Fully open-source RAG for Japanese in a low-resource environment

低コスト環境でも処理可能な日本語対応完全オープンソースRAG

According to ***Goldman Sachs Research*** in April 2023, Generative AI could raise global GDP by `7%` (or almost `$7` trillion). On a total addressable market of `$685` billion for the global software industry, the generative AI software market is estimated to be `$150` billion. Large Language Models (LLMs) are eating software.

Nevertheless, autoregressive LLMs alone have three main problems. LLMs are sometimes generating non-factual or nonsensical statements called ***"stochastic parrots"***, often called by an anthropomorphism ***"hallucination"***. LLMs are trained on 7 billion to 120 billion parameters, requiring a huge amount of computing power and energy. Finally, companies are reluctant to leverage LLM solutions due to the governance of data and privacy concerns.

The current best Generative AI solutions are using LLMs helped by the information retrieval framework called ***Retrieval-Augmented Generation (RAG)***. RAG is dynamically augmenting the generative capabilities of LLMs with external or/and proprietary knowledge. A real open-source RAG coupled with open-source LLM could minimize stochastic parrots while optimizing costs and maintaining data privacy for a real democratization of Generative AI.

This notebook showcases a quick implementation for open-source RAG and LLM in a low-resource environment for Japanese.

- The first element of this RAG is ***Llama-Index*** with vanilla *Hybrid search* (combining retrieval for text search and vector search)
- The second element is a Japanese LLM "ELYZA-japanese-Llama-2-7b-instruct" created by Japanese startup, ***Elyza Inc.***
- The third element is an open-source database ***PostgreSQL*** transformed into a vector database by the library ***PGVector***

As a dataset, this notebook explores 5 documents from the public hearings "AI Strategy for 2023" created by the ***Cabinet Office*** in Japan available here (https://www8.cao.go.jp/cstp/ai/ai_senryaku/ai_senryaku.html)

In the first part, I will explain the technical details of an open-source RAG. In the second part, I will try to explain the business implications of the "token economy". In the last part, I will try

to give a threefold perspective of business innovation and economic competitiveness between the USA, France, and Japan.

```
!pip install git+https://github.com/huggingface/transformers --quiet
!pip install "llama-index==0.8.63.post2" --quiet
!pip install accelerate safetensors bitsandbytes --upgrade --quiet
!pip install sentence_transformers cohere pdfminer.six --upgrade --quiet
```

```
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
```

```
import nest_asyncio
nest_asyncio.apply()
```

- Llama Index | ラマ・インデックス

Llama-Index is a key data framework for a structured retrieval-augmentation generative. Formerly known as GPT-Index, Llama-Index created by ***Jerry Liu*** has been quickly morphing into a model-agnostic framework.

While ***LangChain*** might be easier for beginners, Llama-Index gives a great lower-level API for robust indices or metadata, advanced retrieval, or query interface. Informational retrieval is nothing new, the BM25 retriever is derived from TF-IDF, and Re-Ranker is derived from cross-encoder.

To manage the tradeoff between accuracy and speed, Llama-Index enables us to optimize fundamental components often disregarded in my opinion, a hierarchical ingestion of data and metadata filters.

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

from llama_index import ServiceContext, StorageContext
from llama_index.indices.vector_store import VectorStoreIndex
from llama_index.vector_stores import PGVectorStore
import textwrap

import tqdm, ipywidgets
from IPython.display import Markdown, display
```

- Vector embeddings | 埋め込み

Tokenization has evolved quickly on the last 10 years from character-based, to sub-word units in 2015, to SentencePiece by ***Taku Kudo*** in 2018, and SentenceTransformers by ***Nils Reimers*** in 2019. We now have not word-level or sentence-level but massive text embedding

Reimers in 2019. We now have not word level or sentence level but massive text embedding in more than 100 languages in 2023. Embeddings are a key building block of large language models. For RAG, text embeddings are important for semantic search to calculate similarity metrics such as *cosine distance, inner product, L2 (Euclidean distance),* or *L1 (taxi cab or Manhattan distance)* between various query embeddings and various corpus embeddings.

According to the ***MTEB Benchmark***, we have a mix of proprietary embeddings (***OpenAi, Cohere, VoyageAI***) and open-source embeddings (***BAAI, Infloat, Instructor, Jina***) sometimes of equal performance. Embedding models are the lock on the usage of LLMs and an important factor in the quality of inference of RAGs.

With an open-source perspective in Japanese, we will use the great multilingual ***"Multilingual-e5-large"*** with more than 100 languages by ***Dr. Furu Wei*** and his team at ***Microsoft Research Asia***. Based on the paper, *"Text Embeddings by Weakly-Supervised Contrastive Pre-training"*, ***multilingual-e5-Large*** has 24 layers and an embedding size of 1024. Performances are great with 62.5 for Japanese and 70.5 on average mean reciprocal rank (Avg MRR). However, for a size of 2.24GB, the ***multilingual-e5-Large*** is consuming a lot of computing power.

```
from llama_index.embeddings import HuggingFaceEmbedding
embed_model = HuggingFaceEmbedding(model_name="intfloat/multilingual-e5-large")
```

- Quantized LLM | 量子化の大規模言語モデル

From a perspective of low-resource, I will load the model in a quantized version of 4-Bits by leveraging ***Bitsandbytes*** and ***Accelerated*** by ***Hugging Face***. Developed by ***Tim Dettmers*** of the *University of Washington*, Bitsandbytes is optimizing 4-bit CUDA functions from ***Nvidia*** for the ***PyTorch*** framework.

Tim Dettmers' research on low-bit quantization of large language models and hardware-optimized deep learning has been the real factor of democratization for LLM. Another actor of low-resource LLM, ***Tom Jobbins (The Bloke)*** has been releasing many quantized models financed by the venture capital firm, ***Andreessen Horowitz***.

```
import torch
from transformers import BitsAndBytesConfig

nf4_config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_quant_type="nf4",
    #bnb_4bit_use_double_quant=True  #Nested Quantize is possible to save even
)
```

- LLM from Elyza | 大規模言語モデル、イライザ

According to the ***Rakuda Ranking***, I chose the ***"ELYZA-japanese-Llama-2-7b-fast-instruct"*** from Tokyo-based startup ***Elyza, Inc***. Elyza is based on the English model, ***LlaMA-2*** developed by ***Facebook/Meta AI*** with a community license. The AI team of Elyza has expanded the vocabulary size in Japanese for better inference time. This open-source model at only 7B could be compared to GPT-3.5 in Japanese.

```
from llama_index.llms import HuggingFaceLLM

llm = HuggingFaceLLM(
    model_name="elyza/ELYZA-japanese-Llama-2-7b-instruct",
    tokenizer_name="elyza/ELYZA-japanese-Llama-2-7b-instruct",
    context_window=3900,
    max_new_tokens=1024,
    model_kwargs={"quantization_config": nf4_config},
    generate_kwargs={"temperature": 0.1,"do_sample": True},
    device_map="auto")
```

```
Loading checkpoint shards:  100%  2/2 [00:11<00:00, 5.28s/it]
```

```
from llama_index.text_splitter import SentenceSplitter
from llama_index.node_parser import SimpleNodeParser

text_splitter = SentenceSplitter(
    chunk_size=512,
    chunk_overlap=20,
    paragraph_separator="\n\n\n",
)
node_parser = SimpleNodeParser.from_defaults(text_splitter=text_splitter)

service_context = ServiceContext.from_defaults(llm=llm,embed_model=embed_model,
```

- CJKPDFReader | CJKPDFリーダー

For the specificity of Japanese a particular PDF loader is required for shift-jis encoded documents. The Llama-Hub offers ***CJKPDFReader*** a document for Japanese, Chinese, and Korean based on pdfminer.six.

Unfortunately, we encounter multiple problems due to the formatting of the administrative documents that will create problems with embedding, indexing, and final results.

- Titles are neither in Japanese, nor in English. ***"Giji"*** is a Hepburn reading of **"議事"** proceedings (e.g. government proceedings). The second title is slightly better with ***"gijiyoushi2kai"*** but still wrong as Hepburn of **"議事要旨第2回** (proceedings second meeting)
- The administrative document is dated with the Japanese era calendar scheme as **"令和**

The administrative document is dated with the Japanese era calendar scheme as "令和**5 年** (era number 5 of Reiwa)" as 2023

- The Japanese language doesn't have white space. However, the formatting of the Word file and the transformation into PDF with Acrobat Maker 23 (library 23.1.206) is creating space between words
- In the very important first-page introduction participants such as academic researchers, company leaders, and government officials are presented in a justify and minimalistic way (creating space)

As the financial sector is heavily regulated, even transcripts of earning calls are carefully structured, perfect for metadata such as title, date, company, board of directors, and core chapters. In the USA, the ***Securities and Exchange Commission (SEC)*** required a clear and strict format for 10-Q, 10-K, and 8-K. In France, the reporting format is getting more regulated with a universal registration document (URD) by the ***European Securities Markets Authority (ESMA)***, while in Japan the ***Financial Services Agency (FSA)*** is producing guidelines and starting in 2019 self-regulatory activities with the ***"Open Policy Lab"*** initiative.

```
from pathlib import Path
from llama_index import download_loader

CJKPDFReader = download_loader("CJKPDFReader")
loader = CJKPDFReader()

giji_01 = loader.load_data(file=Path('/content/giji.pdf'))
giji_02 = loader.load_data(file=Path('/content/gijiyoushi2kai.pdf'))
```

```
!pip install psycopg2-binary pgvector asyncpg "sqlalchemy[asyncio]" greenlet --
```

- Vector database | ベクトルデータベース (PostgreSQL + PGVector)

The boom in Generative AI created a spark in the vector database space with popular startups ***Chroma DB, Qdrant, Zilliz, Pinecone,*** etc. but also major players such as ***Elasticsearch, Redis, MongoDB,*** etc.

Powered by the open-source community, ***PostgreSQL*** enabled vector similarity search library ***PGVector*** developed by ***Andrew Kane***. The combination of Postgres and PGVector offers a self-hosted and cloud, enabled hybrid search, store documents, and work in async. The installation is very easy with a Docker image.

- Hierarchical Navigable Small World | 階層的なスモールワールド

PGVector is compatible with the ***IVF Flat (Inverted File with Flat Compression)*** format and the ***HNSW (Hierarchical Navigable Small World)*** format released in August 2023 (version 0.50). HNSW is an approximate K-nearest neighbor search based on navigable small world graphs among top-performing indexes for vector similarity search developed by ***Dmitry Yashunin*** and ***Yury Malkov***. HNSW is slower to build but reduces memory requirements.

Postgres is fully open-source, ACID compliant, implemented in Japanese via ***MeCab***, and cost-efficient. In more than 20 vector databases available with LLama-Index only ***Elasticsearch, Pinecone, Weaviate, Postgres, Azure Cognitive Search, TencentVector DB, MyScale,*** and ***Lantern*** can use hybrid search.

Among them, only ***Postgres + PGVector*** is truly open-source and can leverage the cutting-edge graph-based algorithm of ***HNSW***.

```
import psycopg2

connection_string = "postgresql://postgres:test@0.tcp.eu.ngrok.io:19642" #Ngrok
db_name = "vector_db"
conn = psycopg2.connect(connection_string)
conn.autocommit = True

with conn.cursor() as c:
    c.execute(f"DROP DATABASE IF EXISTS {db_name}")
    c.execute(f"CREATE DATABASE {db_name}")
```

- Ngrok | エングロック

I am using the well-secured ingress platform, ***Ngrok*** for tunneling to connect the local Postgres built via ***Docker*** to my Google Colab Pro.

```
from sqlalchemy import make_url
from llama_index.vector_stores import PGVectorStore

url = make_url(connection_string)
hybrid_vector_store = PGVectorStore.from_params(
    database=db_name,
    host=url.host,
    password=url.password,
    port=url.port,
    user=url.username,
    table_name="ai_law_japan",
    embed_dim=1024,
    hybrid_search=True,
    #text_search_config="japanese", #Thx to MeCab integration into Postgres
)
```

[illegible]

- Hybrid approach of RAG | RAGシステムのハイブリッド検索

- Text plus vector search | テキスト検索とベクトル検索

I believe that a ***hybrid search*** approach should be mandatory for proper RAG. The hybrid option must be enabled at the construction of the ***PGVector*** store and stated in the query engine to combine retrieval from both text search and vector search. The implementation of a hybrid search is straightforward. It can be improved in many aspects.

```
storage_context = StorageContext.from_defaults(vector_store=hybrid_vector_store
```

```
from llama_index.tools import QueryEngineTool, ToolMetadata
from llama_index.schema import MetadataMode

giji_01_hybrid_index = VectorStoreIndex.from_documents(giji_01,storage_context=

giji_02_hybrid_index = VectorStoreIndex.from_documents(giji_02, storage_context
```

- Prompt engineering | プロンプトエンジニアリング

Llama-Index provides great prompt templates, unfortunately only in English. For Japanese, we can override the template with a Q&A template for example. I used many different templates, it was rather difficult to have a clear understanding. ***Prompt engineering feels more like trial and error than programming***. I used the prompt guidelines of Llama 2 mixed with the default prompt system by Elyza for Japanese. Asking for responses in bullet points (マークダウン形式) gave me better results.

```
from llama_index.prompts.prompts import QuestionAnswerPrompt

qa_template = QuestionAnswerPrompt("""
<s>[INST] <<SYS>>
あなたは誠実で優秀な日本人のアシスタントです。
マークダウン形式で以下のコンテキスト情報を元に質問に回答してください。
<</SYS>>

{context_str}

{query_str}
[/INST]""")
```

```
giji_01_hyb_engine = giji_01_hybrid_index.as_query_engine(vector_store_query_mo
                                                          sparse_top_k=2,
                                                          text_qa_template=qa_templ
                                                          )

giji_02_hyb_engine = giji_02_hybrid_index.as_query_engine(vector_store_query_mo
```

```
    sparse_top_k=2,
    text_qa_template=qa_templ
)
```

- Q&A over documents | データを回答させてみる

```
response1 = giji_02_hyb_engine.query("人工知能関連の政策を議論する内閣府の「第2回AI戦略
print(response1)
```

構成員は以下の通りです。

- 座長：松尾豊　東京大学大学院工学系研究科教授
- 構成員：江間有沙　東京大学未来ビジョン研究センター准教授、岡田淳　東京大学大学院工学系研究
- 政府側参加者：岸田文雄　内閣総理大臣、高市早苗　科学技術政策担当大臣、尾崎正直　国務大臣（

なお、議事録は作成されておりません。

Question 1: Who are the members of the second AI strategy meeting?

Answer 1:Correct! (正解!)

The first page doesn't give a relation between the full name and the company. The impressive RAG understood the linear relation between researchers and CEOs to universities or companies. The RAG gave a full list with chairman, the members, and the government officials.

```
response2 = giji_01_hyb_engine.query("第1回AI戦略会議の座長は誰でしょうか？")
print(response2)
```

第1回AI戦略会議の座長は、松尾豊氏です。

Question 2: Who was the chairman of the first AI strategy meeting?

Answer 2: Correct! (正解!)

Even with the double space created by the formatting, the chairman is 松尾豊 (Matsuo Yutaka) is correctly found by the LLM. The model deleted the white space between the family name "松尾(Matsu)" and the personal name "豊(Yutaka)". It even introduced the honorific particle "氏" to be polite in the answer.

```
response3 = giji_02_hyb_engine.query("株式会社ソニーリサーチの代表取締役は誰ですか？")
print(response3)
```

質問者さんにお返事いたします。

株式会社ソニーリサーチの代表取締役は、北野宏明さんです。

Question 3: Who is the representative director of Sony Research?

Answer 3: Correct! (正解!)

The model is finding correctly the CEO of Sony Research, Mr. Kitano Hiroaki (北野宏明) regardless of the space.

```
response4 = giji_02_hyb_engine.query("G7サミットで議論する広島AIプロセスの目的を説明し
print(response4)
```

広島AIプロセスの目的について、G7サミットで議論するために作成した暫定的な論点整理案に基づ

1. 我が国が世界に誇るべきAIの力を活用し、課題を解決し、社会の課題を解決し、国民の安心・安
2. 国民のAIの理解を深め、公正な社会の実現に向けた議論を深める。
3. 国際社会でのAIの活用における課題を共有し、国際的なルールの枠組み作りに貢献する。
4. 我が国のAI戦略の基本方針を踏まえ、G7広島AIプロセスで議論を深め、我が国がAI戦略会議の主

Question 4: Can you explain the Hiroshima A.I Process from G7 summit?

Answer 4: Correct! (正解!)

The answer is correct. The model does explain the ***"Hiroshima AI Process"*** is a comprehensive policy framework. The reply is pretty great but I would love to know "Hiroshima AI Process" is a in partnership with the ***Global Partnership for Artificial Intelligence (GPAI)*** and the ***Organisation for Economic Co-operation and Development (OECD)*** that will take place in Hiroshima in September 2023.

```
response5 = giji_02_hyb_engine.query("ITサービスの貿易赤字はいくらですか？")
print(response5)
```

質問に回答いたします。

ITサービスの貿易赤字は、2030年には10兆円近く生じると予測されています。

Question 5: Can you tell me the trade deficit for the IT service?

Answer 5: Incorrect! (不正解!!!)

The model is giving back the 10 trilion of yen. Correct answer is that the Japanese trade deficit of IT (3 trilion of yen) and Governement cloud (8 trilion yen). The Minister of Internal Affairs and Communications, ***Mr. Matsumoto*** complained about government spending for ChatGpt of OpenAI, when a similar system could be created by local companies.

```
response6 = giji_02_hyb_engine.query("里見経済産業大臣政務官の結論を要約してください。"
print(response6)
```

与えられたコンテキスト情報を元に質問に回答します。

里見経済産業大臣政務官は、AIの利活用による産業競争力の強化を重要視しています。具体的には、

Question 6: Can you summarize the conclusion of Mr. Satomi of the Parliamentary Vice-Minister for Economy, Trade and Industry?

Answer 6: 正解 | Correct!

The answer is correct despite that I didn't put in place a "Document Summary Index" dedicated to summarization in Llama-Index.

- Technical observations | 技術ノート

5 good answers to 6 questions with a fully open-source RAG project (7B LLM quantized at 4-Bits) and low resource setup (T4 Nvidia GPU) is quite amazing. By fine-tuning embedding, optimizing retrieval, metadata filters, or organizing document hierarchies, I believe it would achieve a perfect score.

8-Bits is still fitting on the T4 Nvidia GPU and give better responses. By instantiating at higher floating-point precision such as 16 bits (lower precision) with a V100 Nvidia and 32 bits (normal precision) with a more powerful A100 GPU, it will use more memory, and cost more but enable quicker and better responses.

- Business implications | ビジネスインパクト分析

As of November 2023, we are seeing a pricing war among proprietary systems to reduce barriers to entry and engage the adoption of Generative AI solutions. Newcomer ***Voyage AI*** co-created by an associate professor at ***Stanford***, ***Tengyu Ma*** is cost-cutting with `$0.1` (per million tokens) on a maximum context length of 4096 (first 50 million tokens are free as part of a trial period). As a comparison, ***GPT-4 of OpenAI*** `$0.03` for inputs and `$0.06` for outputs (per 1K tokens) as of November 2023.

Using an open-source embedding model breaks the “token-based business model” giving total freedom to experiment and scale. Alternatively, open-source embedding models created by the university could finance PhD students or professors working at AI labs.

According to the price of Replicate, the Nvidia T4 GPU is at `$0.000225`/sec (`$0.81`/hour) for less than `$600` per month, or `$7,200` per year to run on 4bits or 8bits quantized LLM. PostgresSQL is probably the most stable and available open-source database in the world. A fully open-source RAG can empower software, machine learning, or NLP engineers supported by the open-source community.

- Generative A.I strategy for U.S.A, France, and Japan | アメリカ、フランス、日本の生成ＡＩ戦略

In the USA, since 2021, the success of GPT-3 by OpenAI reoriented the research towards

autoregressive architecture and opened a wave of huge investments in the private sector. At ***Y Combinator***, more than 59 generative AI-focused startups (22%) in the 272 companies in the Winter 2023 cohort.

In France, the most successful AI startup Hugging Face (built by 3 French entrepreneurs) was financed by American venture capital firms in 2019. In the latest Series D funding, in August 2023, Hugging Face raised `$235` million led by tech giants such as ***Google, Amazon, Nvidia, etc*** for a `$4.5` billion valuation. The ***Confiance.AI*** strategy of the French government focuses on the excellence of higher education and partnerships with century-old groups such as ***Airbus, Thales, Renault, Air Liquide, etc.***

The powerful ecosystem of Silicon Valley, cool management style, and high salaries are attracting top talents from around the world in California. France is more or less a laggard in AI-related investments and is the first provider of engineers for leading AI companies.

Due to the language barrier, Japan has a hybrid ecosystem composed of academic labs of prestigious universities, upcoming startups such as ***Elyza, Stockmark, Rinna,*** and tech giants such as ***NTT Data A.I labs, Line A.I (Naver), Megagon Labs (Recruit), CyberAgent (Abema), etc.***

Back in 2021, the popular Bert models of *Tohoku University* were created with Cloud TPUs provided by the *TensorFlow Research Cloud program* of ***Google Inc***. The most prolific Japanese researcher is ***Tatsunori Hashimoto*** at *Stanford University*, while the most influential Japanese researcher is ***Taku Kudo*** (creator of ***MeCab, CaboCha, and CRF++***) at ***Google Inc***.

In 2023, the ***city of Yokosuka*** (Kanagawa Prefecture) in Japan was the first city to use ChatGPT for administrative tasks. The creator of popular anime One Piece, ***Eiichiro Oda*** has used ChatGPT 4.0 to create a new manga chapter. This extremely fast adoption of autoregressive models sparked a copyright debate in Japan. According to ***Professor Tatsuhiro Ueno*** of *Waseda University*, the Japanese Copyright Act has an explicit provision on copyright exception for text and data mining (Art.47septies 著作権法 - 美術の著作物等の展示に伴う複製等 - 第四十七条）under which it shall be allowed to copy any copyright-protected work to promote machine learning without authorization of copyright holders.

The merger ***LY Corporation*** (LINEヤフー株式会社) between ***LINE*** and ***Yahoo Japan*** could be a game changer in the creation of corpus in Japanese. Since embedding models of *Kindai University* and *Nagoya University* for Japanese are still based on SentenceTransformers architecture, a bilingual embedding model in Japanese and English could be a great open-source project to eliminate the first bottleneck.

Consisting of over 500 participants including researchers in natural language processing and computer systems from universities and corporations, ***LLM Study Group*** has been an extraordinary project for Japan that unveiled a great model called LLM-JP 13B. The LLM Study Group can be comparable to the ***BigScience*** project led by ***Hugging Face*** with the

Bloom models.

By reading the guidelines of the French and Japanese governments, the words ***"open-source"*** were never used as a viable solution for promoting innovation and economic development that could be implemented at a large scale. Open source ensures that AI progress benefits all, in the government for administrative tasks, in small and medium companies for business applications, and for research in graduate schools or public hospitals.

Japanese language specificities and academic excellence could give an advantage to local startups but the clock is ticking.

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.*