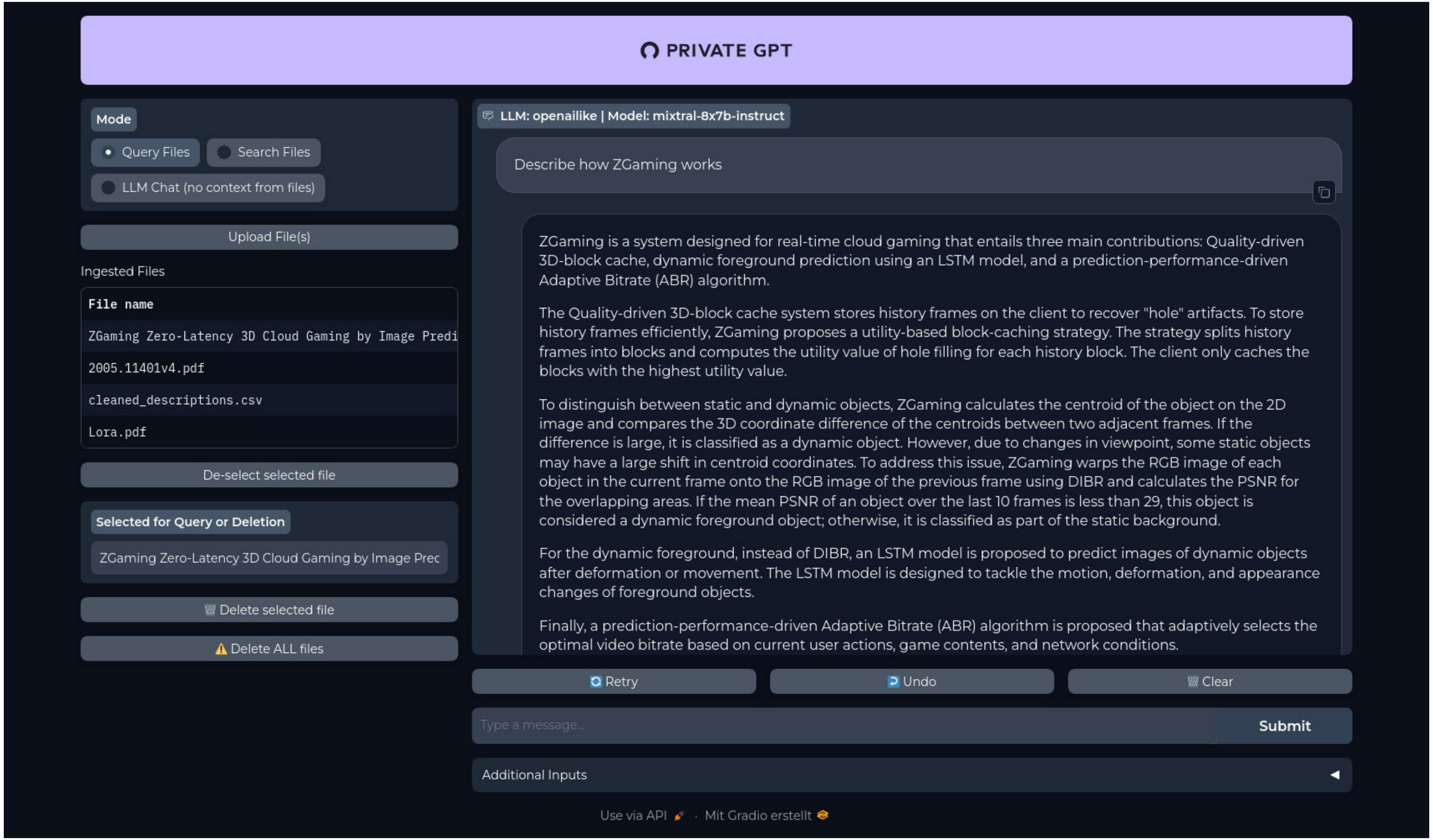
GöAID: Introduction to Retrieval Augmented Generation (RAG) and PrivateGPT

Jan Lenke

04.06.2024



Mode

● Query Files ○ Search Files

○ LLM Chat (no context from files)

Ingested Files

File name
ZGaming Zero-Latency 3D Cloud Gaming by Image Predi
2005.11401v4.pdf
cleaned_descriptions.csv
Lora.pdf

Selected for Query or Deletion

ZGaming Zero-Latency 3D Cloud Gaming by Image Prec

LLM: openailike | Model: mixtral-8x7b-instruct

Describe how ZGaming works

ZGaming is a system designed for real-time cloud gaming that entails three main contributions: Quality-driven 3D-block cache, dynamic foreground prediction using an LSTM model, and a prediction-performance-driven Adaptive Bitrate (ABR) algorithm.

The Quality-driven 3D-block cache system stores history frames on the client to recover "hole" artifacts. To store history frames efficiently, ZGaming proposes a utility-based block-caching strategy. The strategy splits history frames into blocks and computes the utility value of hole filling for each history block. The client only caches the blocks with the highest utility value.

To distinguish between static and dynamic objects, ZGaming calculates the centroid of the object on the 2D image and compares the 3D coordinate difference of the centroids between two adjacent frames. If the difference is large, it is classified as a dynamic object. However, due to changes in viewpoint, some static objects may have a large shift in centroid coordinates. To address this issue, ZGaming warps the RGB image of each object in the current frame onto the RGB image of the previous frame using DIBR and calculates the PSNR for the overlapping areas. If the mean PSNR of an object over the last 10 frames is less than 29, this object is considered a dynamic foreground object; otherwise, it is classified as part of the static background.

For the dynamic foreground, instead of DIBR, an LSTM model is proposed to predict images of dynamic objects after deformation or movement. The LSTM model is designed to tackle the motion, deformation, and appearance changes of foreground objects.

Finally, a prediction-performance-driven Adaptive Bitrate (ABR) algorithm is proposed that adaptively selects the optimal video bitrate based on current user actions, game contents, and network conditions.

PrivateGPT

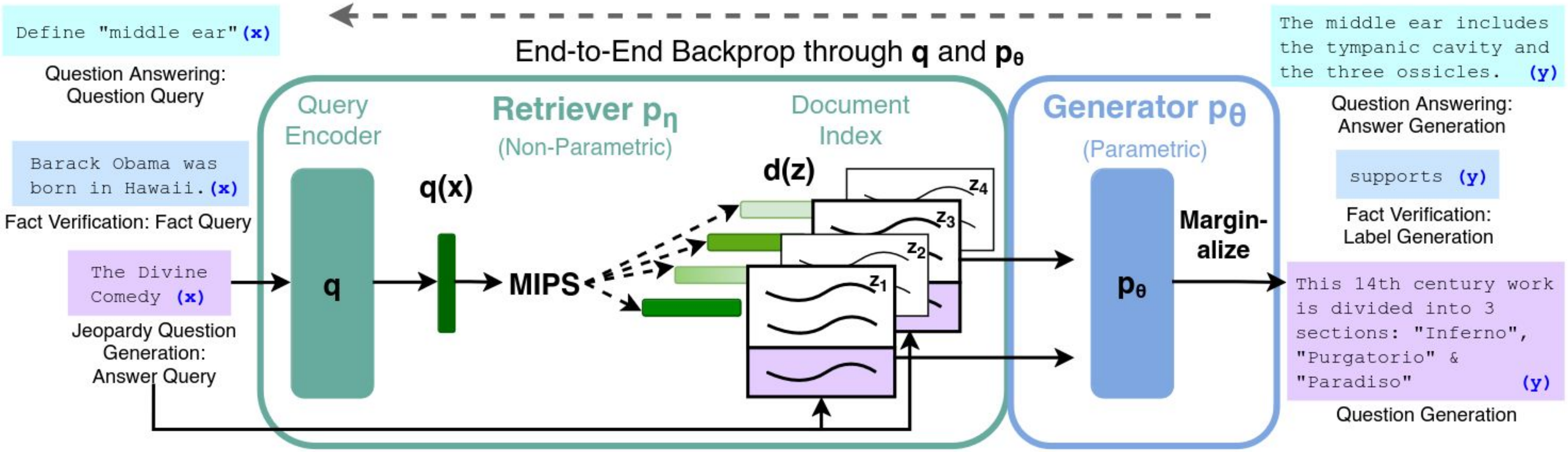

Define "middle ear" (x)
Question Answering: Question Query
Barack Obama was born in Hawaii. (x)
Fact Verification: Fact Query
The Divine Comedy (x)
Jeopardy Question Generation: Answer Query
End-to-End Backprop through q and pθ
Query Encoder
Retriever pη
(Non-Parametric)
Document Index
q
q(x)
MIPS
d(z)
z4
z3
z2
z1
Generator pθ
(Parametric)
pθ
Margin-alize
The middle ear includes the tympanic cavity and the three ossicles. (y)
Question Answering: Answer Generation
supports (y)
Fact Verification: Label Generation
This 14th century work is divided into 3 sections: "Inferno", "Purgatorio" & "Paradiso" (y)
Question Generation

Figure 1: Overview of our approach. We combine a pre-trained retriever (*Query Encoder* + *Document Index*) with a pre-trained seq2seq model (*Generator*) and fine-tune end-to-end. For query x, we use Maximum Inner Product Search (MIPS) to find the top-K documents z_i. For final prediction y, we treat z as a latent variable and marginalize over seq2seq predictions given different documents.

Capabilities of RAG

- Access to external domain-specific knowledge (including confidential information)
- Faster knowledge updating
- Improved performance on knowledge-intensive tasks
- Reduced resource requirements compared to other fine-tuning methods
- Dynamic memory access
- Differentiable access mechanism
- Provenance

Limitations of RAG

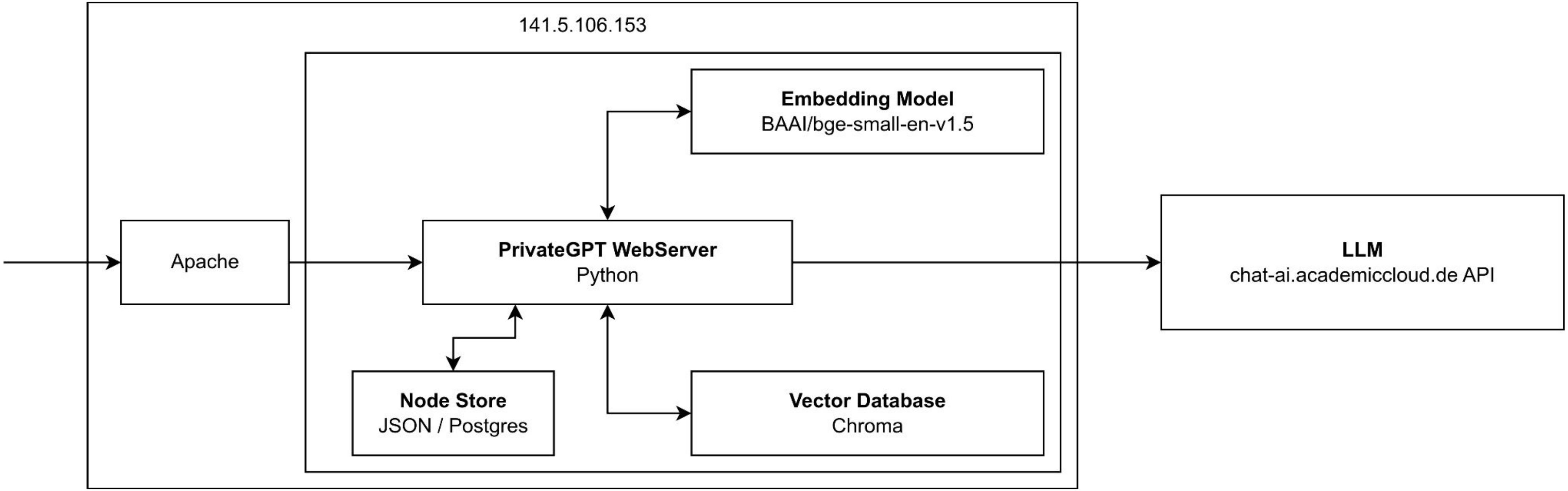
- Performance lags behind task-specific architectures
- Dependence on a pre-trained neural retriever
- Limited to extractive downstream tasks, limited reasoning capability (no iterative reasoning)
- Retrieval challenges

Live Demo

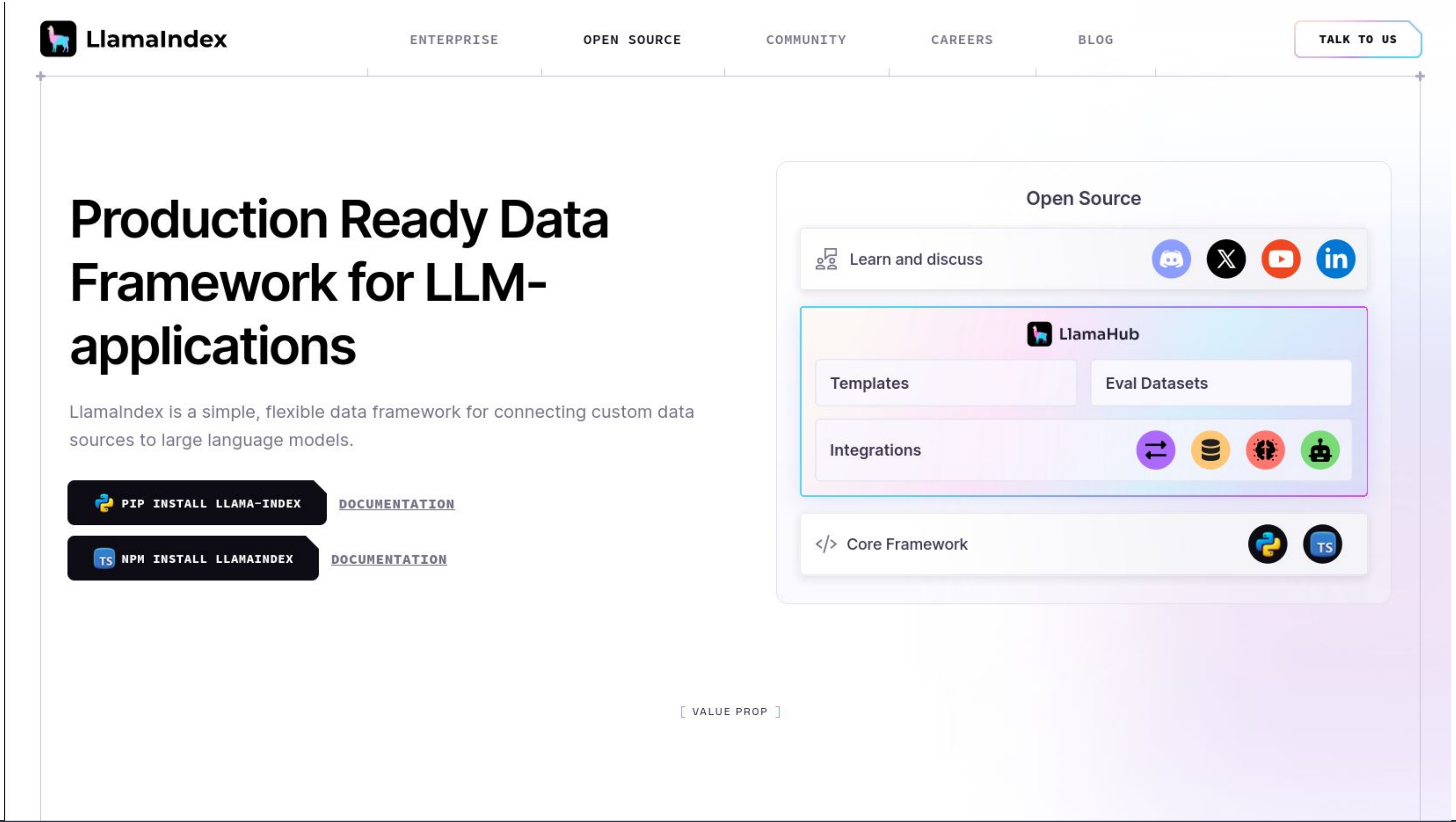
http://141.5.106.153/
Username: gwdg
Password: ragservice

Live Demo

- No RAG:
    - Write a poem about unicorns in the sky
    - Describe how ZGaming works
- Query files:
    - 3603269.3604819-repaired.pdf
        - Describe how ZGaming works
    - Lora.pdf
        - Summarize the 5 most important points from the paper in 200 words or less.
        - How does the low-rank adaptation work?
- Search files:
    - What are the limitations of ZGaming?
- Upload a pdf file



PrivateGPT components



Production Ready Data Framework for LLM-applications

LlamaIndex is a simple, flexible data framework for connecting custom data sources to large language models.

```
PIP INSTALL LLAMA-INDEX
NPM INSTALL LLAMAINDEX
```

RAG implementation: LlamaIndex library

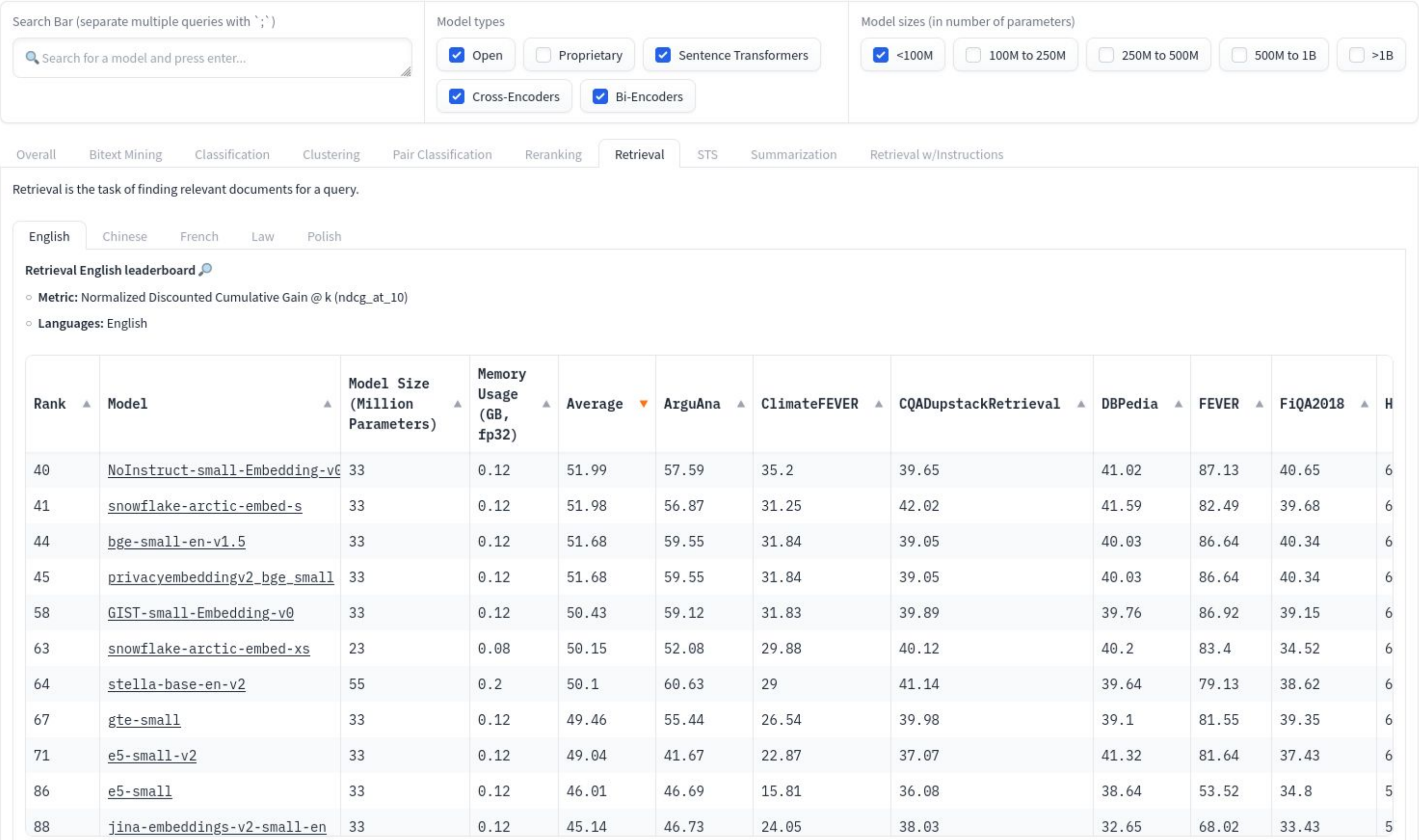
OpenAI-like API

- https://api.openai.com
  - /v1/completions
  - /v1/chat/completions
  - /v1/embeddings
  - …

- https://chat-ai.academiccloud.de
  - /v1/completions
  - /v1/chat/completions

Current setup

- **LLM:** mixtral-8x7b-instruct
    - using chat-ai.academiccloud.de
- **Vector Database:** Chroma
- **Node Store:** simple (.json files)
- **Embedding Model:** BAAI/bge-small-en-v1.5
    - supports any HuggingFace compliant implementation
    - ONNX model
        - 33 million parameters, 133 MB model size
        - Embedding dimension: 384
        - English only (Multilingual variants exist)
        - Rank 44 in MTEB (Retrieval Task, English)
    - Top 2 documents used
- Supported file formats: Text files (.txt, .md, .csv, .json, …), .pdf, .docx, .pptx, .ppt, .pptm, .epub, .mbox, .ipynb, .hwp



Search Bar (separate multiple queries with `;`)

Search for a model and press enter...

Model types

☑ Open □ Proprietary ☑ Sentence Transformers ☑ Cross-Encoders ☑ Bi-Encoders

Model sizes (in number of parameters)

☑ <100M □ 100M to 250M □ 250M to 500M □ 500M to 1B □ >1B

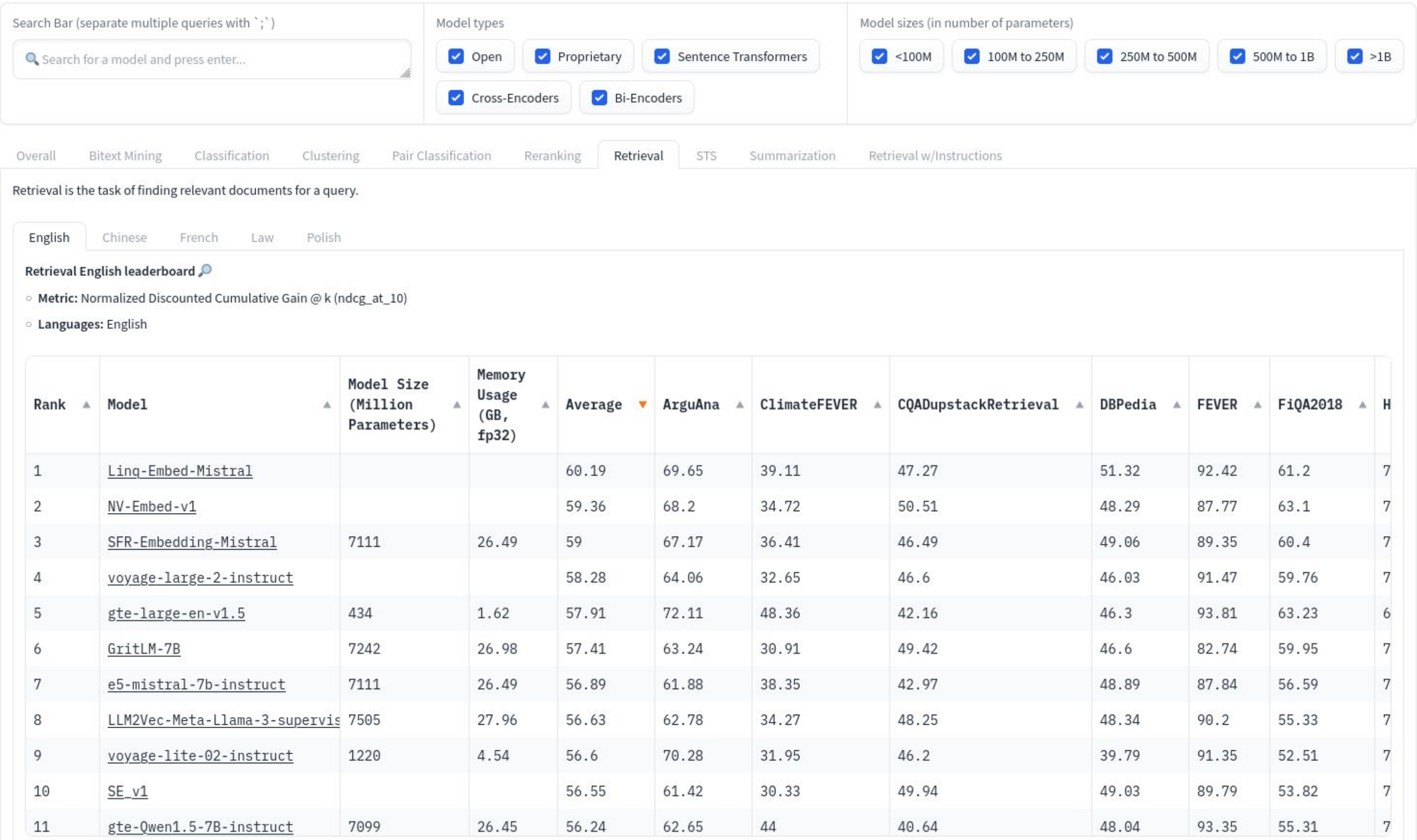
Retrieval is the task of finding relevant documents for a query.

Retrieval English leaderboard 🔎

- **Metric:** Normalized Discounted Cumulative Gain @ k (ndcg_at_10)
- **Languages:** English

Rank ▲	Model ▲	Model Size (Million Parameters) ▲	Memory Usage (GB, fp32) ▲	Average ▼	ArguAna ▲	ClimateFEVER ▲	CQADupstackRetrieval ▲	DBPedia ▲	FEVER ▲	FiQA2018 ▲	H
40	NoInstruct-small-Embedding-v(	33	0.12	51.99	57.59	35.2	39.65	41.02	87.13	40.65	6
41	snowflake-arctic-embed-s	33	0.12	51.98	56.87	31.25	42.02	41.59	82.49	39.68	6
44	bge-small-en-v1.5	33	0.12	51.68	59.55	31.84	39.05	40.03	86.64	40.34	6
45	privacyembeddingv2_bge_small	33	0.12	51.68	59.55	31.84	39.05	40.03	86.64	40.34	6
58	GIST-small-Embedding-v0	33	0.12	50.43	59.12	31.83	39.89	39.76	86.92	39.15	6
63	snowflake-arctic-embed-xs	23	0.08	50.15	52.08	29.88	40.12	40.2	83.4	34.52	6
64	stella-base-en-v2	55	0.2	50.1	60.63	29	41.14	39.64	79.13	38.62	6
67	gte-small	33	0.12	49.46	55.44	26.54	39.98	39.1	81.55	39.35	6
71	e5-small-v2	33	0.12	49.04	41.67	22.87	37.07	41.32	81.64	37.43	6
86	e5-small	33	0.12	46.01	46.69	15.81	36.08	38.64	53.52	34.8	5
88	jina-embeddings-v2-small-en	33	0.12	45.14	46.73	24.05	38.03	32.65	68.02	33.43	5

MTEB Leaderboard: Small non-proprietary models - Retrieval Task - English



Search Bar (separate multiple queries with `;`)

🔍 Search for a model and press enter...

Model types

☑ Open ☑ Proprietary ☑ Sentence Transformers

☑ Cross-Encoders ☑ Bi-Encoders

Model sizes (in number of parameters)

☑ <100M ☑ 100M to 250M ☑ 250M to 500M ☑ 500M to 1B ☑ >1B

Retrieval is the task of finding relevant documents for a query.

Retrieval English leaderboard 🔎

- **Metric:** Normalized Discounted Cumulative Gain @ k (ndcg_at_10)
- **Languages:** English

Rank ▲	Model ▲	Model Size (Million Parameters) ▲	Memory Usage (GB, fp32) ▲	Average ▼	ArguAna ▲	ClimateFEVER ▲	CQADupstackRetrieval ▲	DBPedia ▲	FEVER ▲	FiQA2018 ▲	H
1	Linq-Embed-Mistral			60.19	69.65	39.11	47.27	51.32	92.42	61.2	7
2	NV-Embed-v1			59.36	68.2	34.72	50.51	48.29	87.77	63.1	7
3	SFR-Embedding-Mistral	7111	26.49	59	67.17	36.41	46.49	49.06	89.35	60.4	7
4	voyage-large-2-instruct			58.28	64.06	32.65	46.6	46.03	91.47	59.76	7
5	gte-large-en-v1.5	434	1.62	57.91	72.11	48.36	42.16	46.3	93.81	63.23	6
6	GritLM-7B	7242	26.98	57.41	63.24	30.91	49.42	46.6	82.74	59.95	7
7	e5-mistral-7b-instruct	7111	26.49	56.89	61.88	38.35	42.97	48.89	87.84	56.59	7
8	LLM2Vec-Meta-Llama-3-supervis	7505	27.96	56.63	62.78	34.27	48.25	48.34	90.2	55.33	7
9	voyage-lite-02-instruct	1220	4.54	56.6	70.28	31.95	46.2	39.79	91.35	52.51	7
10	SE_v1			56.55	61.42	30.33	49.94	49.03	89.79	53.82	7
11	gte-Qwen1.5-7B-instruct	7099	26.45	56.24	62.65	44	40.64	48.04	93.35	55.31	7

MTEB Leaderboard: All models - Retrieval Task - English

Additional features

- Bulk Local Ingestion via CLI tool (+ watch changes in directory)
- Reranking of documents
  - Query top k documents to consider for reranking, filter out k-n documents, use only top n documents in response generation (e.g. cross-encoder/ms-marco-MiniLM-L-2-v2)
- Support for audio files (.mp3, .mp4) using OpenAI Whisper model
- Additional configuration:
  - Threshold for RAG: rag.similarity_value
  - Ingest mode: simple, batch, parallel, pipeline

Limitations of the current setup

- Performance & Scalability:
  - Embedding model running on the CPU and on the same node as the webserver
    - → Support the /v1/embeddings API for chat-ai.academiccloud.de
  - Vector database running on the same node as the webserver
  - Sequential ingesting of documents
- Security & Privacy:
  - No separation of data between different users in PrivateGPT
  - Embedding model and vector database running in the same process as the webserver
- No model selection in the UI

Thank you for listening