Open-source RAG using Mixtral 8x7B for financial data in German, Italian, and French

'Libérté, égalité, architecture open-sourcé...'

Mixture-of-Experts 🎆

For the A.I community, 2023 was certainly the year of the ***Large Language Model (LLM)***. As of 2024, the new trendy concept is the ***Mixture-of-Experts (MoE)***. MoE is not something new, it was originally introduced in 1991 in a paper called ***"Adaptative Mixtures of Local Experts"***.

Behind the democratization of MoE, is a Paris-based startup **Mistral AI**. The French company Mistral AI has recently released ***Mixtral 8x7B***, a fully open-source large language model based on the MoE architecture (arXiv:2401.04088). This pre-trained generative Sparse Mixture of Experts, Mixtal 8x7B outperforms Llama 2 70B and ChatGPT 3.5 of OpenAI on many benchmarks and supports English, French, German, Italian, and Spanish.

Mixtral 8x7B weights are open-source and licensed under Apache 2.0. This license makes it possible to understand the MoE architecture better, to use it commercially, and to build on top of another language. According to the paper ***"Aurora: Activating Chinese chat capability for Mixtral-8x7B sparse Mixture-of-Experts through instruct tuning"*** with 3 dialogue datasets in Chinese, researchers increased the Chinese conversational capabilities of Mixtral-8x7B (arXiv:2312.14557).

Although OpenAI has not officially confirmed its proprietary system, there's speculation that GPT-4 is based on an MoE architecture. In 2022, Google researchers *William Fedus, Barret Zoph*, and *Noam Shazeer* gave us great insights into a Mixture of Experts with the paper ***"Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity"*** (arXiv:2101.03961).

The sparse Mixture-of-Experts network can scale to trillion parameter models with reduced cost and latency at a constant computational cost. Mixtral 8x7B is based on Mistral 7B architecture with ***Grouped-Query Attention (GQA)***, ***Sliding-Window Attention (SWA)***, and ***Byte-fallback BPE*** tokenizer and trained on a context size of 32k tokens.

The important change in MoE comes from the ***Feed Forward Neural Networks*** (FFNs). The FFN leverages 8 groups of ***"experts"*** with a dynamic router function that chooses 2 experts per token to process the current state and combine their outputs.

To take an anthropomorphic analogy, a Mixture of Experts is one student asking a question to a group of eight teachers, each expert in their respective field. On the opposite, a Large

Language Model is one student asking one question to a very experienced teacher who needs time to reply.

- Experimentation in French, German, and Italian 🔬

English as a means of scientific communication creates an imperialism of the English language. In my latest notebook, I tried to explain the specificities of Japanese for RAG architecture (Retrieval Augmented Generation) in a low-resource environment. Similarly, this notebook is a preliminary test of a ***vanilla RAG*** with ***Mixtral 8x7B in French, Italian***, and ***German*** over ***financial documents in a low-resource environment.***

Disclaimer: The minimum hardware requirements for Mixtral-8x7B might be in 4-bit precision using 22.5Gb of VRAM, in 8-bit precision using 45Gb of VRAM, and in half-precision using 90GB of VRAM. I was able to run it on Ollama and vLLM on V100 and A100 on Google Colab Pro. In this notebook, I am using a ***quantized model of Mixtral 8x7B*** created by The Bloke and the inference framework, ***LlamaCPP***. The process of quantization diminishes the results, therefore it doesn't correctly define the excellence of the official model, Mixtral 8x7B built by Mistral AI.

```
!pip install git+https://github.com/huggingface/transformers --quiet
!pip install llama-index cohere pymupdf typing_extensions --upgrade --quiet

!pip install llama-index-vector-stores-pgvecto-rs --upgrade --quiet  #Vector se
!pip install llama-index-embeddings-huggingface --upgrade --quiet #Embeddings f
!pip install llama-index-llms-llama-cpp --upgrade --quiet #LLM inference of Lla

!CMAKE_ARGS="-DLLAMA_CUBLAS=on" FORCE_CMAKE=1 pip install "llama-cpp-python==0.
```

```
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
  Building wheel for transformers (pyproject.toml) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 52.0/52.0 kB 3.4 MB/s eta 0:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.4/4.4 MB 56.5 MB/s eta 0:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 15.3/15.3 MB 92.2 MB/s eta 0:
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.0/2.0 MB 91.3 MB/s eta 0:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.1/3.1 MB 108.2 MB/s eta 0:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 30.6/30.6 MB 48.0 MB/s eta 0:
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 75.6/75.6 kB 9.9 MB/s eta 0:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 108.0/108.0 kB 18.4 MB/s eta
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 227.4/227.4 kB 33.7 MB/s eta
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.8/1.8 MB 97.9 MB/s eta 0:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 286.1/286.1 kB 38.3 MB/s eta
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 525.5/525.5 kB 55.2 MB/s eta
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.8/6.8 MB 106.3 MB/s eta 0:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 77.8/77.8 kB 8.7 MB/s eta 0:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 58.3/58.3 kB 9.8 MB/s eta 0:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.4/49.4 kB 8.0 MB/s eta 0:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.4/2.4 MB 101.1 MB/s eta 0:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 92.1/92.1 kB 14.6 MB/s eta 0:
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 60.8/60.8 kB 9.9 MB/s eta 0:0
```

```
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 41.3/41.3 kB 6.5 MB/s eta 0:0
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5.4/5.4 MB 119.4 MB/s eta 0:0
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 58.4/58.4 kB 9.1 MB/s eta 0:0
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 105.7/105.7 kB 16.8 MB/s eta
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 67.3/67.3 kB 11.2 MB/s eta 0:
Installing build dependencies ... done
Getting requirements to build wheel ... done
Preparing metadata (pyproject.toml) ... done
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 698.9/698.9 kB 62.9 MB/s eta
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.6/1.6 MB 87.2 MB/s eta 0:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 67.6/67.6 kB 10.5 MB/s eta 0:
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 138.5/138.5 kB 19.9 MB/s eta
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 46.0/46.0 kB 7.0 MB/s eta 0:0
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 71.5/71.5 kB 11.3 MB/s eta 0:
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 50.8/50.8 kB 8.1 MB/s eta 0:0
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 341.4/341.4 kB 37.6 MB/s eta
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.4/3.4 MB 106.9 MB/s eta 0:0
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.3/1.3 MB 87.6 MB/s eta 0:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 130.2/130.2 kB 18.8 MB/s eta
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 86.8/86.8 kB 14.0 MB/s eta 0:
Building wheel for pypika (pyproject.toml) ... done
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 178.1/178.1 kB 7.0 MB/s eta 0
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.3/3.3 MB 46.0 MB/s eta 0:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 755.5/755.5 MB 2.4 MB/s eta 0
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 23.7/23.7 MB 54.6 MB/s eta 0:
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 823.6/823.6 kB 68.4 MB/s eta
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 14.1/14.1 MB 93.6 MB/s eta 0:
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 731.7/731.7 MB 2.3 MB/s eta 0
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 410.6/410.6 MB 3.1 MB/s eta 0
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 121.6/121.6 MB 13.6 MB/s eta
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 56.5/56.5 MB 30.6 MB/s eta 0:
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 124.2/124.2 MB 7.9 MB/s eta 0
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 196.0/196.0 MB 3.9 MB/s eta 0
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 166.0/166.0 MB 10.4 MB/s eta
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 99.1/99.1 kB 15.1 MB/s eta 0:
```

- *Llama-Index* 

Llama-Index is a robust data framework created by ***Jerry Liu*** and ***Simon Suo*** perfect for LLM-powered solutions. The latest version 0.10.x has been improving the overall implementation. Llama-Index modified the structured packages into template/integrate, incorporated the llama-hub inside, and deprecated ServiceContext.

More flexible, Llama-Index enables us to change ***global settings*** and ***local settings*** for chunkers, vector stores, queries, or LLMs. This global/local construction could be good for fallback. For example, if my LLM 1 (Mixtral-7x8B) is down momentarily for technical reasons, my LLM 2 (ChatGPT4.0) could be loaded.

```
import os, sys, logging, warnings
warnings.filterwarnings('ignore')

import nest_asyncio
nest_asyncio.apply()
```

```
nest_asyncio.apply()

from llama_index.core import Settings #Incorporate settings function
from llama_index.vector_stores.pgvecto_rs import PGVectoRsStore
from llama_index.core.node_parser import SentenceSplitter

import tqdm, textwrap, ipywidgets
from IPython.display import Markdown, display, clear_output


from transformers import AutoTokenizer

Settings.tokenizer = AutoTokenizer.from_pretrained("mistralai/Mixtral-8x7B-Inst
```

```
tokenizer_config.json: 100%  1.46k/1.46k [00:00<00:00, 125kB/s]
tokenizer.model: 100%  493k/493k [00:00<00:00, 26.4MB/s]
tokenizer.json: 100%  1.80M/1.80M [00:01<00:00, 1.46MB/
```

- *BGE M3-Embedding* 🥉

Great alternative to the ***multilangual E5-Large-instruct***, the ***Beijing Academy of Artificial Intelligence (BAAI)*** introduced recently their new embedding model, ***BGM-3***. According to their paper published in February 2024, ***"BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation"***, the model can perform ***dense retrieval, multi-vector retrieval***, and ***sparse retrieval*** on more than 100 languages (arXiv:2402.03216).

The research team incorporated a ***Self-Knowledge Distillation*** approach optimized with the batching strategy of size 128, 1024, 4096, and 8192. The pre-training consisted of 1.2 Billion multilingual unsupervised data to create a dense score. Then the fine-tuning introduced multilingual labeled data, and synthetic data (to mitigate the shortage of long documents). The hybrid approach in multi-stage integrate and normalise the dense retrieval, the multi-vector retrieval, and the lexical retrieval. The M3-Embedding model of BGE offers SoTA performance against other multilingual embedding models.

```
from llama_index.embeddings.huggingface import HuggingFaceEmbedding

Settings.embed_model = HuggingFaceEmbedding(model_name="BAAI/bge-m3") #HuggingF
```

```
config.json: 100%  687/687 [00:00<00:00, 60.7kB/s]
model.safetensors: 100%  2.27G/2.27G [00:05<00:00, 381MB/
```

```
tokenizer_config.json: 100%  1.31k/1.31k [00:00<00:00, 110kB/s]
sentencepiece.bpe.model: 100%  5.07M/5.07M [00:00<00:00, 202MB/
```

```
from llama_index.core import SimpleDirectoryReader, VectorStoreIndex
from llama_index.llms.llama_cpp import LlamaCPP
from llama_index.llms.llama_cpp.llama_utils import (
    messages_to_prompt,
    completion_to_prompt,
)
```

- *Quantized model of Mixtral-8X7B* ⚡

Tom Jobbins (The Bloke) has been releasing many quantized LLMs. For the official model Mixtral-8x7b, the most important choice is to calculate the performance and cost trade-off.

For our quantized model, you need to choose the trade-off between model size in GB and RAM consumption against the quality of the response. By playing on the smaller models, I witnessed many strange responses. So I decided on the following model:

- Mixtral-8x7b-instruct-v0.1.Q5_K_M.gguf
- Quant method: Q5_K_M
- Bits: 5
- Size: 32.23 GB
- Max RAM required: 34.73 GB

```
model_url = "https://huggingface.co/TheBloke/Mixtral-8x7B-Instruct-v0.1-GGUF/re
```

- *LlamaCPP - Inference* 💻

Created by ***Georgi Gerganov***, the inference framework ***LLaMA CPP*** written in C, C++, and CUDA became one of the used tools to enjoy LLM in local machines. It was originally designed to support Meta's LlaMA. It evolved into something much bigger with a dynamic community. LlamaCPP can be used with LlaMA 2, Alpaca, Falcon, Baichuan, Mistral, Bloom, Qwen, etc, and even few multimodal models such as LLaVA, BakLLaVA, and Yi-VL.

In our case, it's important to enable ***BLAS*** during the installation to accelerate CUDA kernels in our Nvidia GPU for the T4 Nvidia. Mixtral 8X7B can be run efficiently on single GPUs with high-performance specialized kernels. To optimize and accelerate the inference, the number of layers to offload to GPU can be changed with the function n_gpu_layers and the tradeoff could be found with the maximum batch size (n_batch) and context windows (n_cxt).

```
llm = LlamaCPP(
    model_url=model_url,
    temperature=0.1,
    max_new_tokens=1000,
    context_window=3900,
    generate_kwargs={},
    model_kwargs={"n_gpu_layers":1, "f16_kv": True}, #At least 1 for BLAS - Off
    messages_to_prompt=messages_to_prompt,
    completion_to_prompt=completion_to_prompt,
    verbose=True,
)
```

```
Downloading url https://huggingface.co/TheBloke/Mixtral-8x7B-Instruct-v0.1-
total size (MB): 32229.28
30737it [01:27, 351.74it/s]
llama_model_loader: loaded meta data with 26 key-value pairs and 995 tensor
llama_model_loader: Dumping metadata keys/values. Note: KV overrides do not
llama_model_loader: - kv   0:                       general.architecture st
llama_model_loader: - kv   1:                               general.name st
llama_model_loader: - kv   2:                      llama.context_length u3
llama_model_loader: - kv   3:                    llama.embedding_length u3
llama_model_loader: - kv   4:                         llama.block_count u3
llama_model_loader: - kv   5:                 llama.feed_forward_length u3
llama_model_loader: - kv   6:                llama.rope.dimension_count u3
llama_model_loader: - kv   7:                llama.attention.head_count u3
llama_model_loader: - kv   8:             llama.attention.head_count_kv u3
llama_model_loader: - kv   9:                      llama.expert_count u3
llama_model_loader: - kv  10:                 llama.expert_used_count u3
llama_model_loader: - kv  11:     llama.attention.layer_norm_rms_epsilon f3
llama_model_loader: - kv  12:                       llama.rope.freq_base f3
llama_model_loader: - kv  13:                          general.file_type u3
llama_model_loader: - kv  14:                       tokenizer.ggml.model st
llama_model_loader: - kv  15:                      tokenizer.ggml.tokens ar
llama_model_loader: - kv  16:                      tokenizer.ggml.scores ar
llama_model_loader: - kv  17:                  tokenizer.ggml.token_type ar
llama_model_loader: - kv  18:               tokenizer.ggml.bos_token_id u3
llama_model_loader: - kv  19:               tokenizer.ggml.eos_token_id u3
llama_model_loader: - kv  20:           tokenizer.ggml.unknown_token_id u3
llama_model_loader: - kv  21:           tokenizer.ggml.padding_token_id u3
llama_model_loader: - kv  22:              tokenizer.ggml.add_bos_token bo
llama_model_loader: - kv  23:              tokenizer.ggml.add_eos_token bo
llama_model_loader: - kv  24:                    tokenizer.chat_template st
llama_model_loader: - kv  25:               general.quantization_version u3
llama_model_loader: - type  f32:   65 tensors
llama_model_loader: - type  f16:   32 tensors
llama_model_loader: - type q8_0:   64 tensors
llama_model_loader: - type q5_K:  833 tensors
llama_model_loader: - type q6_K:    1 tensors
llm_load_vocab: special tokens definition check successful ( 259/32000 ).
llm_load_print_meta: format           = GGUF V3 (latest)
llm_load_print_meta: arch             = llama
llm_load_print_meta: vocab type       = SPM
llm_load_print_meta: n_vocab          = 32000
llm_load_print_meta: n_merges         = 0
llm_load_print_meta: n_ctx_train      = 32768
llm_load_print_meta: n_embd           = 4096
llm_load_print_meta: n_head           = 32
```

```
llm_load_print_meta: n_head_kv        = 8
llm_load_print_meta: n_layer          = 32
llm_load_print_meta: n_rot            = 128
llm_load_print_meta: n_embd_head_k    = 128
llm_load_print_meta: n_embd_head_v    = 128
llm_load_print_meta: n_gqa            = 4
llm_load_print_meta: n_embd_k_gqa     = 1024
llm_load_print_meta: n_embd_v_gqa     = 1024
llm_load_print_meta: f_norm_eps       = 0.0e+00
llm_load_print_meta: f_norm_rms_eps   = 1.0e-05
llm_load_print_meta: f_clamp_kqv      = 0.0e+00
llm_load_print_meta: f_max_alibi_bias = 0.0e+00
llm_load_print_meta: n_ff             = 14336
```

```
Settings.llm = llm
Settings.node_parser = SentenceSplitter(chunk_size=512, chunk_overlap=20) #chur
```

- *RAG over complex financial documents* 💹

In the latest version, ***Llama-Index*** incorporates ***PyMuPDFReader*** directly. We will be using the half-year financial reports of 2023 in PDF format from three European companies:

- ***Hermès International for French***
- ***Porsche AG for German***
- ***Brunello Cucinelli S.p.A for Italian***

Financial departments are outsourcing their layout to graphic agencies. In finance, documents constituting regulated information are often proofread by legal departments and external PR agencies.

In an high performing RAG system, the ingestion of complex PDFs is the most important part of the work. The constant switching between text data and tabular data is a big issue for financial documents. Tabular data problems could be solved with tools such as ***Camelot***, ***Tabula***, ***Pdfplumber***, ***LlamaParse***, and ***Unstructured***.

The trendy solution of using multimodal LLMs to scan images of tabular data in a financial report to create a summary is rather computationally expensive. It's like using a huge LLM for a simple name-entity recognition task. Scanned images inside a PDF file can be handled via Optical Character Recognition (OCR) systems with spatial attention.

```
from google.colab import files
uploaded = files.upload()
```

Browse… No files selected. Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving CUCINELLI_Semestrale_2023_web.pdf to CUCINELLI_Semestrale_2023_web.p
Saving Halbjahresfinanzbericht 2023.pdf to Halbjahresfinanzbericht 2023.pdf
Saving hermes-rapport-financier-semestriel-2023-fr-01.pdf to hermes-rapport-
```

- *Metadata is a weapon* 🔪

Metadata might be one of the most disregarded parameters. In the RAG system, the file's name is a key value for metadata. As ***General Michael V. Hayden*** (former N.S.A. and C.I.A. Director) famously said: ***"We kill people based on metadata."*** at Johns Hopkins University in April 2014.

In our 3 examples, we have 3 mistakes:

- The French luxury house, ***Hermès*** made a small typo on the file's name "hermes-rapport-financier-semetriel-2023-fr-01.pdf". The "s" is missing on an import word "semestriel" (half-year) in French.
- The German automobile constructor, ***Porsche*** "Halbjahresfinanzbericht 2023.pdf", doesn't give information about the company, just "half-year financial report 2023".
- And finally, the Italian house, ***Brunello Cucinelli*** included a confusing extra meta-data "web" version in the name "CUCINELLI_Semestrale_2023_web.pdf".

For metadata, a proper naming of the file grants ***spatio-temporal insights*** that are critical in finance. For a ***clear hierarchical naming system*** for PDF files, it's mandatory to include the full name of the company, the ticker, the financial report, the date, and the language.

Example: *"Hermes RMS EPA financial report half-year 2023 FR 30 June 2023.pdf"*.

```
from pathlib import Path
from llama_index.core import download_loader

PyMuPDFReader = download_loader("PyMuPDFReader")

loader = PyMuPDFReader()
hermes = loader.load(file_path="/content/hermes-rapport-financier-semetriel-202
porsche = loader.load(file_path="/content/Halbjahresfinanzbericht 2023.pdf")
cucinelli = loader.load(file_path="/content/CUCINELLI_Semestrale_2023_web.pdf")
```

- *Format freedom* 🎨

The U.S. Securities and Exchange Commission in the USA has clearly defined a structure for their company fillings such as 10K, 10Q, etc. The European Union gave creative freedom to public companies to publish their regulated information. For NLP-powered quantitative analysis, the coverage of listed companies in the European Union must be crafted in a bespoke manner. Regardless of the language, we are still victims of word games, semantics is always a b*tch (Gil Scott-Heron).

- ***Porsche AG*** (or Dr. Ing. h.c. F. Porsche AG) did its IPO on 29 September 2022, under the umbrella of Volkswagen Group. Porsche AG is a young public company with an austere

report in black and white, text unjustified, and very light at ***863Kb for 46 pages***.

- ***Hermès International*** gave us an interactive pdf, with a stylish orange layout, extremely pleasant to read at ***1,732Kb for 36 pages***.
- ***Brunello Cucinelli S.p.A*** provides a glossy half-year financial report with more than 15 high-quality pictures for ***3,528Kb for 112 pages*** in Italian (at 15,026Kb due to compression issues for the English format).

Below we can witness the different layouts used by companies in their financial reports. The tables of contents are unbalanced towards notes for Hermès, in all capitals for Porsche, or with many periods for Brunello Cucinelli.

`hermes[1]`

```
Document(id_='5d9be335-6dc7-4678-a086-ee7ab4d6ab7d', embedding=None,
metadata={'total_pages': 36, 'file_path': '/content/hermes-rapport-
financier-semetriel-2023-fr-01.pdf', 'source': '2'},
excluded_embed_metadata_keys=[], excluded_llm_metadata_keys=[],
relationships={}, text='1\nCHIFFRES CLÉS\n3\nPrincipales données
consolidées\ndu premier semestre 2023\n3\n2\nRAPPORT SEMESTRIEL
D’ACTIVITÉ\n5\n2.1\nFaits marquants du semestre\n5\nActivité à fin juin
par métier\n6\nActivité à fin juin par zone géographique\n5\n2.2\nChiffre
d’affaires et activité du premier semestre\n5\n2.3\nCommentaires sur les
comptes semestriels consolidés\nrésumés\n7\n2.3.1 Compte de
résultat\n7\n2.3.2 Flux de trésorerie et investissements\n8\n2.3.3
Situation financière\n8\n2.4\nPerspectives\n9\n2.5\nRisques et
incertitudes\n9\n2.6\nTransactions avec les parties liées\n9\n3\nCOMPTES
SEMESTRIELS CONSOLIDÉS\nRÉSUMÉS AU 30 JUIN 2023\n11\n3.1\nCompte de
résultat consolidé\n11\n3.2\nÉtat du résultat global
consolidé\n11\n3.3\nBilan consolidé\n12\n3.4\nÉtat de variation des
capitaux propres consolidés\n13\n3.5\nÉtat des flux de trésorerie
consolidés\n14\n3.6\nAnnexe aux comptes semestriels consolidés
résumés\n15\nNote 1\nPrincipes et méthodes comptables\n16\nNote
2\nIndicateurs alternatifs de performance\n16\nNote 3\nInformation
sectorielle\n18\nNote 4\nÉléments relatifs à l’activité
opérationnelle\n19\nNote 5\nAvantages au personnel\n21\nNote
6\nImmobilisations incorporelles, corporelles et\ncontrats de
location\n22\nNote 7\nParticipations dans les entreprises
associées\n24\nNote 8\nActifs et passifs financiers – Trésorerie
nette\n25\nNote 9\nGestion des risques de marché et
instruments\ndérivés\n26\nNote 10\nCapitaux propres – Résultat par
action\n26\nNote 11\nProvisions pour risques et charges et\nengagements
hors bilan\n27\nNote 12\nTransactions avec les parties liées\n27\nNote
13\nÉvénements postérieurs à la clôture\n27\n4\nRAPPORT DES COMMISSAIRES
AUX\nCOMPTES SUR L’INFORMATION
FINANCIÈRE\nSEMESTRIELLE\n29\n5\nDÉCLARATION DES RESPONSABLES\nDU RAPPORT
FINANCIER SEMESTRIEL\n33\nSOMMAIRE\n', start_char_idx=None,
end_char_idx=None, text_template='{metadata_str}\n\n{content}',
metadata_template='{key}: {value}', metadata_seperator='\n')
```

`porsche[1]`

```
Document(id_='e86490c7-e0d9-4e42-8c39-e7687ab84b2b', embedding=None,
metadata={'total_pages': 46, 'file_path': '/content/
```

```
    Halbjahresfinanzbericht 2023.pdf', 'source': '2'},
    excluded_embed_metadata_keys=[], excluded_llm_metadata_keys=[],
    relationships={}, text=' \n \n \n \nINHALT \n \n \n \n \n3 WESENTLICHE
    KENNZAHLEN \n \n \n KONZERN-ZWISCHENLAGEBERICHT \n5 GESCHÄFTSVERLAUF \n8
    ERTRAGS-, FINANZ- UND VERMÖGENSLAGE \n16 PROGNOSE-, CHANCEN- UND
    RISIKOBERICHT \n \n \n KONZERN-ZWISCHENABSCHLUSS (KURZFASSUNG) \n20
    KONZERN-GEWINN- UND VERLUSTRECHNUNG \n21 KONZERN-GESAMTERGEBNISRECHNUNG
    \n22 KONZERN-BILANZ \n23 KONZERN-EIGENKAPITALVERÄNDERUNGSRECHNUNG \n25
    KONZERN-KAPITALFLUSSRECHNUNG \n26 \nKONZERN-ANHANG \n  \n44 VERSICHERUNG
    DER GESETZLICHEN VERTRETER \n45 BESCHEINIGUNG NACH PRÜFERISCHER DURCHSICHT
    \n46 WEITERE INFORMATIONEN \n \n', start_char_idx=None, end_char_idx=None,
    text_template='{metadata_str}\n\n{content}', metadata_template='{key}:
    {value}', metadata_seperator='\n')


cucinelli[1]

    Document(id_='d0226d52-9f20-4c8c-bd0d-01e94664a71d', embedding=None,
    metadata={'total_pages': 112, 'file_path': '/content/
    CUCINELLI_Semestrale_2023_web.pdf', 'source': '2'},
    excluded_embed_metadata_keys=[], excluded_llm_metadata_keys=[],
    relationships={}, text='1\nRELAZIONE FINANZIARIA SEMESTRALE AL 30 GIUGNO
    2023\nINDICE \nDATI SOCIETARI
    .....
    4\nCOMPOSIZIONE DEGLI ORGANI SOCIALI AL 30 GIUGNO 2023
    .....
    5\nORGANIGRAMMA SOCIETARIO DEL GRUPPO AL 30 GIUGNO 2023
    ..... 6\nCOMPOSIZIONE
    DEL GRUPPO AL 30 GIUGNO 2023
    .....
    7\nRETE DISTRIBUTIVA
    .....
    8\nRELAZIONE INTERMEDIA DEL CONSIGLIO DI AMMINISTRAZIONE \nSULL'ANDAMENTO
    DELLA GESTIONE AL 30 GIUGNO 2023\nINFORMAZIONI SOCIETARIE
    .....
    DI SINTESI AL 30 GIUGNO 2023
    .....
    RISULTATI DEL GRUPPO NEL PRIMO SEMESTRE 2023
    .....34\nAN
    DEI RICAVI
    .....
    \n \n– \nRICAVI SUDDIVISI PER AREA GEOGRAFICA
    .....34\n
    \n \n– \nRICAVI SUDDIVISI PER CANALE DISTRIBUTIVO
    .....36\n \n
    \n– \nRICAVI SUDDIVISI PER LINEA DI PRODOTTO E PER TIPOLOGIA DI CLIENTE
    FINALE .....37\nANALISI DEL CONTO ECONOMICO
    .....
    \n \n– \nEBITDA E RISULTATI
    .....
    \n \n– \nCOSTI OPERATIVI
    .....
    \n \n– \nAMMORTAMENTI, ONERI FINANZIARI NETTI, IMPOSTE E RISULTATO NETTO
    .....42\nANALISI DEI SALDI PATRIMONIALI E FINANZIARI
    .....4
    \n \n– \nCAPITALE CIRCOLANTE NETTO
    .....
    \n \n– \nIMMOBILIZZAZIONI E ALTRE ATTIVITÀ/(PASSIVITÀ) NON CORRENTI
    .....46\n \n \n– \nINVESTIMENTI
```

```
.....................................................................
\n \n— \nINDEBITAMENTO FINANZIARIO
NETTO.....................................................................
\n \n— \nPATRIMONIO NETTO
.....................................................................
SUL GOVERNO SOCIETARIO
.....................................................................
start_char_idx=None, end_char_idx=None,
text_template='{metadata_str}\n\n{content}', metadata_template='{key}:
{value}', metadata_seperator='\n')
```

- *PGvectoR.s* 🧮

Written in Rust, ***Pgvecto.Rs*** is a Postgres extension that focuses on vector similarity search function. Developed by ***Allen Zhou*** and ***Ce Gao*** of ***Tensorchord***, the beta version already has amazing features, evolving extremely fast over the past few months. Pgvecto.rs supports vector length up to 65.535 perfect for LLM. The implementation in Rust enables various indexing algorithms with vector representations, including fp16, int8, and even sparse vectors.

Pgvecto.Rs provides basic distance metrics (Euclidean, Dot, and Cosine distance), indexing algorithms (flat, IVF, and HNSW), and full text/vector search. Last month, the talented team of Tensorchord implemented the ***VBASE*** filtering according to the paper ***VBASE: Unifying Online Vector Similarity Search and Relational Queries via Relaxed Monotonicity"*** (*Microsoft Research Asia, 2023*).

VBASE revealed a tremendous acceleration against top K-based systems with minimum changes. On the fly, VBase creates a unified query engine with index traversal at the beginning, the impressive ***Relaxed Monotonicity***, a filter in the middle, and a Termination Check updated again with a "residual" of the relaxed monotonicity.

Thanks to the usage of Rust, ***Pgvecto.Rs*** handles memory management perfectly improving the performance of Postgres. Most importantly for us, it gives a powerful tool for the open-source community that will allow independent developers and companies to fully respect data governance. Hopefully, Pgvecto.Rs will be integrated in more vendors and will be deployed in more solutions.

```
!pip install psycopg2-binary asyncpg sqlalchemy[asyncio] greenlet --upgrade --c
!pip install "pgvecto_rs[sdk]" --quiet
```

```
                                        ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.0/3.0 MB 38.8 MB/s eta 0:00
                                        ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.7/2.7 MB 87.7 MB/s eta 0:00
                                        ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.1/3.1 MB 68.7 MB/s eta 0:00
```

```
from llama_index.core import StorageContext
from pgvecto_rs.sdk import PGVectoRs #Python SDK
'''
```

```
I am using the well-secured ingress platform, Ngrok for tunneling to connect
the local Postgres built via Docker to my Google Colab Pro.
'''
PORT = os.getenv("DB_PORT", 13854) #Ngrok TCP tuneling
HOST = os.getenv("DB_HOST", "7.tcp.eu.ngrok.io") #Ngrok TCP tuneling
USER = os.getenv("DB_USER", "postgres")
PASS = os.getenv("DB_PASS", "mysecretpassword")
DB_NAME = os.getenv("DB_NAME", "postgres")

URL = "postgresql+psycopg://{username}:{password}@{host}:{port}/{db_name}".form
    port=PORT,
    host=HOST,
    username=USER,
    password=PASS,
    db_name=DB_NAME,
)


rms_client = PGVectoRs(
    db_url=URL,
    collection_name="hermes_Q1_2023",
    dimension=1024, # BGE-M3 Dimension of 1024 / Sequence length of 8192
)


rms_vector_store = PGVectoRsStore(client=rms_client)
rms_storage_context = StorageContext.from_defaults(vector_store=rms_vector_stor

rms_index = VectorStoreIndex.from_documents(hermes, storage_context=rms_storage
```

```
Parsing nodes: 100%  36/36 [00:00<00:00, 169.34it/s]
```

```
Generating embeddings: 100%  80/80 [00:08<00:00, 14.77it/
```

🚨 French | Hermès International S.C.A (EPA: RMS) 🚨

```
from typing import ItemsView
from llama_index.core import PromptTemplate

fr_template = PromptTemplate("""
<s>[INST]
Vous êtes un assistant en analyse financière. Répondez à la question en vous bas

---------------------
Contexte: {context_str}
---------------------

Question: {query_str}
Réponse:
[/INST]""")
```

```
rms_query_engine = rms_index.as_query_engine(text_qa_template=fr_template, use_a

response1 = rms_query_engine.query("Quelle est le chiffre d’affaires consolidé a
```

```
llama_print_timings:        load time =   21756.38 ms
llama_print_timings:      sample time =      20.49 ms /    39 runs   (    0
llama_print_timings: prompt eval time =   36272.04 ms /   601 tokens (   60
llama_print_timings:        eval time =   13815.35 ms /    38 runs   (  363
llama_print_timings:       total time =   50239.79 ms /   639 tokens
```

```
print(textwrap.fill(str(response1)))
```

```
 Le chiffre d’affaires consolidé du groupe Hermès au premier semestre
2023 s’élève à 6 698 M€.
```

```
response2 = rms_query_engine.query("Quelle est l'évolution d'Hermès par zone géo
```

```
Llama.generate: prefix-match hit

llama_print_timings:        load time =   21756.38 ms
llama_print_timings:      sample time =      86.31 ms /   165 runs   (    0
llama_print_timings: prompt eval time =   17649.44 ms /   287 tokens (   61
llama_print_timings:        eval time =   59856.75 ms /   164 runs   (  364
llama_print_timings:       total time =   78140.93 ms /   451 tokens
```

```
print(textwrap.fill(str(response2)))
```

```
 Au premier semestre 2023, toutes les zones géographiques d'Hermès ont
enregistré des hausses supérieures ou égales à 20% par rapport à la
même période de l'année précédente. L'Asie a connu une croissance
exceptionnelle en bénéficiant d'une base de comparaison favorable au
deuxième trimestre. Les ventes en magasins du groupe ont augmenté de
25% à taux de change constants, tandis que les ventes en gros ont
connu une hausse de 26%, profitant du rebond des ventes aux voyageurs.
Hermès continue de développer son réseau de distribution exclusif.
```

French: *Correct mais...*

- 2 questions, 2 correct answer

In our experiment in French with Hermès, the vanilla RAG handles perfectly the retrieval of information in text data. On the second question *"Can you give us the breadown of revenues by region for the half 2023?"*, our system doesn't handle well the tabular data. The breakdown of revenues by region is an borderless table on page 5. Please note that Hermès (like many luxury groups) is separating France from Europe, and Japan from APAC to reveal those two historical strategic markets. Talking about Asia as an unified market might be too weak in the luxury industry for equity research.

🍺 German | Dr. Ing. h.c. F. Porsche AG (ETR: P911) 🍺

```
P911_client = PGVectoRs(
    db_url=URL,
    collection_name="porsche_Q1_2023",
    dimension=1024, # BGE-M3 Dimension of 1024 / Sequence length of 8192
)


P911_vector_store = PGVectoRsStore(client=P911_client)
P911_storage_context = StorageContext.from_defaults(vector_store=P911_vector_st

P911_index = VectorStoreIndex.from_documents(porsche, storage_context=P911_stor
```

```
Parsing nodes: 100%|| 46/46 [00:00<00:00, 163.24it/s]
```

```
ge_template = PromptTemplate("""
<s>[INST]
Du bist ein hilfsbereiter Finanzexperte. Beantworte Fragen basierend auf dem ur

---------------------
Kontext: {context_str}
---------------------

Frage: {query_str}
Antwort:
[/INST]""")


P911_query_engine = P911_index.as_query_engine(text_qa_template=ge_template, us


response3 = P911_query_engine.query("Wieviel Umsatzerlöse hat die Porsche AG im
```

```
Llama.generate: prefix-match hit

llama_print_timings:        load time =   21756.38 ms
llama_print_timings:      sample time =      23.72 ms /    45 runs   (    0
llama_print_timings: prompt eval time =   33709.65 ms /   549 tokens (   61
llama_print_timings:        eval time =   18841.17 ms /    44 runs   (  428
llama_print_timings:       total time =   52724.49 ms /   593 tokens


print(textwrap.fill(str(response3)))
```

```
 Die Porsche AG hat im Halbjahres 2023 Umsatzerlöse in Höhe von 20.431
Mio. € erwirtschaftet.


response4 = P911_query_engine.query("Wieviele Fahrzeuge hat die Porsche AG ausg
```

```
Llama.generate: prefix-match hit
```

```
llama_print_timings:        load time =   21756.38 ms
llama_print_timings:      sample time =      22.00 ms /    42 runs   (    0
llama_print_timings: prompt eval time =   20031.24 ms /   500 tokens (   40
llama_print_timings:        eval time =   18053.90 ms /    41 runs   (  440
llama_print_timings:       total time =   38264.37 ms /   541 tokens
```

```
print(textwrap.fill(str(response4)))
```

```
 Im ersten Halbjahr 2023 hat die Porsche AG Konzern 167.354 Fahrzeuge
an Kunden ausgeliefert.
```

German: *Perfekt!*

- 2 questions, 2 correct answers...

In our experiment in German with Porsche AG, the model replied perfectly according to the financial report on the two questions. During testing with offloading, our vanilla RAG could sometimes replied in English.

However, even by being stuck in English, our Mixtral RAG correctly found the 167.354 vehicles on page 5 and impressively ***adjusted the comma*** used in German numeric accounting rules to the English numeric accounting system 167.354 ***with a period***.

🐑 Italian | Brunello Cucinelli S.p.A (BIT: BC) 🐑

```
bc_client = PGVectoRs(
    db_url=URL,
    collection_name="bc_Q1_2023",
    dimension=1024, # BGE-M3 Dimension of 1024 / Sequence length of 8192
)
```

```
bc_vector_store = PGVectoRsStore(client=bc_client)
bc_storage_context = StorageContext.from_defaults(vector_store=bc_vector_store)

bc_index = VectorStoreIndex.from_documents(cucinelli, storage_context=bc_storag
```

```
Parsing nodes: 100%                                                 112/112 [00:00<00:00, 220.71it/
                                                                    s]
```

```
it_template = PromptTemplate("""
<s>[INST]
Lei è un utile assistente finanziario. Rispondete alla domanda in base al conte

---------------------
Contesto: {context_str}
```

```
---------------------

Domanda: {query_str}
Risposta:
[/INST]""")


bc_query_engine = bc_index.as_query_engine(text_qa_template=it_template, use_as


response5 = bc_query_engine.query("Il Brunello Cucinelli conclude il primo seme

    Llama.generate: prefix-match hit

    llama_print_timings:        load time =   21756.38 ms
    llama_print_timings:      sample time =      23.30 ms /    43 runs   (    0
    llama_print_timings: prompt eval time =   34490.39 ms /   580 tokens (   59
    llama_print_timings:        eval time =   15846.85 ms /    42 runs   (  377
    llama_print_timings:       total time =   50506.23 ms /   622 tokens


print(textwrap.fill(str(response5)))

     Il Brunello Cucinelli ha registrato ricavi consolidati di Euro
    543.942 migliaia al termine del primo semestre del 2023.


response6 = bc_query_engine.query("Può spiegare l'obiettivo dell'accordo tra Br

    Llama.generate: prefix-match hit

    llama_print_timings:        load time =   21756.38 ms
    llama_print_timings:      sample time =      58.58 ms /   108 runs   (    0
    llama_print_timings: prompt eval time =   19873.27 ms /   501 tokens (   39
    llama_print_timings:        eval time =   41562.33 ms /   107 runs   (  388
    llama_print_timings:       total time =   61876.17 ms /   608 tokens


print(textwrap.fill(str(response6)))

     L'obiettivo dell'accordo tra Brunello Cucinelli e Chanel è quello di
    rafforzare la filiera italiana del lusso, con Chanel che investe nel
    "Progetto Lanificio Cariaggi" come simbolo di eccellenza e made in
    Italy. Questa collaborazione mira a creare una crescita umana e
    professionale di qualità per entrambe le parti nei prossimi decenni.
```

Italian: *Fantastico!*

- 2 questions, 2 correct answers

In our experiment in Italian with Brunello Cucinelli, our Mixtral RAG surprisingly found the solution for the first question of 543,9 in the column chart without even a title, or mention of the italian word "ricavi (revenues)". The full amount is stated 543.942 millions with "il fatturato consolidato del Gruppo (the consolidated turnover/sales of the group)".

My second question is rather tricky by being extremely broad and abstract. *"Can you explain the objective of the parternship between Brunello Cucinelli and Chanel?"*. Our Mixtral RAG gives good explanation of the investment between Brunello Cucinelli and the reasoning behind the investment in Cariaggi Lanificio S.p.A. for their respective supply chains. It even quoted the Chairman for the name of the project "Progetto Lanificio Cariaggi".

- Technical observations ⚠

Positive observations: the combination of ***BGE-M3*** for the embedding model, ***PgVecto.Rs*** for the vector search on Postgres database, ***Mixtral 8x7B*** for the multilingual LLM, and ***LLama-Index*** for the orchestration provides sublime performance. This open-source Mixtral RAG could empower many projects ***to circumvent proprietary systems***, and ***empower data governance*** in order to ***respect the EU Artificial Intelligence Act***.

Negative observations: LlamaCPP is an amazing tool for experimentation, but it might be too brittle for production. The loading of parameters (gpu layers/batch size) are giving different outputs, sometimes responses in English. ***The inferences on a MoE architecture with a basic Nvidia T4 GPU are high quality, yet rather slow***. The real difficulty of the notebook was the so-called ***"prompt engineering"*** or finding correct prompt templates in French, Italian, and German. One change of word in the prompt template could drastically change the quality of the response.

- Partial conclusion, future development 🏁

Many benchmarks do not reflect the actual linguistic performance of LLMs. More than open-washing, we have a lot of “benchmarketing”. ***By being available fully Open-source***, Mixtral 8x7B enables students, developers, and SMEs in Europe to experience a powerful model and its variations. The cost of development and heavy computing power are the two remaining roadblocks.

According to OpenAI business terms, data created by OpenAI's models couldn't be used commercially. Under the Apache 2.0 license, this Mixtral gives us a powerful tool ***to build synthetic data for many domain-specific datasets.***

Their paper ***“Mixtral of Experts”*** from January 2024 doesn’t reveal much about the dataset used during training. Magic of a European team, Mixtral 8x7B outperforms LlaMA 2 70B in 4 languages: French, German, Spanish, and Italian, but not in English. I am wondering if Mistral A.I team will create models of complex languages like Chinese, Japanese, Greek, Farsi, Arabic, etc or focus only on European languages to lead the European market.

Mixtral is indeed impressive in English and has also ***pretty amazing capabilities in French, German, Spanish, and Italian***. The Sparse Mixture of Experts (SMoE) language model could be the defacto architecture for 2024. In the same vision of multiple small structures to create

a powerful one, we are also seeing the rise of the ***"Merger of LLMs"***. The great concept of ***"Mindstorms" for Natural Language-Based Societies of Mind (NLSOMs)*** demonstrates incredible engineering prowess.

Thank you for reading!

Please feel free to contact me if you have any questions.

Akim Mousterou

Disclaimer: *None of the content published on this notebook constitutes a recommendation that any particular security, portfolio of securities, transaction, or investment strategy is suitable for any specific person. None of the information providers or their affiliates will advise you personally concerning the nature, potential, value, or suitability of any particular security, portfolio of securities, transaction, investment strategy, or other matter.*