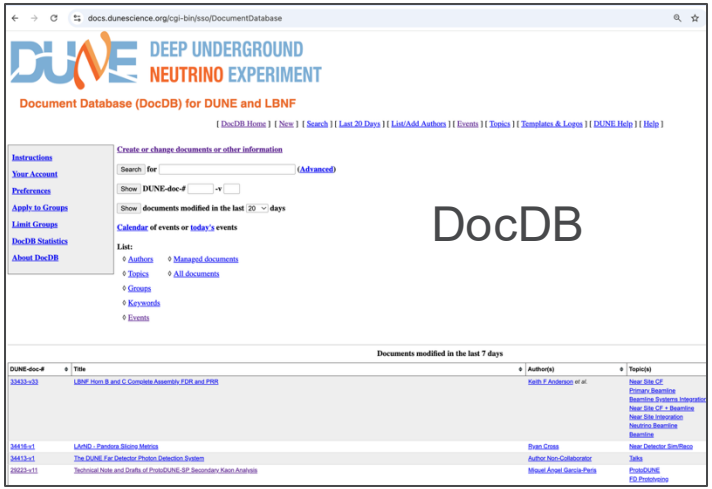
LLM uses retrieved documents to **answer** the question asked

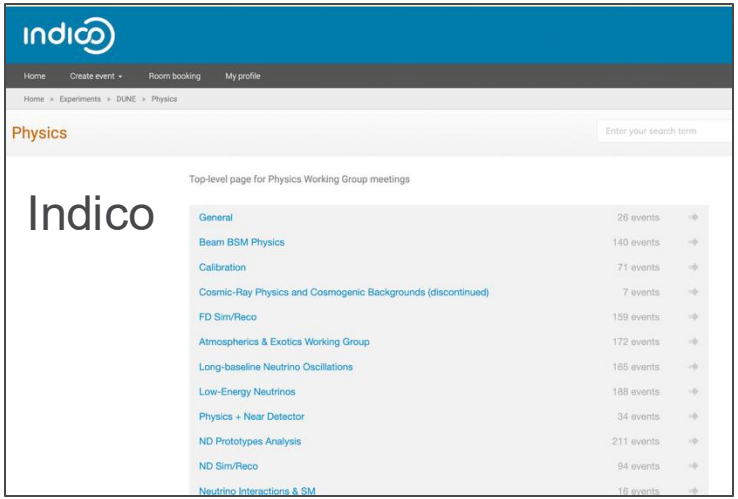
WORKFLOW


Internal sources
Curated DUNE-specific documents pulled from Fermilab's internal repositories.
Embeddings
Text transformed into high-dimensional vectors using LLM-compatible embedding models.
Retrieval
Semantic search retrieves relevant documents based on query similarity.
Frontend for DUNE
Secure web app for DUNE collaboration to query and interact with results.

DOCUMENT EXTRACTION

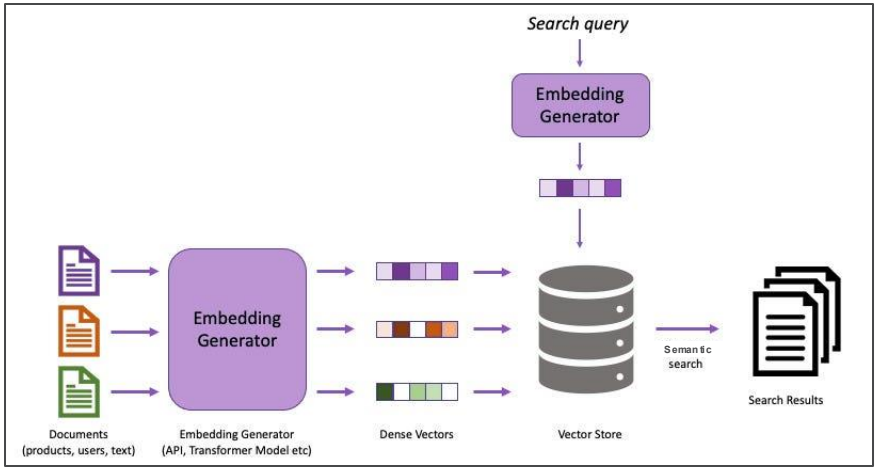
- To enable retrieval-based workflows, documents from DUNE collaboration platforms are extracted and cleaned using two tailored scripts
- Current Sources: **DocDB** and **Indico** platforms
- Access via shared collaboration credentials
- Process Summary:
  - HTML documents are parsed using BeautifulSoup
  - Extracted content includes:
    - Text from pdf, pptx, docx, txt etc
    - Metadata: URL, title, authors, creation date, last updated date, source
- Current focus is on text content only





EMBEDDING AND INDEXING

- Embedding:
  - Each document is embedded using the **multi-qa-mpnet-base-dot-v1** model from **Hugging Face's Sentence Transformers**
  - Embeddings capture semantic meaning to enable context-aware retrieval
- Indexing & Storage:
  - **FAISS (Facebook AI Similarity Search)** vector space is used for efficient similarity search
  - Embedding pipeline checks for pre-processed documents to **avoid redundancy**
- Data Stored:
  - Document **embeddings** (vector representations)
  - Associated **metadata** (title, author, date, source, etc.)
  - Indexed document IDs
  - **FAISS vector index** for rapid retrieval



UTILIZING AURORA FOR LARGE-SCALE EMBEDDING

- **Objective:**
  To scale embedding and indexing of DUNE collaboration documents, we are leveraging the **Aurora exascale supercomputer** at Argonne National Laboratory (ANL)
- **System Overview – Aurora:**
  - Each **compute node** includes
    - 2 × Intel CPUs
    - 6 × Intel GPUs
    - System comprises **>10,000 nodes**, enabling exascale-scale parallel processing
- **Resource Allocation:**
  - Awarded **DD Allocation** under Aurora Early Science Program
    - **5,000 node-hours**
    - **5 TB** of high-speed storage on the **FLARE** system

Aurora System Specifications

Compute Node
2 Intel Xeon CPU Max Series processors: 64GB HBM on each, 512GB DDR5 each; 6 Intel Data Center GPU Max Series, 128GB HBM on each, RAMBO cache on each; Unified Memory Architecture; 8 SlingShot 11 fabric endpoints

CPU-GPU Interconnect
CPU-GPU: PCIe; GPU-GPU: Xe Link

System Performance
Exascale

Platform
HPE Cray EX supercomputer

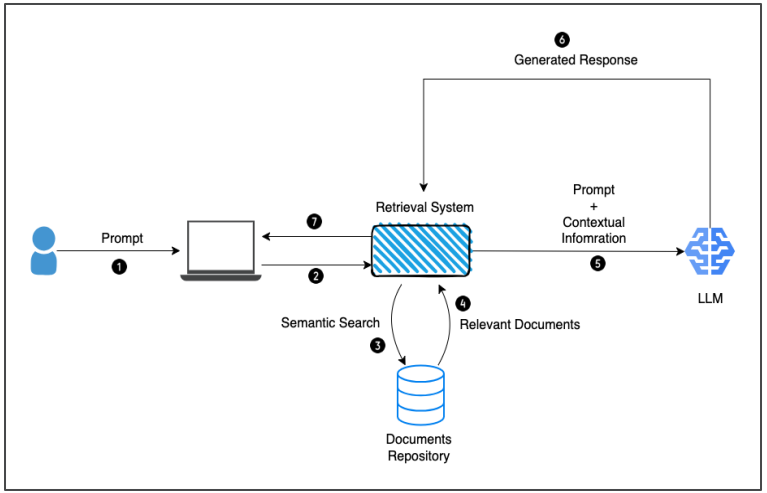
SCALABLE EMBEDDING VIA BALSAM WORKFLOW

- **Workflow Setup:**
  - Created project spaces and apps using the **Balsam workflow manager**
  - Developed a portable app: **LLM_DUNE-GPT**
    - Easily deployable across ALCF systems by adjusting configuration files
    - Fully scriptable in Python for modular control
- **Execution on Aurora:**
  - **Embedding and indexing jobs** are distributed and executed via Balsam
  - Supports **parallel job execution**, with each job processing separate documents
  - Allows efficient scaling of compute-intensive embedding step
- **Monitoring & Utilization:**
  - Monitor job progress and resource utilization per node
  - Automatic logging and job tracking for robustness

```python
from balsam.api import ApplicationDefinition, BatchJob, Job, Site
import yaml

'''
Create and store Embeddings
'''

yaml_file_path = '/lus/flare/projects/LLM_for_DUNE/users/aditya/BALSAM/L
site_name = "LLM_DUNEGPT"
path_module = f"/lus/flare/projects/LLM_for_DUNE/users/aditya/DUNEGPT_au

"""
Functions
"""

def get_env_from_yaml():
    with open(yaml_file_path, 'r') as file:
        yaml_content = file.read()
    yaml_dict = yaml.safe_load(yaml_content)
    env = yaml_dict['base_envs'][0]['env']
    return env


"""
embed
----------------------------------------
"""

env = get_env_from_yaml()
app_id = "embed_docdb_indico"

class embed(ApplicationDefinition):
    site = site_name
    print(env)
    environment_variables = env

    command_template = f"{path_module}/new_venv/bin/python cli.py index
    def shell_preamble(self):
        return f'''
        export INDEX={self.job.data["i"]}
        module load autoconf cmake
        module load frameworks
        module load cmake
        module use /soft/modulefiles
        cd {path_module}
        source {path_module}/new_venv/bin/activate
        source {path_module}/init.sh
        '''

embed.sync()
```

RETRIEVAL & RESPONSE GENERATION

- **Retrieval Workflow:**
  - User query triggers **semantic search** on FAISS to retrieve top-matching documents based on vector similarity
  - Retrieved documents are injected into a **custom prompt** to provide domain-specific context
- **Response Generation:**
  - Prompt is sent via API calls to **Argo running GPT-4o**, generating accurate and relevant responses
  - **Data security** is ensured throughout the process
- **Future Plans:**
  - Transfer embeddings output from Aurora to Fermilab machines for localized processing
  - Explore **Fermilab-hosted APIs** like **Ollama** to reduce latency and enhance data control

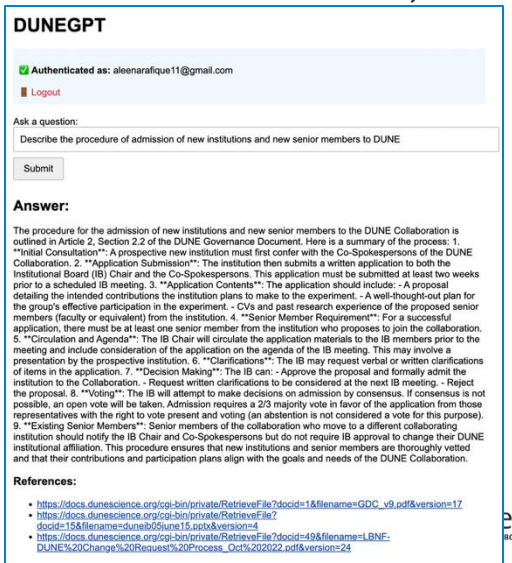


FERMILAB SINGLE SIGN-ON (SSO) AUTHENTICATION

- As a Fermilab-hosted experiment, **DUNE requires Fermilab authentication** for application deployment
- Implemented Fermilab **SSO authentication** integrated into the application
- Access restricted exclusively to **DUNE collaboration members**
- Application currently in **development phase**, including ongoing document processing
- Production-ready app planned for **near future release**
- Pending **Fermilab global webpage approval** for full deployment and public access

FRONTEND / WEB INTERFACE

- Developed app using **FastAPI**, enabling responsive handling of user queries
- Integrates semantic search across extracted texts from **Indico** and **DocDB** sources
- Returns responses based on a combination of:
  - **Cosine similarity** between query and document embeddings
  - **Model confidence** in the generated answer
- Initial testing with sample questions shows **accurate and relevant results**, consistent with expectations



SUMMARY AND NEXT STEPS

- Processing embedding jobs on **Aurora** for the large volume of DUNE documents (over 10,000 per database) is currently underway
- Upon completion, embeddings will be **transferred to Fermilab machines** for localized retrieval
- Exploring the use of **Fermilab-hosted LLMs**, if available, to enhance system integration and reduce latency
- **Preliminary website design** is ready; planned launch for DUNE users post **Fermilab webpage approval**
- Code repository available at: https://github.com/DUNE/LLM_for_DUNE/tree/main

BACKUP SLIDES

SYSTEM WORKFLOW: QUERYING WITH RAG + API (ARGO/OLLAMA)

- Plan is to create embedding and indexing using Aurora
- The FAISS vector space will then be copied over to Fermilab
- We are exploring the possibilities of using Fermilab-hosted API such as Ollama
- When a user submits a query, the system performs the following steps:
  1. **Retrieve top-matching documents** from FAISS based on vector similarity.
  2. **Construct a custom prompt** by injecting the retrieved document content into the context.
  3. **Send the prompt** to the **API (Argo, Ollama etc)** for final response generation.
- This pipeline leverages RAG to deliver accurate, context-aware answers based on DUNE documentation.