Offer Search App

Dependencies

- Anaconda or Miniconda
- Python = 3.10
- PyTorch = 2.1.0
- More in requirements.txt

> Ideal setup requires a Graphic Card

Installation

1. Clone repo

```
git clone https://github.com/95anantsingh/search-app.git
```

2. Create conda environment

```
conda env create -f env.yml
```

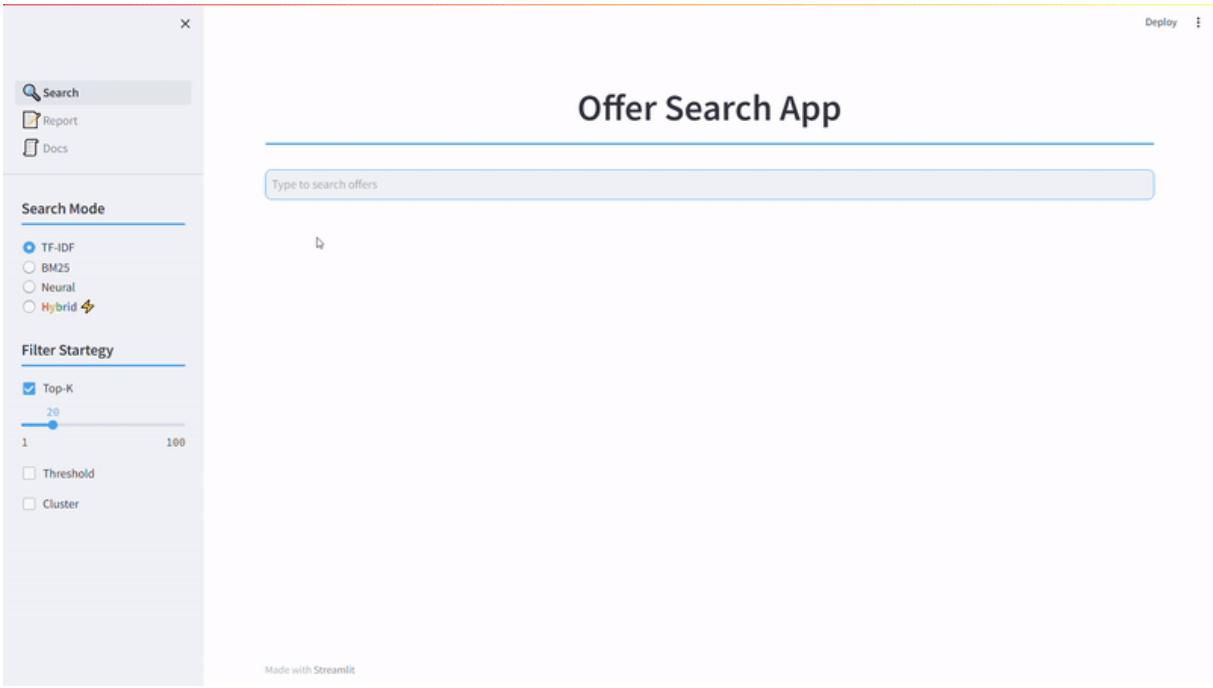
3. Download NLTK Data

```
python -m nltk.downloader punkt stopwords
```

Run App

```
cd search-app
conda activate search
streamlit run 🔍_Search.py
```

Now search for offers with options on the left to see results.



Report

Project report can be found here or you can find it in the app as well.

Project Structure

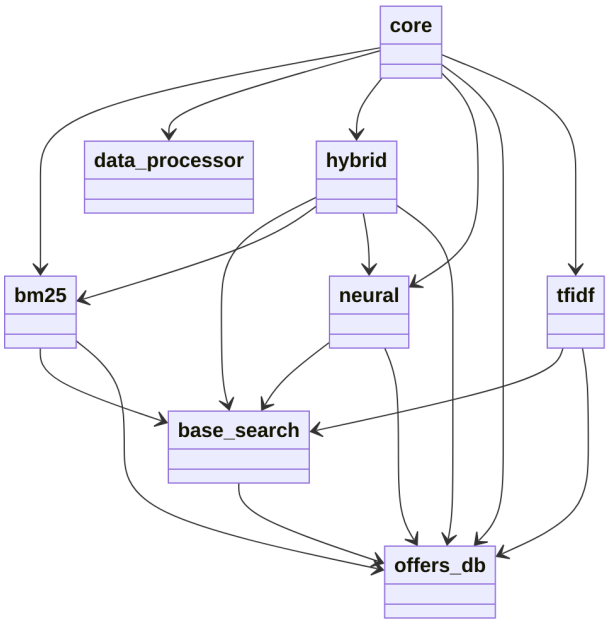
Folder Name	Description
.streamlit	Configuration files for Streamlit
.vscode	Visual Studio Code settings and files
core	Core application module
├─ base_search.py	Base search class
├─ bm25.py	BM25 search class
├─ data_processor.py	Data processing code
├─ hybrid.py	Hybrid search class
├─ **init**.py	Initialization module
├─ neural.py	Neural search class
├─ offers_db.py	Offers database class
├─ tfidf.py	TF-IDF search class
data	Data used by the application
├─ processed	Processed data files
│ ├─ database.sqlite	Offers SQLite database

Folder Name	Description
│ ├─ syn_queries.json	Synthetic queries
│ ├─ true_scores.csv	True scores (CSV)
│ ├─ true_scores_gold.csv	True scores (gold) (CSV)
│ └─ true_scores_syn.csv	True scores (synthetic) (CSV)
└─ raw	Raw data files
notebooks	Jupyter Notebook files
├─ eval.ipynb	Evaluation notebook
├─ queries.ipynb	Query Generation notebook
└─ search_exp.ipynb	Search experiment Notebook
vectors	Vector Database files
├─ bm25	BM25 model files
├─ neural	Neural model files
│ └─ retrieval	FAISS Vector Database Files
└─ tfidf	TF-IDF files
pages	Application web pages
🔍_Search.py	Streamlit App File
env.yml	Environment configuration file
README.md	Repository README file
requirements.txt	Python package requirements

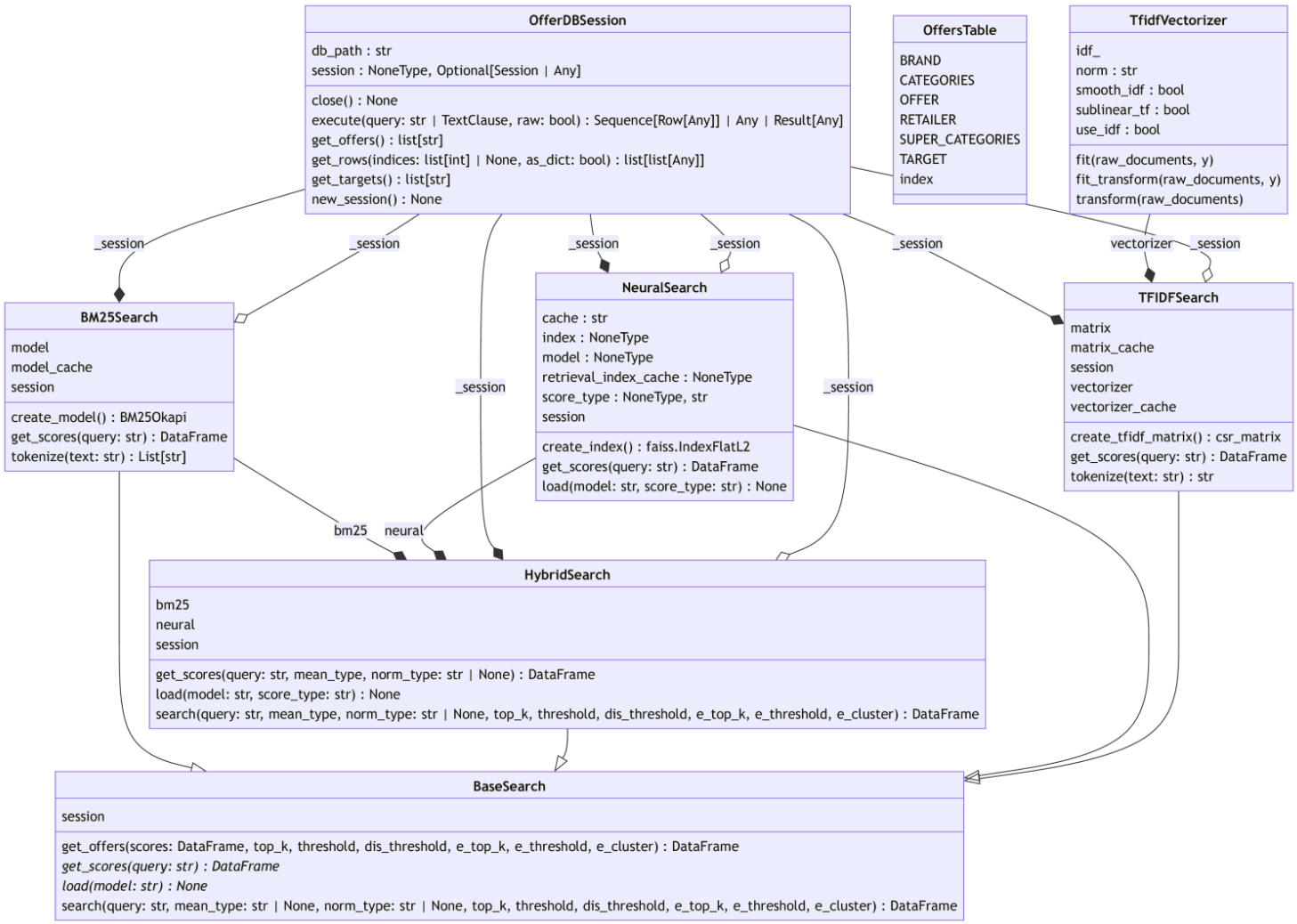
UML Diagrams

The pacakge `core` has the main code of this app. UML diagrams are shown below.

1. Core Package


core
data_processor
hybrid
bm25
neural
tfidf
base_search
offers_db

2. Classes


OfferDBSession
db_path : str
session : NoneType, Optional[Session | Any]
close() : None
execute(query: str | TextClause, raw: bool) : Sequence[Row[Any]] | Any | Result[Any]
get_offers() : list[str]
get_rows(indices: list[int] | None, as_dict: bool) : list[list[Any]]
get_targets() : list[str]
new_session() : None
OffersTable
BRAND
CATEGORIES
OFFER
RETAILER
SUPER_CATEGORIES
TARGET
index
TfidfVectorizer
idf_
norm : str
smooth_idf : bool
sublinear_tf : bool
use_idf : bool
fit(raw_documents, y)
fit_transform(raw_documents, y)
transform(raw_documents)
_session
vectorizer
BM25Search
model
model_cache
session
create_model() : BM25Okapi
get_scores(query: str) : DataFrame
tokenize(text: str) : List[str]
NeuralSearch
cache : str
index : NoneType
model : NoneType
retrieval_index_cache : NoneType
score_type : NoneType, str
session
create_index() : faiss.IndexFlatL2
get_scores(query: str) : DataFrame
load(model: str, score_type: str) : None
TFIDFSearch
matrix
matrix_cache
session
vectorizer
vectorizer_cache
create_tfidf_matrix() : csr_matrix
get_scores(query: str) : DataFrame
tokenize(text: str) : str
bm25
neural
HybridSearch
bm25
neural
session
get_scores(query: str, mean_type, norm_type: str | None) : DataFrame
load(model: str, score_type: str) : None
search(query: str, mean_type, norm_type: str | None, top_k, threshold, dis_threshold, e_top_k, e_threshold, e_cluster) : DataFrame
BaseSearch
session
get_offers(scores: DataFrame, top_k, threshold, dis_threshold, e_top_k, e_threshold, e_cluster) : DataFrame
get_scores(query: str) : DataFrame
load(model: str) : None
search(query: str, mean_type: str | None, norm_type: str | None, top_k, threshold, dis_threshold, e_top_k, e_threshold, e_cluster) : DataFrame

Contact

If you have any question, please email anant.singh@nyu.edu