


```
# 1) Cài đặt
!pip install sentence-transformers langchain chromadb pypdf
```

 [Hiện kết quả đã ẩn](#)

```
!pip install -U datasets
```

 [Hiện kết quả đã ẩn](#)

```
# 2) Tải & tiền xử lý
from datasets import load_dataset
ds = load_dataset("clapAI/vietnamese-law-corpus", split="train").shuffle(seed=42).select(range(2000)) # demo nhỏ
docs = [d["content"] for d in ds]
```

 /usr/local/lib/python3.11/dist-packages/huggingface\_hub/utils/\_auth.py:94: UserWarning:  
The secret `HF\_TOKEN` does not exist in your Colab secrets.  
To authenticate with the Hugging Face Hub, create a token in your settings tab (<https://huggingface.co/settings/tokens>), set it as secret.  
You will be able to reuse this secret in all of your notebooks.  
Please note that authentication is recommended but still optional to access public models or datasets.

```
warnings.warn(
  README.md:      1.06k/? [00:00<00:00, 99.0kB/s]

(...)00000-of-00003-ae41a8f817340e7f.parquet: 100%      85.3M/85.3M [00:00<00:00, 172MB/s]
(...)00001-of-00003-48160c270fdda591.parquet: 100%      103M/103M [00:00<00:00, 172MB/s]
(...)00002-of-00003-f3bce523b10adacf.parquet: 100%     110M/110M [00:00<00:00, 198MB/s]

Generating train split: 100%      514936/514936 [00:05<00:00, 130353.96 examples/s]
```

```
# 3) Chunk văn bản
from langchain.text_splitter import RecursiveCharacterTextSplitter
spl = RecursiveCharacterTextSplitter(chunk_size=512, chunk_overlap=128)
chunks = sum([spl.split_text(t) for t in docs], [])
```

```
!pip install -U langchain-community
```

 [Hiện kết quả đã ẩn](#)

```
# 4) Tạo embeddings
from langchain.embeddings import HuggingFaceEmbeddings
emb = HuggingFaceEmbeddings(model_name="sentence-transformers/paraphrase-multilingual-mpnet-base-v2")
all_embeddings = emb.embed_documents(chunks)
```

```
# 5) Lưu vào VectorDB (Chroma local)
import chromadb

# create (or open) the collection
client = chromadb.PersistentClient(path="./law_db")
vectordb = client.create_collection(name="vn_law", get_or_create=True)
```

```
# prepare IDs
ids = [f"id_{i}" for i in range(len(chunks))]

# pick a batch size safely under the reported max (5 461)
BATCH_SIZE = 5000
```

```
for start in range(0, len(chunks), BATCH_SIZE):
    end = min(start + BATCH_SIZE, len(chunks))
    vectordb.add(
        ids=ids[start:end],
        documents=chunks[start:end],
        embeddings=all_embeddings[start:end],
    )
    print(f"Added batch {start}-{end-1}")
```

```
print("✓ All chunks added successfully.")
```

 [Hiện kết quả đã ẩn](#)

```
!pip install -U langchain-huggingface
```

 [Hiện kết quả đã ẩn](#)

```
from huggingface_hub import login
```

```
# Nếu bạn đã set HUGGINGFACEHUB_API_TOKEN trong env thì login() sẽ tự dùng token đó.
# Nếu chưa, nó sẽ yêu cầu bạn paste token vào.
login(new_session=False)
```



```
# 6) Xây RAG chain
# 1) Tạo một Hugging Face text-generation pipeline cho bloomz-560m
from transformers import AutoTokenizer, AutoModelForCausalLM, pipeline
import torch
```

```
model_id = "bigscience/bloomz-560m"
tokenizer = AutoTokenizer.from_pretrained(model_id)
model = AutoModelForCausalLM.from_pretrained(model_id).to(
    "cuda" if torch.cuda.is_available() else "cpu"
)
```

```
hf_pipe = pipeline(
    task="text-generation",
    model=model,
    tokenizer=tokenizer,
    device=0 if torch.cuda.is_available() else -1,
    max_new_tokens=256,
    do_sample=False,
)
```

```
# -----
# 2) Bọc thành LangChain LLM
from langchain.llms import HuggingFacePipeline

llm = HuggingFacePipeline(pipeline=hf_pipe)
```

```
# -----
# 3) Xây RetrievalQA chain như trước
from langchain.chains import RetrievalQA

qa_chain = RetrievalQA.from_chain_type(
    llm=llm,
    retriever=vectordb.as_retriever(search_kwargs={"k": 4}),
    return_source_documents=True,
)
```

 [Hiện kết quả đã ẩn](#)

```
# 7) Truy vấn
query = "Thời hiệu khởi kiện hợp đồng dân sự là bao lâu?"
print(qa_chain(query)["result"])
```

 [Hiện kết quả đã ẩn](#)

