Revolutionizing Legal Contracts: The Neural Search & QA Solution

Bridging the Gap between Law and Technology with a Twist of Fun

Meet the team

The Meowsterminds ✨😈

As a pair of enthusiastic engineers, we are striving to make a difference in the field of law.

Smiral

Rishav

Revolutionizing Legal Contracts with Neural Search & QA interface

Introducing a groundbreaking solution for the legal industry.

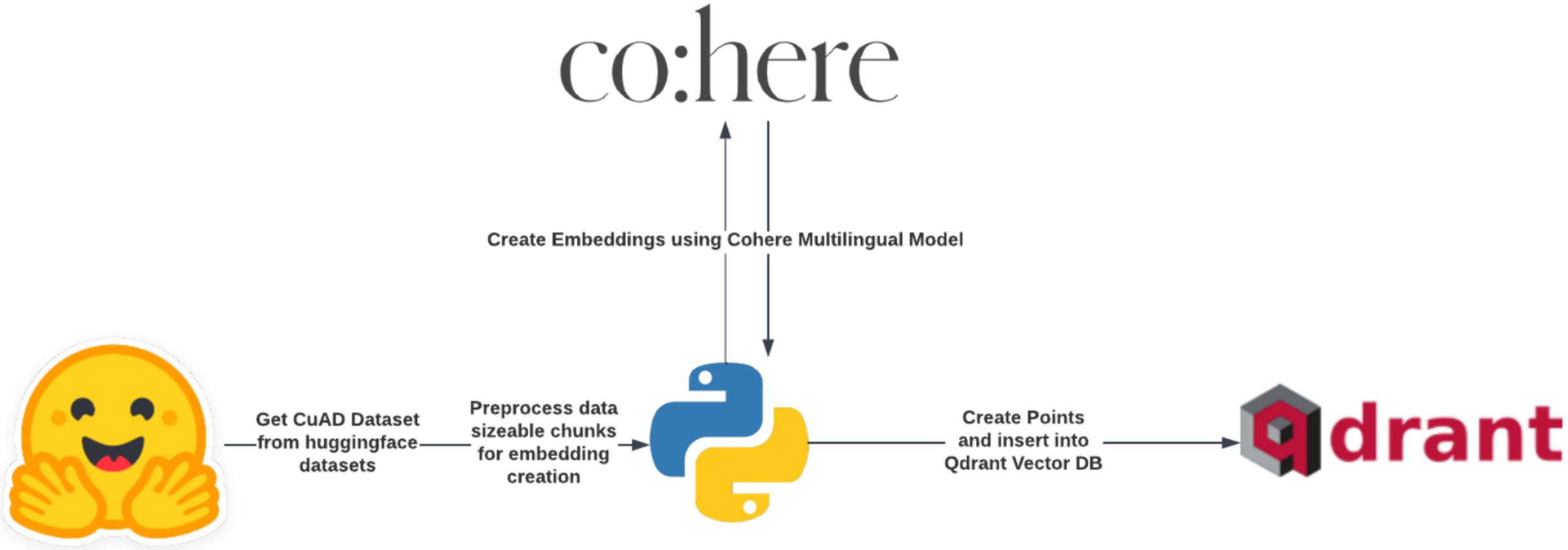
Join us in transforming the way legal contracts are managed and searched through cutting-edge AI technology.

Problem Statement

- The legal industry is facing challenges in efficiently managing and accessing large volumes of legal contracts and data.
- Lawyers and legal professionals spend a significant amount of time manually searching and reviewing legal contracts, which is time-consuming and prone to errors.
- Legal contracts are often written in complex legal language that can be difficult for non-legal professionals to understand.
- The lack of efficient solutions for legal contract management and review can result in legal disputes and financial losses for businesses.

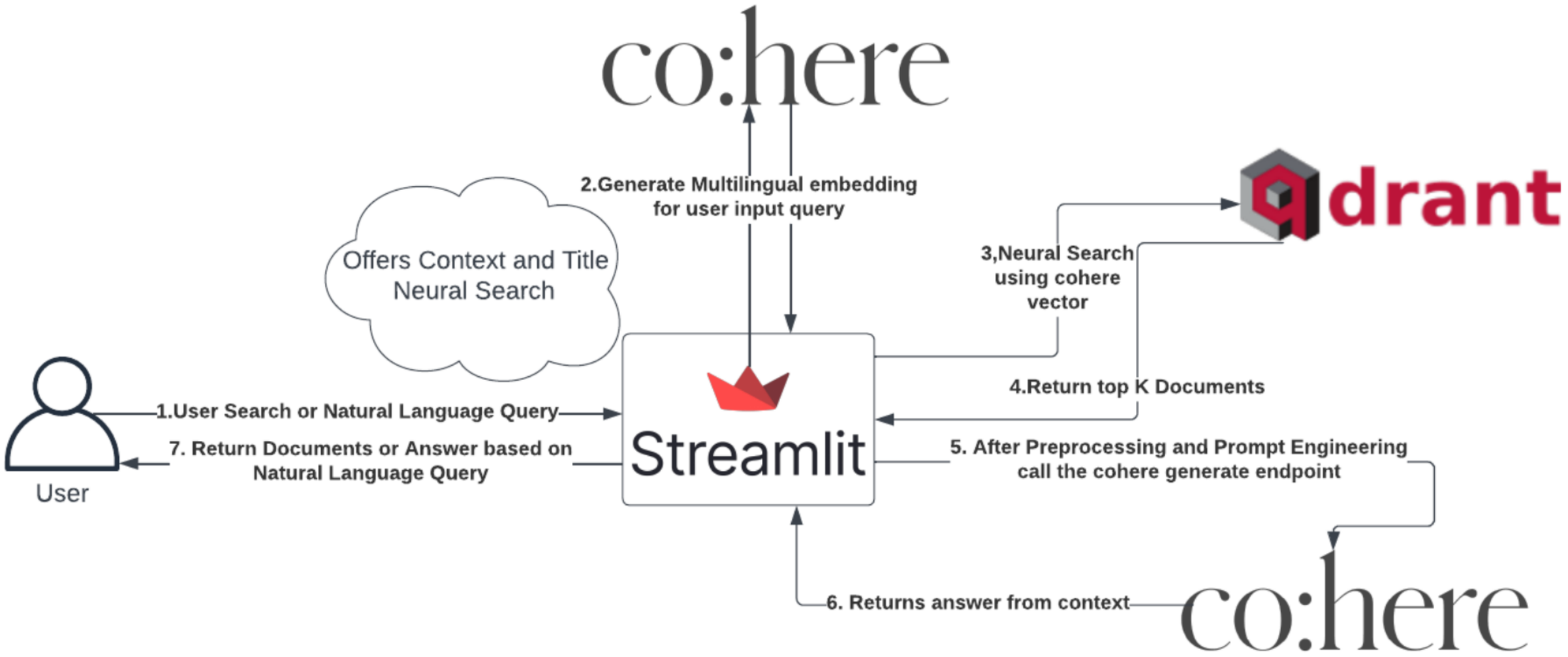
Solution Overview

Overview of the Legal Contracts Neural Search & QA solution


co:here
Create Embeddings using Cohere Multilingual Model
Get CuAD Dataset from huggingface datasets
Preprocess data sizeable chunks for embedding creation
Create Points and insert into Qdrant Vector DB
qdrant

Data Ingestion Process

1. We get a sample dataset from huggingface which is CUAD. This has legal contracts with title and context
2. We preprocess and break the contract context to chunks for creation of embeddings
3. We call Cohere Multilingual model to create sentence embeddings of the context chunks
4. We index the embeddings with original document as payload to retrieve later on the app.


co:here
2.Generate Multilingual embedding
for user input query
Offers Context and Title
Neural Search
qdrant
3,Neural Search
using cohere
vector
4.Return top K Documents
1.User Search or Natural Language Query
7. Return Documents or Answer based on
Natural Language Query
Streamlit
5. After Preprocessing and Prompt Engineering
call the cohere generate endpoint
User
6. Returns answer from context
co:here

LegalFruit Search !

1. User selects options from the frontend streamlit dashboard such as title search or context search. They also have the option to input natural language queries instead of search queries.
2. Once a query has been initiated based on selected options, we create a query vector by calling Cohere's Multilingual Embedding model.
3. This Vector can be used to initiate a neural search on Qdrant DB to retrieve top documents. Based on user selected options these documents can be presented on the frontend or can be sent for further processing
4. Later the documents retrieved are preprocessed , chunked and fit in and engineered prompt to call Cohere's generate endpoint to generate Natural Language Answer which is then sent to frontend

Use Cases

- Legal research: To search for relevant legal documents related to a particular case or topic.
- Due diligence: To assess legal risks related to a potential business transaction or investment opportunity.
- Compliance: To ensure compliance with legal requirements by searching for relevant legal documents related to regulations or laws.
- Contract management: To manage contracts more effectively by searching for and retrieving legal documents related to contracts between parties.
- Education and training: To aid aspiring lawyers to gain a better understanding of the law by searching for legal documents related to a particular subject or topic.

DEMO

Future improvements and Scope

- Having a Real-time Doc Upload mechanism which uploads the data to a datastore like s3 and sends a real time message using kafka for OCR Extraction and indexing to Qdrant DB.
- Frontend can be refactored to a better UX with React. Backend can be separated by creating an API on FastAPI
- Better prompt engineering can help gather better outputs.
- Context length is usually a blocker due to limited performance by LLM , engineering to reduce context length by either summarizing or refining of outputs can be done to gather better context.