

Building a Chess Training Assistant (GenAI RAG) using MongoDB Atlas Vector search, OpenAI, Sentence transformer and Streamlit

1. Introduction	3
1.1 Personalization and Adaptive Learning	3
1.2 Enhanced Game Analysis	3
1.3 Creative Play and Strategy Exploration	3
2. Preparing for Chess tournaments	3
3. Project NextMOVE: A GenAI Chess Training Assistant	5
4. Problem statement	5
5. What is vector embedding and similarity search?	5
5.1 Vector embedding	6
5.2 Similarity search	6
5.2.1 Cosine similarity	6
6. Why MongoDB and Atlas Vector Search?	6
6.1 Similarity Search	7
6.2 Integrated with MongoDB	7
6.3 Efficient Indexing	7
6.4 High Performance	7
6.5 Flexible Data Types	7
6.6 Security and Compliance	7
6.7 Ease of Use	8
6.8 Support for Modern Applications	8
6.9 Seamless Scaling	8
6.10 Comprehensive Monitoring	8
7. NextMOVE: Key components	9
7.1 Python	9
7.2 MongoDB Atlas and Atlas Vector Search	9
7.2.1 Sign up for a MongoDB account	10
7.2.2 Create a MongoDB cluster	10
7.2.3 Add a Database user	11
7.2.4 Configure a Network connection	12

7.2.5 Create database and collections	13
7.2.6 Get connection string	14
7.3 Sentence Transformer	14
7.4 OpenAI	14
7.4.1 OpenAI	15
7.4.2 Generate Open AI API secret key	15
8. Architecture	15
9. Approach	17
10. Tutorial for creating the NextMOVE system	18
10.1 pip install required packages	18
10.2 import statements	19
10.3 Helper functions	20
10.4 Verify connectivity to MongoDB	21
10.5 Verify OpenAI API secret key	22
10.5.1 Get OpenAI API secret key	22
10.5.2 Verify by sending a prompt and retrieving a response	22
10.6 Upload chess games	23
10.7 Get MongoDB database name, and the collection names for games and chat_history	24
10.8 Upload chess games and store along with their embeddings in the collection	24
10.8.1 Choose chess game to upload	24
10.8.2 Extract text from the chosen pdf file of chess game	25
10.8.3 Get the embedding of the chess game text	25
10.8.4 Insert game document (text + embedding)	27
10.9 Get user chess move query	28
10.10 Retrieve relevant documents based on cosine similarity search of the embeddings	28
10.11 Generate response to the user query	41
10.12 Save chat history	41
10.13 Verify the number of records inserted into the collections	41
11. Screenshots for using NextMOVE project via Streamlit application	42
11.1 Step 1	42
11.2 Step 2	44
11.4 Step 4	45
11.5 Step 5	47
11.6 Step 6	48
11.7 Step 7	48
11.8 Step 8	48
11.9 Step 9	49
12. Code base	49
13. Conclusion	49

1. Introduction

Chess training is undergoing a remarkable transformation with the advent of Generative AI (GenAI), revolutionizing how players learn, prepare, and improve their skills. Traditionally, chess training involved studying [openings](#), [middlegames](#), and [endgames](#), learning strategy and tactics, and analyzing past games. While these methods are still fundamental, GenAI has introduced a new dimension to chess training that enhances personalization, creativity, and efficiency.

1.1 Personalization and Adaptive Learning

GenAI allows for highly personalized training programs tailored to the individual strengths and weaknesses of each player. AI-driven tools can analyze a player's past games, identify patterns, and suggest targeted exercises to improve specific areas of their play. Whether it's refining opening strategies, improving mid-game tactics, or mastering endgames, GenAI ensures that training is optimized for each player's unique needs.

1.2 Enhanced Game Analysis

The ability of GenAI to analyze vast amounts of data has made game analysis more insightful and accessible. Players can now receive instant feedback on their moves, with AI not only suggesting the best moves, but also explaining the underlying strategies. This deep understanding helps players learn faster and more effectively, bridging the gap between amateur and expert analysis.

1.3 Creative Play and Strategy Exploration

One of the most exciting aspects of GenAI in chess training is its capacity for creative play. AI engines like [AlphaZero](#) have demonstrated unconventional and innovative strategies that challenge traditional chess theory. By training with GenAI, players are exposed to a broader range of ideas and possibilities, encouraging them to think outside the box and develop a more versatile style of play.

2. Preparing for Chess tournaments

In the high-stakes world of competitive chess, preparation is far more than a mere strategy—it's a meticulous science that separates champions from contenders. For top chess championships, grandmasters engage in exhaustive analysis of past games, scrutinizing every move of their opponents to gain a crucial edge. The advent of artificial intelligence has revolutionized this process, enabling the creation of personalized chess training assistants that tailor strategies based on opponents. In World Chess Championship matches, rigorous opponent-specific

preparation is paramount. Leading grandmasters delve deep into their opponents' styles and tendencies, utilizing advanced [chess engines](#), [chess base](#), and specialized [software](#) to refine their tactics and outmaneuver their rivals on the global stage.

This tutorial delves deep into how MongoDB's powerful technology, particularly MongoDB Atlas Search, can elevate the preparation process to new heights by assisting in recommending the next best moves for a given chess game position.

For illustrative purposes, let's consider a game position.



Image courtesy: lichess.org

The above example is a board position after 6 moves from the [Catalan Opening](#) Closed variation. This game just got underway and is in the [opening phase](#) of the game. The primary objective is to come up with recommendation for the best possible moves given any board position of a chess game.

The International Chess Federation [FIDE](#) maintains the chess game notation and analysis notes of all past games played in FIDE tournaments. They are available in the Chess base repository as well. The games are available as [Portable Game notation \(pgn\)](#) files, pdf files or in other formats. Each file can have one or more games. The games are available for downloading. These games are study material. Players have their own unique style and tendencies. Preparation involves understanding these nuances in detail.

3. Project NextMOVE: A GenAI Chess Training Assistant

World Chess championship preparation requires studying games from past decades, and analyzing several hundreds of opening lines and variations, middle game strategy and tactics, and end game technique and precision. Games are available as text data. Because GenAI has the potential to study a huge corpus of text data, this article explores building a Chess training assistant GenAI application.

NextMOVE is a project which came out of a GenAI hackathon. The project is an effort to build a personalized Chess Training Assistant. NextMOVE's objective is to assist chess players in preparing for chess matches and tournaments. This can be achieved by analyzing and leveraging the opponent's past games, which are uploaded to the assistant. Next moves for each board position can be recommended based on chess knowledge bases, philosophies of each chess opening, and factoring the style and tendencies of players.

4. Problem statement

The problem statement boils down to building a recommendation system by studying thousands of text files of past games, and coming up with the best possible next move recommendations for any given board position. Vectors are great candidates for representing game text data. Similarity searches performed in high-dimensional vector space would yield next move recommendations.

5. What is vector embedding and similarity search?

First, let's take a closer look into vector embedding and similarity search.

5.1 Vector embedding

A [vector embedding](#) is a numerical representation of data, typically in the form of a high-dimensional vector. This representation captures the semantic meaning or essential features of the data, allowing for more efficient processing and comparison by machines. They are used in a variety of machine learning and data analysis applications.

5.2 Similarity search

[Similarity search](#) is a process in which a system identifies and retrieves data points from a database that are similar to a given query. This concept is widely used in fields like information retrieval, machine learning, and data science to find items that are most relevant or similar to a user's input.

There are many approaches to perform similarity search. One of the approaches is cosine similarity.

5.2.1 Cosine similarity

[Cosine similarity](#) is a key metric that helps to understand the [semantic similarity](#) between diverse datasets and documents in a robust way. It is widely used in a variety of fields. Most common use cases of cosine similarity are in the areas of [natural language processing](#), search algorithms, and [recommendation systems](#).

Cosine similarity is a [measure of similarity](#) between two non-zero vectors defined in an [inner product space](#). Cosine similarity is the [cosine](#) of the angle between the vectors, which can be used to quantify how similar the vectors are. The cosine similarity will return a value between -1 and 1; a value closer to 1 indicates greater similarity.

6. Why MongoDB and Atlas Vector Search?

[MongoDB](#) is a popular open-source, NoSQL database designed for storing and managing large volumes of unstructured or semi-structured data. Unlike traditional relational databases, MongoDB uses a flexible, document-oriented model that allows data to be stored in JSON-like documents with dynamic schemas. This approach makes it well-suited for handling diverse data types and rapidly evolving application requirements.

[MongoDB Atlas Vector Search](#) is a feature that allows you to perform similarity searches on high-dimensional data, such as vectors representing text, images, or other complex data types.

This feature is particularly useful for applications involving machine learning, natural language processing, and recommendation systems.

Here are the salient features of MongoDB Atlas Vector Search:

6.1 Similarity Search

- **Vector Similarity:** Enables searching for items based on vector similarity rather than exact matches, which is crucial for applications like recommendation engines and semantic search.
- **[Cosine Similarity](#), [Euclidean Distance](#), and [Dot Product](#):** Supports various similarity metrics to find the most relevant results based on your use case.

6.2 Integrated with MongoDB

- **Native Integration:** Fully integrated into MongoDB Atlas, allowing you to store and search vector data alongside other types of data in the same database.
- **Aggregation Framework Support:** Integrates with MongoDB's aggregation pipeline, enabling complex queries that combine vector search with traditional data queries.

6.3 Efficient Indexing

- **HNSW Indexing:** Utilizes Hierarchical Navigable Small World (HNSW) graphs for efficient indexing and fast retrieval of similar vectors, even in large datasets.
- **Scalability:** Designed to scale efficiently, making it suitable for large-scale applications with millions of vectors.

6.4 High Performance

- **Low Latency:** Optimized for low-latency searches, ensuring fast response times for applications requiring real-time similarity search.
- **Parallel Processing:** Leverages MongoDB's distributed architecture to process queries in parallel, improving search performance.

6.5 Flexible Data Types

- **Support for High-Dimensional Data:** Handles vectors of varying dimensions, making it suitable for a wide range of applications, including image search, text embeddings, and more.
- **Mixed Workloads:** Allows mixing vector search with other types of queries (text, numeric, etc.) within the same database and query.

6.6 Security and Compliance

- **Secure Data Handling:** Inherits MongoDB Atlas's security features, including encryption at rest and in transit, ensuring that vector data is securely managed.
- **Compliance:** Compliant with various industry standards and regulations, providing peace of mind for sensitive applications.

6.7 Ease of Use

- **Simple API:** Provides an intuitive API for adding and querying vector data, making it accessible even to developers who are new to vector search.
- **Integration with Developer Tools:** Works seamlessly with MongoDB's ecosystem of tools, including Atlas Search, Charts, and the aggregation framework.

6.8 Support for Modern Applications

- **Machine Learning & AI:** Ideal for use cases involving machine learning models, such as natural language processing (NLP), image recognition, and recommendation systems.
- **Recommendation Engines:** Enables building recommendation systems that rely on similarity between user preferences or product features.

6.9 Seamless Scaling

- **Global Clusters:** Supports deployment across multiple regions, ensuring high availability and performance globally.
- **Auto-Scaling:** Automatically adjusts resources based on workload demands, ensuring consistent performance as your application grows.

6.10 Comprehensive Monitoring

- **Performance Insights:** Provides monitoring tools to track the performance of vector searches, helping you optimize your queries and indexes.
- **Integrated Logging:** Logs and analytics are integrated into MongoDB Atlas, providing visibility into how vector searches are performing in real time.

These features make MongoDB Atlas Vector Search a powerful tool for modern applications that require advanced similarity search capabilities, enabling developers to build intelligent, responsive, and scalable solutions and recommendation systems.

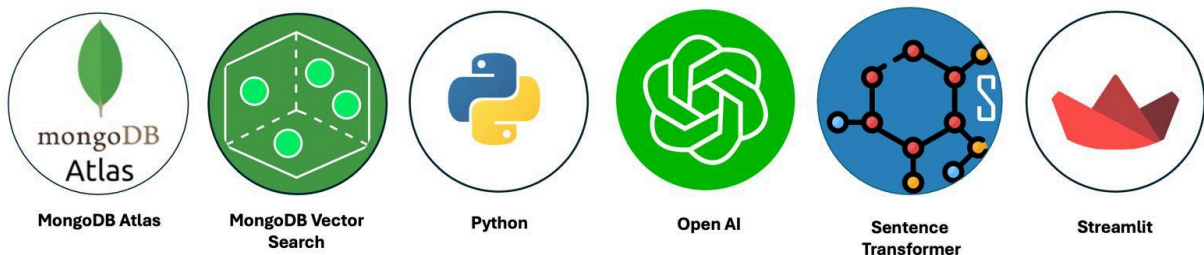
Since NextMOVE is fundamentally a recommendation system, and MongoDB and Atlas Vector Search perfectly meets the requirements for the problem statement, we went with MongoDB and Atlas Vector Search for building the NextMOVE recommendation system.

Now, lets look at the steps to build the NextMOVE - Chess training assistant system.

7. NextMOVE: Key components

The NextMOVE - Chess training assistant system is built in Python. The application uses MongoDB, OpenAI's GPT-3.5, and a sentence transformer model. The software can be run as a Streamlit web application.

Here are the key components:



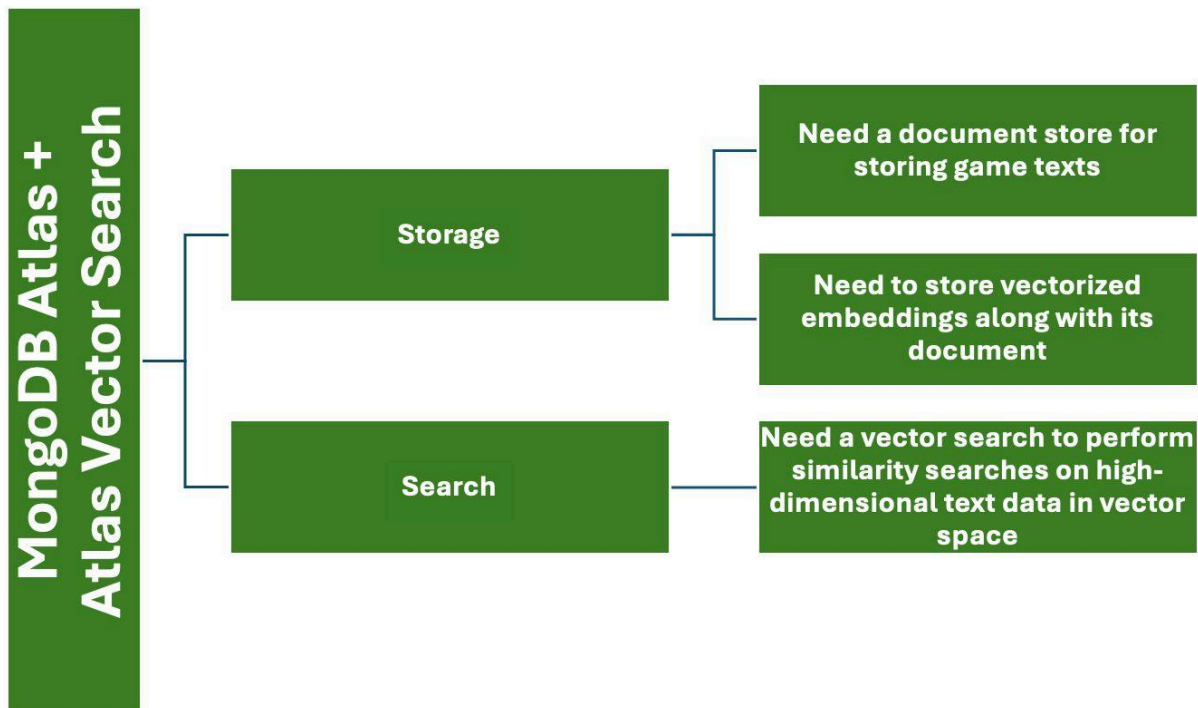
7.1 Python

We will use Python for developing the software. Let's choose Python version 3.x.

7.2 MongoDB Atlas and Atlas Vector Search

MongoDB serves as the database system for storing, retrieving, and managing data related to chess games and chat history. It plays a central role in persisting the data that the application works with.

The following are the requirements:



The MongoDB [Start with Guides](#) documentation has all the step-by-step instructions for the required tasks.

7.2.1 Sign up for a MongoDB account

If you do not have a MongoDB account, create one by following the instructions given [here](#).

7.2.2 Create a MongoDB cluster

Create a cluster by following the instructions given [here](#).

For this tutorial, let's choose the **Free tier M0** cluster of MongoDB Atlas which comes with **512 MB storage, shared RAM and shared vCPU**. At the time of writing, the default MongoDB version supported for the Free tier M0 is 7.x.

The step by step documentation is given [here](#).



Free

CURRENT

For learning and exploring MongoDB in a cloud environment.

STORAGE

512 MB

RAM

Shared

vCPU

Shared

Features


✓ Free forever

7.2.3 Add a Database user

Create a database user by following the instructions given [here](#).

Set up the username and password and role as given in the screenshot.

Add New Database User

Create a database user to grant an application or user access to databases and collections in your clusters in this Atlas project. Granular access control can be configured with default privileges or custom roles. You can grant access to an Atlas project or organization using the corresponding [Access Manager](#) 

Authentication Method

Password

Certificate

AWS IAM
(MongoDB 4.4 and up)


Federated Auth
(MongoDB 7.0 and up) 


MongoDB uses [SCRAM](#) as its default authentication method.

Password Authentication

e.g. new-user_31

Enter password SHOW

 Autogenerate Secure Password

 Copy

Database User Privileges

Configure role based access control by assigning database user a mix of one built-in role, multiple custom roles, and multiple specific privileges. A user will gain access to all actions within the roles assigned to them, not just the actions those roles share in common. **You must choose at least one role or privilege.** [Learn more about roles.](#)

Built-in Role

Select one [built-in role](#) for this user.

0 SELECTED



Select Role

Atlas admin

Read and write to any database

Only read any database

or create a custom role in the [Custom Roles](#)  tab.

Specific Privileges

Select multiple privileges and what database and collection they are associated with.
Leaving collection blank will grant this role for all collections in the database.

Choose the **Atlas admin** role to the user. The Atlas admin role would be needed to create collections, indexes, etc..

7.2.4 Configure a Network connection

We need to configure a network connection by following the instructions given [here](#) and whitelist the IP address from where the mongodb client would connect to the MongoDB cluster.

The IP address to be whitelisted could be either the IP address of the end-user application or it could be a wildcard with 0.0.0.0. Please note that the 0.0.0.0 wildcard would allow connections from anywhere into the mongodb database.

Add IP Access List Entry ✕

Atlas only allows client connections to a cluster from entries in the project's IP Access List. Each entry should either be a single IP address or a CIDR-notated range of addresses. [Learn more](#).

ADD CURRENT IP ADDRESS

Access List Entry:

Enter IP Address or CIDR Notation

Comment:

Optional comment describing this entry



This entry is temporary and will be deleted in

6 hours ▼

Cancel

Confirm

7.2.5 Create database and collections

Under the mongodb cluster, let us create a database and two collectons as given below:

Object type	Object name	Description
Database	nextmove_db	Please create the NextMOVE database by following the instructions given here .
Collection	games	Please create this collection under the nextmove_db database for storing the games of interest along with its embeddings. The documentation is given here .
Collection	chat_history	Please create this collection under the nextmove_db database for storing the chat_history of the user's queries and system responses

Note: The database name **nextmove_db**, and the collection names **games** and **chat_history** would be needed later for programmatic access.

7.2.6 Get connection string

For programmatic access, the connection string to connect to the MongoDB cluster is needed. The instructions to get the connection string are given [here](#).

The connection uri is a string value in the following format:

```
"mongodb+srv://<DB_USER_NAME>:<DB_USER_PASSWORD>@<MONGODB_CLUSTER_HOSTNAME>/?retryWrites=true&w=majority&appName=<MONGODB_CLUSTER_NAME>"
```

It is made up of the following:

- **DB_USER_NAME**
- **DB_USER_PASSWORD**
- **MONGODB_CLUSTER_HOSTNAME**
- **MONGODB_CLUSTER_NAME**

Note: The **mongodb connection uri** string would be needed later for programmatic access.

7.3 Sentence Transformer

The Sentence Transformer model plays a critical role in converting text data (specifically, the content of chess game PDFs and user queries) into vector representations. These vector representations are then used for tasks like similarity comparison and document retrieval. The pre-trained [Sentence Transformer model \(paraphrase-MiniLM-L6-v2\)](#) is used to map sentences and paragraphs to a 384-dimensional dense vector space and can be used for semantic search.

7.4 OpenAI

The Chess training assistant software uses OpenAI's GPT-3.5 model to generate natural language responses to user queries. By processing the user's question along with related game

data, GPT-3.5 provides intelligent, context-aware advice, enhancing the overall functionality and interactivity of the software.

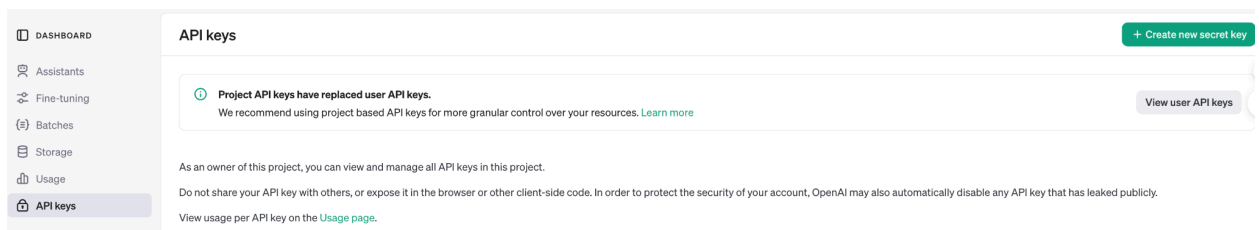
7.4.1 OpenAI

[Create an OpenAI account](#) if there isn't one or use an existing account.

7.4.2 Generate Open AI API secret key

Integration with GPT-3.5 is through API. So, generate an Open AI API key by following the steps [here](#).

Here is the OpenAI API key screenshot for creating a new secret key.

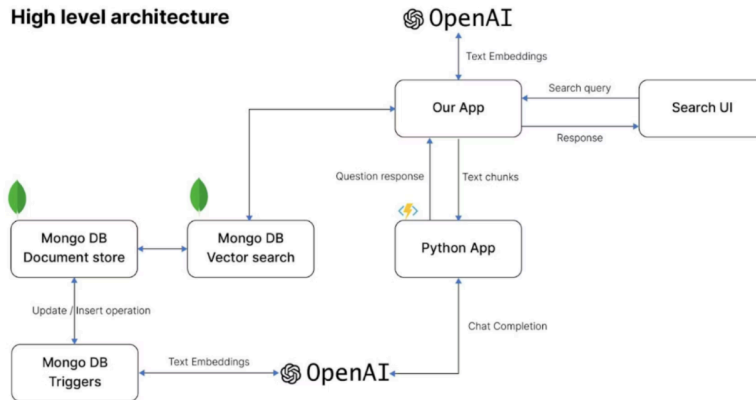


8. Architecture

Here is the high level architecture diagram:

ARCHITECTURE

High level architecture

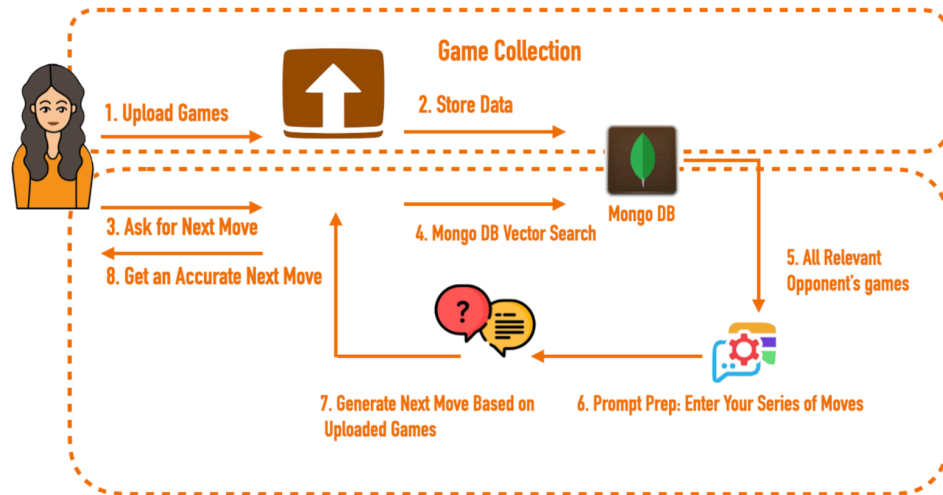


Tech stack:

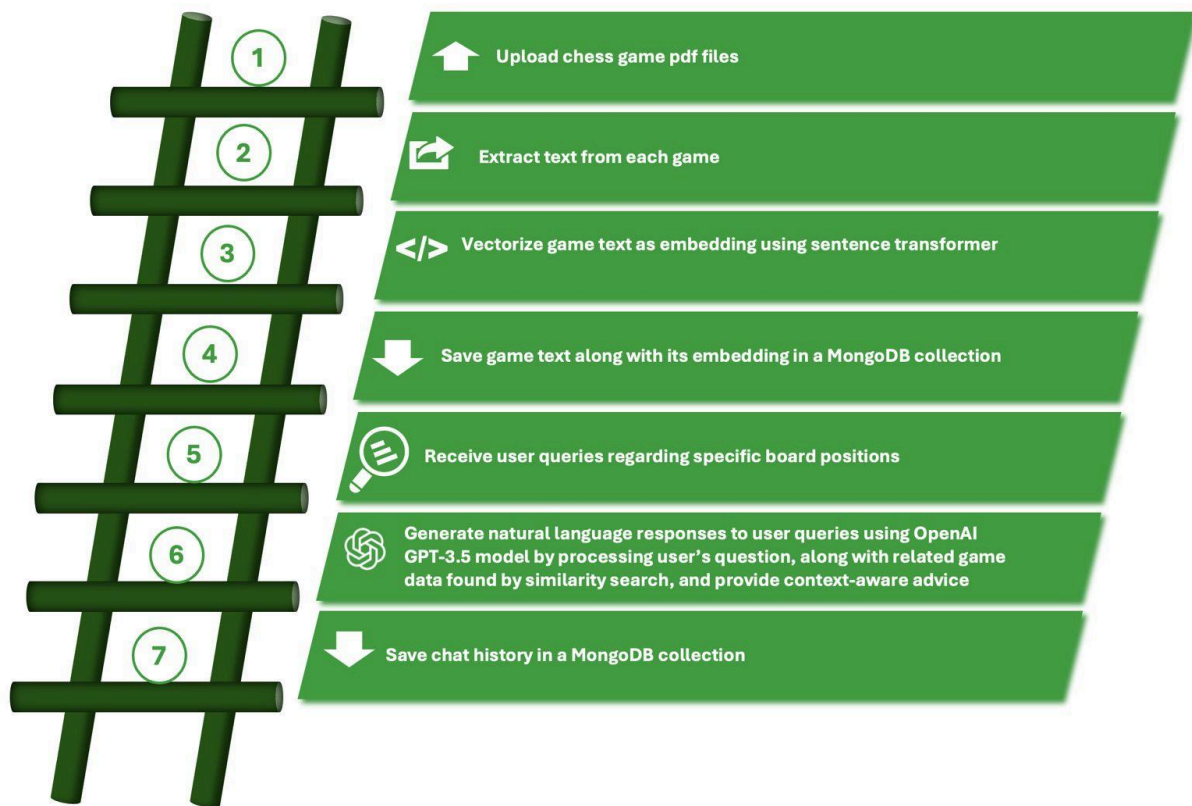
- Python
- MongoDB Atlas Vector Search
- Open AI
- LLM
- Streamlit



FLOW



9. Approach



10. Tutorial for creating the NextMOVE system

10.1 pip install required packages

Install the following required packages:

Package name	Description
pymongo	PyMongo is a Python distribution containing tools for working with MongoDB, and is the recommended way to work with MongoDB from Python
pymupdf	PyMuPDF is a high-performance Python library for data extraction, analysis, conversion & manipulation of PDF (and other) documents.
PyPDF2	A pure-python PDF library capable of splitting, merging, cropping, and

	transforming PDF files.
sentence-transformers	SentenceTransformers is a Python framework for state-of-the-art sentence, text and image embeddings.
openai	The OpenAI Python library provides convenient access to the OpenAI API from applications written in the Python language. It includes a pre-defined set of classes for API resources that initialize themselves dynamically from API responses which makes it compatible with a wide range of versions of the OpenAI API.
streamlit	Streamlit lets you transform Python scripts into interactive web apps. This is optional if web apps are not needed.

`pip3 install -U pymongo pymupdf PyPDF2 sentence-transformers openai==0.28 streamlit`

10.2 import statements

The following are the required import statements from the installed packages:

Import statement	Description
os	This module provides a portable way of using operating system dependent functionality. The software needs this for setting up environment variables.
getpass	Prompts the user for inputting the password without echoing
openai	The OpenAI Python library provides convenient access to the OpenAI API from applications written in Python
pymongo.mongo_client , pymongo.server_api , pymongo.database , pymongo.collection	Tools for connecting to MongoDB objects
tempfile	This module creates temporary files. This is used while extracting text from pdf files
pymupdf , PyPDF2	High performance Python libraries for data extraction, analysis, conversion & manipulation of PDF documents

sentence_transformers	Sentence Transformers (a.k.a. SBERT) is the go-to Python module for accessing, using, and training state-of-the-art text and image embedding models. It can be used to compute embeddings using Sentence Transformer models (quickstart) or to calculate similarity scores using Cross-Encoder models (quickstart).
numpy	An array object represents a multidimensional, homogeneous array of fixed-size items. This is used during vectorizing the text
streamlit , streamlit.runtime.uploaded_file_manager	Streamlit module makes it easy for visualizing, mutating, and sharing data.

```
import os
import getpass
import openai
```

```
from pymongo.mongo_client import MongoClient
from pymongo.server_api import ServerApi
from pymongo.database import Database
from pymongo.collection import Collection
import tempfile
import pymupdf
from sentence_transformers import SentenceTransformer
from streamlit.runtime.uploaded_file_manager import UploadedFile
from numpy import ndarray
import PyPDF2
import streamlit
```

10.3 Helper functions

The following are the helper functions used by the software:

Function name	Description
init_config_parameters()	This function verifies that all necessary configuration parameters (like MongoDB credentials and the OpenAI API key) are set.
get_mongodb_cluster_connection_uri()	Constructs MongoDB connection URI from the username, password, and cluster hostname

<code>get_mongodb_cluster_client()</code>	Creates a connection to the MongoDB cluster using the URI and returns the MongoClient object
<code>get_mongodb_database()</code>	Connects to the specified MongoDB database and returns a Database object
<code>get_collection()</code>	Retrieves a specific collection from MongoDB database
<code>extract_text_from_pdf()</code>	Extracts and returns text from a given PDF file
<code>get_sentence_transformer_model()</code>	Loads and returns pre-trained paraphrase-MiniLM-L6-v2 sentence transformer model
<code>vectorize_text()</code>	Converts text into a vector using sentence transformer model and returns it as a NumPy array
<code>save_embedding_to_collection()</code>	Saves vectorized text (embedding) along with the original text to the specified MongoDB collection
<code>retrieve_relevant_docs()</code>	Retrieves and returns documents from MongoDB that are most relevant to a user query by calculating cosine similarity between query vector and stored vectors
<code>cosine_similarity()</code>	Calculates and returns cosine similarity between two vectors
<code>generate_response()</code>	Generates a response using OpenAI's GPT-3.5 by augmenting the user's query with relevant documents
<code>save_chat_history()</code>	Saves the user's query and the generated response to the MongoDB chat history collection
<code>get_text_from_pdf()</code>	Extracts text from a PDF file uploaded via Streamlit

10.4 Verify connectivity to MongoDB

Please verify connectivity to MongoDB cluster by using the `mongodb_cluster_uri` value from section [7.2.6 Get connection string](#).

try:

```
# Connect to MongoDB
client = MongoClient(mongodb_cluster_uri, server_api=ServerApi('1'))
print(" client: ", client)

# Send a ping to confirm a successful connection
client.admin.command('ping')
```

```
print("Pinged your deployment. You successfully connected to MongoDB!")
except Exception as e:
    print(e)
```

10.5 Verify OpenAI API secret key

10.5.1 Get OpenAI API secret key

Please verify the validity of the OpenAI API secret key by supplying the OPENAI_API_KEY from section [7.4.2 Generate Open AI API secret key](#) and sending a prompt and retrieving a response.

```
os.environ["OPENAI_API_KEY"] = getpass.getpass("Enter your OpenAI API Key: ")
openai.api_key = os.getenv("OPENAI_API_KEY")
assert (openai.api_key is not None), "OpenAI API key not found."
assert ((openai.api_key).startswith("sk-")), "OpenAI API key not supplied."
```

10.5.2 Verify by sending a prompt and retrieving a response

```
# Test the OpenAI key by sending a prompt and retrieving a response which is correct
response = openai.ChatCompletion.create(
    model="gpt-3.5-turbo",
    messages=[
        {
            "role": "user",
            "content": "What is Chess?"
        }
    ]
)
print(response.choices[0]["message"]["content"])
```

Here is a sample OpenAI response:

Chess is a two-player strategy board game played on an 8x8 square board. The objective of the game is to checkmate the opponent's king, where the king is threatened with capture and there is no way to remove the threat. The game is played by moving pieces such as pawns, rooks, knights, bishops, queens, and kings in different ways around the board, with each piece having

its own unique movement rules. Chess is a game of skill, strategy, and foresight, and is considered one of the most popular and enduring board games in the world.

This verifies that the supplied OPENAI_API_KEY is valid.

10.6 Upload chess games

1. Create a folder **chess_games**
2. Under the **chess_games** folder, upload the chess games of the top World Chess Grand Masters of interest. Currently, games are in pdf format and each pdf file has one game.

Here is a screenshot of the **chess_games** folder along with a bunch of uploaded sample chess game pdf files:



10.7 Get MongoDB database name, and the collection names for games and chat_history

Please supply the names of the database and collection names here as set up in section [7.2.5 Create database and collections](#).

```
mongodb_database_name = input('Enter the MongoDB database name: ')
assert mongodb_database_name is not None, "mongodb_database_name not found."
assert len(mongodb_database_name) > 0, "mongodb_database_name not found."
```

```
mongodb_database_games_collection_name = input('Enter the MongoDB database games
collection name: ')
assert mongodb_database_games_collection_name is not None,
"mongodb_database_games_collection_name not found."
assert len(mongodb_database_games_collection_name) > 0,
"mongodb_database_games_collection_name not found."
```

```
mongodb_database_chat_history_collection_name = input('Enter the MongoDB database chat
history collection name: ')
assert mongodb_database_chat_history_collection_name is not None,
"mongodb_database_chat_history_collection_name not found."
assert len(mongodb_database_chat_history_collection_name) > 0,
"mongodb_database_chat_history_collection_name not found."
```

10.8 Upload chess games and store along with their embeddings in the collection

The upload chess games and store them along with their embeddings in the **games** collection. Upload games one by one at this time.

10.8.1 Choose chess game to upload

Here is a sample chess game file for upload:


```
chess_game_path = '/content/chess_games/2022_07_29_FabianoCaruana_vs_LirenDing.pdf'
```

10.8.2 Extract text from the chosen pdf file of chess game

Using the **get_text_from_pdf()** helper function, extract text from the game pdf file.

```
if chess_game_path:
    game_text = get_text_from_pdf_file(chess_game_path)

    assert game_text is not None, "game_text not set."
    print("\n Extracted game_text: \n", game_text)
```

Output:

Extracted game_text:

```
[Event "44th Olympiad"][Site "Chennai IND"][Date "2022.08.06"][EventDate
"2022.07.29"][Round "8.2"][Result "0-1"][White "Fabiano Caruana"][Black "D Gukesh"][ECO
"B31"][WhiteElo "2783"][BlackElo "2684"][PlyCount "90"]1. e4 c5 2. Nf3 Nc6 3. Bb5 g6 4. O-O
Bg7 5. Bxc6 bxc6 6. Re1 Qc7 7. h3 d6 8. e5 dxe5 9. d3 c4 10. Nc3 cxd3 11. cxd3 Nh6 12. Nxe5
Nf5 13. Bf4 Qb7 14. Na4 f6 15. Nf3 O-O 16. d4 g5 17. Bh2 h5 18. Re4 Qd7 19. Qc2 Rf7 20.
Rae1 Bf8 21. Qe2 Qd5 22. Nc3 Qd7 23. Qc4 Qb7 24. b4 e6 25. Rb1 Qd7 26. Rbe1 Qb7 27.
Rb1 Qd7 28. a3 a5 29. Na4 Qd8 30. bxa5 Rxa5 31. Nc5 Qd5 32. Qe2 Rxa3 33. Rd1 Rfa7 34.
g4 hxg4 35. hxg4 Nh6 36. Bg3 e5 37. Nxe5 fxe5 38. Rxe5 Bxg4 39. Qd2 Qf3 40. Rxg5+ Rg7 41.
Re1 Bh3 42. Bd6 Bxd6 43. Rxg7+ Kxg7 44. Qg5+ Kh7 45. Ne4 Qxe4 0-1
```

10.8.3 Get the embedding of the chess game text

Using the **vectorize_text(text)** helper function, get the embedding using the **paraphrase-MiniLM-L6-v2** sentence transformer model

```
if game_text:
    # Vectorize text
    game_embedding = vectorize_text(game_text)
    assert game_embedding is not None, "game_embedding not set."
    print("\n Vectorized game_embedding: \n", game_embedding)
```

Vectorized game_embedding:

[0.2567692 0.3684017 -0.06755236 -0.13758254 -0.0200319 -0.09046234
0.09211368 -0.02257897 -0.18462443 -0.1594061 0.05559651 -0.49572527
0.12740204 0.16313681 -0.08947769 -0.0302004 -0.00577444 -0.36823016
-0.0826666 -0.11545002 -0.14053847 -0.13760409 -0.2133454 0.335554
0.05343148 0.07255784 -0.12466898 0.14485192 -0.2191895 0.20909098
0.14475402 0.33113736 0.20040914 0.60128385 -0.03053452 0.01252158
-0.41501397 0.4271993 0.14966115 0.03023961 0.3914386 -0.14401028
0.21564054 0.16804025 0.32681295 0.20339845 0.09364618 0.13810675
-0.34627023 0.04835742 -0.01199693 -0.10801706 -0.2586795 -0.3303762
-0.07739836 -0.20683141 0.10660323 -0.21686874 0.16713475 -0.12191925
-0.21565747 -0.19280463 -0.55348116 0.1250669 0.0224112 -0.26241055
0.26136693 0.04500067 0.320032 0.09029071 0.36296198 0.10028452
-0.08646931 -0.09876262 0.02086782 0.41583735 0.00248126 0.16969077
-0.12222876 -0.40576893 0.06872654 -0.14645195 0.17233285 0.0048948
0.3383916 -0.04284348 0.15841305 0.187723 0.29364663 -0.18179902
0.3758855 0.41307053 0.15360288 0.01841654 -0.2806887 0.29619974
0.25629655 -0.29467845 0.19654995 0.22924986 -0.05913305 -0.00931274
-0.3014965 0.05252716 -0.16880654 -0.233789 -0.16099925 -0.29959413
-0.03845017 0.5204412 -0.17979476 0.02447064 -0.112783 -0.07317369
-0.18061477 0.2593152 0.13253227 0.22779231 -0.43408605 0.13811877
-0.23796107 0.0703135 0.16072589 -0.0579242 -0.06561121 0.27652788
0.20673525 0.12084162 -0.14777578 -0.49225572 -0.12409134 -0.16050455
-0.03943775 -0.1963321 -0.12324284 -0.29399624 0.241011 -0.2661574
-0.13057184 -0.14968352 0.28212318 -0.3699065 0.2960595 -0.06082282
0.21616033 0.07950626 0.18559718 0.00682699 0.21739662 -0.30613133
-0.05907704 0.0631406 -0.08095888 0.6477256 0.22445402 -0.12321607
0.1005993 0.17842795 0.22355023 0.01102505 0.00617341 0.05820905
0.0504476 0.10405947 -0.02871764 -0.0196567 0.13581201 -0.3376009
0.05738129 0.19785348 -0.35927734 0.12254949 0.01459521 0.14646912
-0.16533035 -0.32953244 0.39375025 0.24473982 -0.22963493 0.0651249
-0.19954887 -0.21642205 0.14495146 -0.08352657 -0.1265716 -0.1273988
-0.11784583 0.18337697 0.19053698 0.01005965 -0.43495047 0.01690659
-0.17846082 -0.00563451 -0.04858462 -0.04497155 0.14596488 -0.3697239
0.22267324 0.07848533 -0.01848284 0.00574766 -0.08415469 -0.1809941
0.48901877 -0.14525807 0.02026547 -0.33196557 0.03964838 0.26616257
-0.0545083 -0.15387765 0.27029908 -0.29293308 0.28233358 0.2382546
0.01658374 -0.16925426 -0.5057198 -0.15132958 -0.02140377 -0.21081556
-0.04773902 -0.3041022 -0.12070415 0.12021817 -0.16486664 -0.17143854
0.40122497 -0.23395628 -0.10593325 0.14170767 0.36996898 -0.12982711
0.05053724 -0.10593335 0.07973413 0.05694453 -0.07974184 0.23737665
0.08229482 0.11486363 -0.2102621 0.16293208 -0.05240969 0.106218

-0.430364	-0.10953853	-0.36351347	0.12919407	0.2741824	-0.22440284
-0.01271678	0.18371674	-0.15450779	-0.20596366	-0.22599246	0.24088135
0.02910654	0.05520485	0.3484499	0.06350661	-0.18966419	-0.03407986
0.15075323	0.03893606	-0.20263813	0.34546697	0.05961641	-0.03397607
-0.02748372	-0.01148197	-0.08075281	-0.23410393	-0.18219283	-0.16473728
0.02613735	0.2700936	0.24663495	0.0817993	0.20001177	-0.2770778
-0.26788598	-0.14671764	-0.23019218	-0.35900807	0.02207949	-0.07481954
-0.16293755	0.02469859	0.28983483	0.52811486	-0.14910991	-0.28454703
0.14217149	0.25141153	-0.3738244	-0.05891595	0.08894309	-0.18299912
0.06351259	0.61717606	0.15187621	0.09787294	-0.2216517	0.11385196
-0.00078264	0.25524455	0.01284791	-0.01758968	0.08683662	-0.23410252
0.3360705	0.3839215	-0.06258121	0.2013149	0.11170942	0.15246911
0.14266774	0.05322169	0.23750311	-0.06989338	-0.00097872	0.27246836
0.0743616	0.31378832	-0.45626524	-0.35826463	0.3506911	0.05788286
0.02145157	0.07856058	0.07023494	-0.10176709	-0.2575848	-0.27836835
0.02904245	0.26204076	-0.21164681	-0.246203	-0.31066257	0.07806059
0.5764674	-0.21239875	-0.17663357	-0.07297447	-0.39991575	0.04234906
-0.08066636	0.02626494	-0.25789344	0.03186365	0.13546461	-0.4327457
-0.08893563	-0.00591825	-0.2798763	-0.04670785	0.4087094	0.05005709
0.07134064	-0.04500493	0.13647747	0.5021792	0.47571903	-0.09938538
-0.27453274	-0.13317062	-0.01288525	-0.329114	-0.3840867	0.08607898
0.23172358	-0.3453854	0.17557018	0.11270425	0.11785696	0.02073364
0.07264437	0.0060477	0.08456499	-0.47762164	-0.3691914	0.12638813

10.8.4 Insert game document (text + embedding)

The following code snippet is used to store the game text along with its embedding in the MongoDB collection.

```
games_collection = nextmove_db[mongodb_database_games_collection_name]
chat_history_collection = nextmove_db[mongodb_database_chat_history_collection_name]

# Insert game vector along with its embedding into MongoDB
save_embedding_to_collection(game_embedding, game_text, games_collection)
```

Note: If the database and collections are not created before, MongoDB creates databases and collections in a **'lazy'** fashion, when a document is inserted into a collection.

Now upload all games into the MongoDB collection.

10.9 Get user chess move query

Now, stored game is ready for further analysis. Users can send in queries for evaluating board positions, and for suggestions for best possible move sequences.

```
user_query = input('Enter your query: ')
assert user_query is not None, "user_query not found."
print(" user_query: ", user_query)
```

Enter your query: What could the reigning World Chess Champion Liren Ding play after the moves 1. d4 Nf6 2. c4 e6 3. Nf3 d5 4. g3 Be7 5. Bg2 O-O 6. O-O c6 ?

user_query: What could the reigning World Chess Champion Liren Ding play after the moves 1. d4 Nf6 2. c4 e6 3. Nf3 d5 4. g3 Be7 5. Bg2 O-O 6. O-O c6 ?

10.10 Retrieve relevant documents based on cosine similarity search of the embeddings

Now, the user query needs to be served. We need to retrieve relevant documents based on the user queries. The [cosine similarity](#) is used to search for similar vectors.

```
relevant_docs = retrieve_relevant_docs(user_query, games_collection)
print(" relevant_docs: ", relevant_docs)
```

```
relevant_docs: [{'_id': ObjectId('66aed90c70937d04cd52c0c4'), 'text': '[Event "12th Norway Chess 2024"]\n[Site "Stavanger NOR"]\n[Date "2024.05.29"]\n[Round "3.2"]\n[White "Caruana, Fabiano"]\n[Black "Ding, Liren"]\n[Result "1-0"]\n[WhiteElo "2805"]\n[BlackElo "2762"]\n[ECO "C50"]\n1. e4 e5 2. Nf3 Nc6 3. Bc4 Nf6 4. d3 Bc5 5. O-O O-O 6. Nbd2 d6 7. c3 \na5 8. h3 h6 9. Re1 Be6 10. Bb5 Qb8 11. Re2 Qa7 12. Bxc6 bxc6 13. a4 \nNd7 14. Nf1 d5 15. d4 exd4 16. Nxd4 Qb6 17. Be3 Bxd4 18. Bxd4 Qb7 19. \nexd5 cxd5 20. Ng3 Rae8 21. Qd2 c5 22. Be3 Kh7 23. Rae1 Qc6 24. Bf4 \nQxa4 25. Nf5 Qc6 26. Nxd7 Kxd7 27. Bxh6+ Kh7 28. Bxf8 Rxf8 29. Qg5 \nRe8 30. c4 Rb8 31. Rxe6 1-0\n', 'vector': [0.027073096483945847, 0.44056519865989685, -0.08691849559545517, -0.28480657935142517, -0.09544820338487625, 0.11994956433773041, -0.184430792927742, -0.024239234626293182, 0.12317286431789398, -0.12151265144348145, 0.05672769248485565, -0.2802247107028961, 0.1506723314523697, -0.23731565475463867, 0.022006865590810776, 0.11264384537935257, -0.3668924570083618, -0.16388411819934845, -0.21983051300048828, 0.05691417679190636, -0.2022455334663391,
```

-0.17728756368160248, -0.28528252243995667, 0.11766955256462097,
0.0396481528878212, 0.07566004991531372, -0.21221822500228882,
0.17749711871147156, -0.1464407742023468, 0.109429731965065, 0.03294447064399719,
0.25134459137916565, 0.1036597266793251, 0.3871048092842102, 0.059703316539525986,
-0.12132531404495239, -0.3942906856536865, 0.04787498712539673,
-0.03741531819105148, 0.2292879819869995, 0.23295892775058746,
0.022589700296521187, -0.15756653249263763, -0.0625820904970169,
0.19871969521045685, 0.42329126596450806, 0.1225365698337555, 0.14001047611236572,
-0.36512595415115356, 0.3329830765724182, -0.10336294770240784,
-0.05474776774644852, -0.23896081745624542, -0.23465421795845032,
-0.018513834103941917, -0.19255362451076508, -0.19811056554317474,
-0.04849938303232193, 0.08700096607208252, -0.23720969259738922,
0.1170225739479065, -0.2231387048959732, -0.48399609327316284, 0.1987454891204834,
0.08300220966339111, 0.10060993582010269, 0.02173895388841629,
-0.14848202466964722, 0.06641849130392075, 0.2535170912742615,
0.38354548811912537, -0.030875669792294502, -0.21904565393924713,
-0.27847468852996826, 0.06511358171701431, 0.5050056576728821,
-0.1535647213459015, -0.07591573148965836, -0.08512567728757858,
-0.21147838234901428, -0.08187003433704376, -0.07755057513713837,
0.0396653488278389, -0.16114851832389832, 0.37936654686927795,
-0.16368791460990906, 0.30254489183425903, 0.21119645237922668,
0.5708807706832886, -0.32763931155204773, 0.20485083758831024, 0.2078612595796585,
0.1374625414609909, 0.1268608272075653, -0.10013647377490997, 0.407863050699234,
0.42706507444381714, -0.0867689773440361, 0.0967334657907486, 0.1939121037721634,
0.08897697925567627, 0.385518342256546, -0.21013839542865753, -0.0674416646361351,
0.11376294493675232, 0.015485167503356934, -0.0011874549090862274,
-0.2537999153137207, 0.0456358976662159, 0.4104626774787903, 0.023916523903608322,
-0.12898704409599304, 0.08578234910964966, -0.11511985957622528,
-0.31667080521583557, 0.22355413436889648, 0.20110070705413818,
0.08243800699710846, -0.4178588092327118, 0.1827288269996643, 0.13434675335884094,
-0.04024096950888634, 0.07661694288253784, 0.15364578366279602,
-0.10352850705385208, 0.15546588599681854, 0.10815325379371643,
0.22084443271160126, -0.27540522813796997, -0.47254568338394165,
0.025165045633912086, -0.2456035315990448, 0.28032732009887695,
0.11997025460004807, -0.2814770042896271, -0.3142278790473938, 0.3496909439563751,
-0.12941543757915497, -0.21066772937774658, 0.002081606537103653,
-0.1596221625804901, -0.1868278831243515, 0.21669861674308777,
-0.05247632414102554, 0.4494839608669281, -0.015223084017634392,
0.1770208179950714, 0.011878923512995243, 0.3570932447910309,
-0.43460506200790405, -0.08903411775827408, 0.050158288329839706,
-0.2780316174030304, 0.5991400480270386, 0.18845577538013458,
-0.17779704928398132, 0.14449182152748108, 0.04171563312411308,
0.09585457295179367, -0.06285607814788818, -0.02127634361386299,
-0.026220068335533142, 0.16567561030387878, 0.404679536819458,
0.03856422007083893, 0.14956039190292358, -0.01133008673787117,
-0.3648379445075989, 0.06118407100439072, 0.06156390905380249,

-0.26599764823913574, 0.07203595340251923, -0.0015428541228175163,
-0.013542018830776215, -0.16279351711273193, -0.3367401957511902,
0.12434142082929611, 0.057582493871450424, -0.2000414878129959,
0.2101554572582245, -0.11733616143465042, 0.261706680059433, 0.4308749735355377,
-0.3808358311653137, -0.3168443441390991, 0.10911983996629715,
-0.40364980697631836, 0.3571797013282776, 0.036342449486255646,
0.17231863737106323, 0.09210126101970673, 0.1311301589012146,
-0.17713376879692078, 0.0985269844532013, -0.519740641117096, -0.06999856978654861,
0.17566020786762238, -0.3110314607620239, 0.3735275864601135,
0.027564633637666702, 0.11098684370517731, -0.08397150039672852,
-0.02965320460498333, -0.535146951675415, 0.2452421337366104, -0.14439715445041656,
0.1429620236158371, -0.39828646183013916, 0.011417297646403313,
0.09255632013082504, -0.09609518945217133, -0.29277685284614563,
-0.08633622527122498, -0.3063664734363556, 0.2713797688484192, 0.3212961256504059,
0.1373305320739746, -0.12000986188650131, -0.6183642148971558,
-0.04078860953450203, -0.03827144205570221, -0.30258867144584656,
-0.12628376483917236, -0.27168962359428406, -0.26161062717437744,
-0.15527528524398804, -0.15141747891902924, -0.267210990190506,
0.10969327390193939, -0.1609935164451599, -0.08396643400192261,
-0.14379143714904785, 0.27953916788101196, 0.10925883054733276,
0.2313588559627533, -0.2410900741815567, 0.13896329700946808, 0.15356029570102692,
0.22973386943340302, 0.024247216060757637, 0.2363290935754776,
0.10898976773023605, 0.03230762854218483, -0.022604499012231827,
-0.06187841668725014, -0.12979862093925476, -0.7016453742980957,
0.12645554542541504, -0.11008548736572266, 0.18861663341522217,
0.1551690697669983, -0.13860411942005157, -0.06024371460080147,
0.11237838864326477, -0.003658311441540718, -0.02564200758934021,
0.07579085230827332, 0.11932715028524399, 0.03461054712533951,
0.049415361136198044, 0.4848979413509369, 0.22972843050956726, -0.158794566988945,
-0.2782105505466461, 0.4713420569896698, -0.09082083404064178, -0.242597758769989,
0.2974507808685303, -0.12206916511058807, -0.0092929657548666,
0.06506107747554779, -0.19129058718681335, 0.29021361470222473,
-0.3405936062335968, 0.0017972616478800774, -0.08865484595298767,
-0.2364894598722458, 0.114812932908535, -0.14984023571014404, 0.10874905437231064,
-0.11666102707386017, -0.3358709216117859, -0.22068065404891968,
-0.03421276807785034, -0.23300468921661377, -0.2122921645641327,
-0.1780969798564911, 0.1660243570804596, 0.0735713541507721, 0.029855381697416306,
0.09126738458871841, 0.4201326370239258, -0.03529251739382744,
-0.20303210616111755, -0.06473685055971146, 0.30210036039352417,
-0.28129225969314575, 0.26150280237197876, 0.2451748549938202,
-0.23657146096229553, 0.09123792499303818, 0.5168223977088928, 0.1701768934726715,
-0.2560133635997772, -0.19905713200569153, -0.1475244015455246,
0.06617462635040283, 0.29447275400161743, -0.10854395478963852,
0.18468685448169708, 0.16780626773834229, -0.2026384323835373,
0.04717939719557762, 0.500515878200531, 0.16306215524673462, 0.12633004784584045,
-0.2716456651687622, 0.3035275936126709, 0.3226501941680908, 0.2487526535987854,

0.13047708570957184, -0.2950231432914734, 0.0076784249395132065,
0.3875802159309387, 0.19160857796669006, 0.03944425657391548,
-0.15976496040821075, -0.5472540855407715, -0.05327674746513367,
-0.009879928082227707, -0.05387115478515625, 0.1352187693119049,
0.2028554081916809, -0.09425413608551025, 0.1291055828332901,
0.023456763476133347, 0.08165695518255234, 0.24127642810344696,
-0.1821156144142151, 0.03531723842024803, -0.16781803965568542,
-0.055523186922073364, 0.7213495969772339, -0.21129260957241058,
-0.3121160864830017, -0.1712653636932373, -0.12705251574516296,
-0.07547234743833542, -0.007882552221417427, 0.16082853078842163,
-0.003818003460764885, 0.15790486335754395, 0.3323564827442169,
-0.33748501539230347, 0.052357278764247894, -0.14928558468818665,
-0.28944846987724304, 0.19759774208068848, 0.1033133938908577,
-0.034712307155132294, -0.1667354851961136, 0.15853387117385864,
-0.07972780615091324, 0.426746666431427, 0.427879273891449, -0.18021941184997559,
-0.26343756914138794, -0.23434898257255554, 0.15368570387363434,
-0.027111437171697617, -0.5365806818008423, 0.24384239315986633,
0.42695483565330505, -0.39816659688949585, 0.06640453636646271,
0.015237946063280106, 0.11398448050022125, -0.170368954539299,
-0.24272263050079346, -0.08385323733091354, -0.024701697751879692,
-0.5110821723937988, -0.21909019351005554, 0.1538912057876587]], {'_id':
**ObjectId('66af3d85015bdc60fc773e3'), 'text': '[Event "44th Olympiad"]\n[Site "Chennai
IND"]\n[Date "2022.08.06"]\n[EventDate "2022.07.29"]\n[Round "8.2"]\n[Result
"0-1"]\n[White "Fabiano Caruana"]\n[Black "D Gukesh"]\n[ECO "B31"]\n[WhiteElo
"2783"]\n[BlackElo "2684"]\n[PlyCount "90"]\n1. e4 c5 2. Nf3 Nc6 3. Bb5 g6 4. O-O Bg7 5.
Bxc6 bxc6 6. Re1 Qc7 7. h3 \n d6 8. e5 \n dxe5 9. d3 c4 10. Nc3 cxd3 11. cxd3 Nh6 12. Nxe5
Nf5 13. Bf4 Qb7 14. \n Na4 f6 15. \n Nf3 O-O 16. d4 g5 17. Bh2 h5 18. Re4 Qd7 19. Qc2 Rf7
20. Rae1 Bf8 21. \n Qe2 Qd5 \n 22. Nc3 Qd7 23. Qc4 Qb7 24. b4 e6 25. Rb1 Qd7 26. Rbe1
Qb7 27. Rb1 \n Qd7 28. a3 a5 \n 29. Na4 Qd8 30. bxa5 Rxa5 31. Nc5 Qd5 32. Qe2 Rxa3 33.
Rd1 Rfa7 34. \n g4 hxg4 35. \n hxg4 Nh6 36. Bg3 e5 37. Nxe5 fxe5 38. Rxe5 Bxg4 39. Qd2
Qf3 40. Rxg5+ \n Rg7 41. \n Re1 Bh3 42. Bd6 Bxd6 43. Rxg7+ Kxg7 44. Qg5+ Kh7 45. Ne4
Qxe4 0-1 \n', 'vector': [0.25676921010017395, 0.3684017062187195, -0.0675523579120636,
-0.13758254051208496, -0.020031895488500595, -0.0904623419046402,
0.0921136811375618, -0.022578971460461617, -0.18462443351745605,
-0.15940609574317932, 0.05559650808572769, -0.49572527408599854,
0.12740203738212585, 0.1631368100643158, -0.08947768807411194,
-0.030200395733118057, -0.005774440243840218, -0.3682301640510559,
-0.08266659826040268, -0.11545002460479736, -0.14053846895694733,
-0.13760408759117126, -0.21334539353847504, 0.33555400371551514,
0.05343148112297058, 0.07255784422159195, -0.12466897815465927,
0.1448519229888916, -0.21918949484825134, 0.2090909779071808, 0.14475402235984802,
0.33113735914230347, 0.2004091441631317, 0.601283848285675, -0.030534517019987106,
0.01252157986164093, -0.41501396894454956, 0.4271993041038513,
0.14966115355491638, 0.030239608138799667, 0.3914386034011841,
-0.1440102756023407, 0.21564054489135742, 0.16804024577140808, 0.3268129527568817,
0.2033984512090683, 0.09364618360996246, 0.13810674846172333,**

-0.34627023339271545, 0.04835741966962814, -0.011996932327747345,
-0.10801706463098526, -0.25867950916290283, -0.33037620782852173,
-0.07739835977554321, -0.2068314105272293, 0.1066032275557518,
-0.21686874330043793, 0.16713474690914154, -0.12191925197839737,
-0.21565747261047363, -0.19280463457107544, -0.5534811615943909,
0.12506690621376038, 0.022411201149225235, -0.26241055130958557,
0.26136693358421326, 0.04500066861510277, 0.320032000541687, 0.09029071033000946,
0.3629619777202606, 0.10028451681137085, -0.08646930754184723,
-0.0987626239657402, 0.02086782082915306, 0.4158373475074768,
0.0024812649935483932, 0.16969077289104462, -0.12222876399755478,
-0.4057689309120178, 0.0687265396118164, -0.1464519500732422, 0.17233285307884216,
0.004894801881164312, 0.33839160203933716, -0.04284348338842392,
0.15841305255889893, 0.18772299587726593, 0.2936466336250305, -0.1817990243434906,
0.3758854866027832, 0.41307052969932556, 0.15360288321971893, 0.01841653883457184,
-0.28068870306015015, 0.2961997389793396, 0.2562965452671051, -0.2946784496307373,
0.19654995203018188, 0.22924986481666565, -0.059133049100637436,
-0.009312735870480537, -0.3014965057373047, 0.052527155727148056,
-0.1688065379858017, -0.23378899693489075, -0.16099925339221954,
-0.29959413409233093, -0.038450174033641815, 0.5204411745071411,
-0.17979475855827332, 0.02447064220905304, -0.11278299987316132,
-0.07317369431257248, -0.18061476945877075, 0.2593151926994324,
0.1325322687625885, 0.22779230773448944, -0.43408605456352234,
0.13811877369880676, -0.2379610687494278, 0.07031349837779999, 0.1607258915901184,
-0.05792419612407684, -0.06561121344566345, 0.27652788162231445,
0.2067352533340454, 0.1208416223526001, -0.1477757841348648, -0.49225571751594543,
-0.12409134209156036, -0.160504549741745, -0.03943774849176407,
-0.19633209705352783, -0.12324284017086029, -0.29399624466896057,
0.24101099371910095, -0.2661573886871338, -0.13057184219360352,
-0.14968352019786835, 0.2821231782436371, -0.3699065148830414, 0.2960594892501831,
-0.06082282215356827, 0.21616032719612122, 0.07950626313686371,
0.18559718132019043, 0.006826990284025669, 0.21739661693572998,
-0.3061313331127167, -0.05907703563570976, 0.06314060091972351,
-0.08095888048410416, 0.6477255821228027, 0.22445401549339294,
-0.12321607023477554, 0.10059930384159088, 0.17842794954776764,
0.2235502302646637, 0.011025051586329937, 0.006173405796289444,
0.058209046721458435, 0.05044759809970856, 0.10405946522951126,
-0.028717638924717903, -0.019656695425510406, 0.13581201434135437,
-0.3376008868217468, 0.057381290942430496, 0.1978534758090973,
-0.3592773377895355, 0.12254948914051056, 0.014595209620893002,
0.1464691162109375, -0.16533035039901733, -0.3295324444770813,
0.39375025033950806, 0.2447398155927658, -0.22963492572307587,
0.06512489914894104, -0.1995488703250885, -0.21642205119132996,
0.1449514627456665, -0.08352657407522202, -0.12657159566879272,
-0.1273988038301468, -0.11784583330154419, 0.1833769679069519, 0.1905369758605957,
0.010059649124741554, -0.43495047092437744, 0.01690659299492836,
-0.17846082150936127, -0.005634509027004242, -0.04858462139964104,

-0.04497154802083969, 0.1459648758172989, -0.3697238862514496, 0.222673237323761,
0.07848533242940903, -0.018482841551303864, 0.005747657269239426,
-0.08415468782186508, -0.18099409341812134, 0.4890187680721283,
-0.1452580690383911, 0.02026546746492386, -0.3319655656814575,
0.039648376405239105, 0.26616257429122925, -0.05450829863548279,
-0.1538776457309723, 0.2702990770339966, -0.29293307662010193, 0.2823335826396942,
0.2382546067237854, 0.01658373698592186, -0.16925425827503204, -0.505719780921936,
-0.15132957696914673, -0.02140376903116703, -0.21081556379795074,
-0.047739021480083466, -0.30410221219062805, -0.12070415169000626,
0.1202181726694107, -0.164866641163826, -0.1714385449886322, 0.4012249708175659,
-0.23395627737045288, -0.10593324899673462, 0.1417076736688614, 0.369968980550766,
-0.1298271119594574, 0.050537243485450745, -0.10593335330486298,
0.07973413169384003, 0.05694453418254852, -0.07974183559417725,
0.2373766452074051, 0.08229482173919678, 0.11486362665891647,
-0.21026210486888885, 0.16293208301067352, -0.05240969359874725,
0.10621799528598785, -0.4303640127182007, -0.10953852534294128,
-0.3635134696960449, 0.12919406592845917, 0.27418240904808044,
-0.22440284490585327, -0.012716777622699738, 0.18371674418449402,
-0.15450778603553772, -0.20596365630626678, -0.22599245607852936,
0.24088135361671448, 0.029106542468070984, 0.0552048534154892, 0.3484498858451843,
0.0635066106915474, -0.18966418504714966, -0.03407985717058182,
0.1507532298564911, 0.038936056196689606, -0.20263813436031342,
0.3454669713973999, 0.059616412967443466, -0.03397607430815697,
-0.02748372033238411, -0.011481966823339462, -0.08075281232595444,
-0.23410393297672272, -0.1821928322315216, -0.1647372841835022,
0.026137351989746094, 0.2700935900211334, 0.24663494527339935,
0.08179929852485657, 0.20001177489757538, -0.2770777940750122,
-0.2678859829902649, -0.1467176377773285, -0.2301921844482422, -0.3590080738067627,
0.02207949385046959, -0.07481954246759415, -0.16293755173683167,
0.024698585271835327, 0.2898348271846771, 0.5281148552894592,
-0.14910991489887238, -0.28454703092575073, 0.1421714872121811, 0.2514115273952484,
-0.3738243877887726, -0.05891595035791397, 0.08894309401512146,
-0.1829991191625595, 0.06351259350776672, 0.6171760559082031, 0.15187621116638184,
0.09787294268608093, -0.22165170311927795, 0.11385196447372437,
-0.0007826359942555428, 0.25524455308914185, 0.012847906909883022,
-0.01758967712521553, 0.08683662116527557, -0.2341025173664093,
0.3360705077648163, 0.3839215040206909, -0.06258121132850647, 0.20131489634513855,
0.11170941591262817, 0.15246911346912384, 0.1426677405834198, 0.0532216876745224,
0.23750311136245728, -0.06989337503910065, -0.0009787213057279587,
0.27246835827827454, 0.07436159998178482, 0.3137883245944977,
-0.45626524090766907, -0.35826462507247925, 0.3506911098957062,
0.05788286030292511, 0.021451570093631744, 0.07856057584285736,
0.07023493945598602, -0.10176709294319153, -0.257584810256958,
-0.27836835384368896, 0.029042446985840797, 0.26204076409339905,
-0.21164681017398834, -0.2462030053138733, -0.31066256761550903,
0.0780605897307396, 0.5764673948287964, -0.21239875257015228, -0.1766335666179657,

-0.07297447323799133, -0.39991575479507446, 0.04234905540943146,
-0.08066636323928833, 0.026264943182468414, -0.257893443107605,
0.03186364844441414, 0.135464608669281, -0.43274569511413574,
-0.08893562853336334, -0.005918249487876892, -0.2798762917518616,
-0.046707846224308014, 0.40870940685272217, 0.05005709081888199,
0.0713406354188919, -0.04500493407249451, 0.13647747039794922, 0.5021792054176331,
0.47571903467178345, -0.09938538074493408, -0.2745327353477478,
-0.13317061960697174, -0.01288524828851223, -0.32911399006843567,
-0.3840866982936859, 0.08607897907495499, 0.2317235767841339, -0.3453854024410248,
0.17557017505168915, 0.11270424723625183, 0.117856964468956, 0.02073364332318306,
0.07264436781406403, 0.006047704257071018, 0.08456499129533768,
-0.4776216447353363, -0.36919140815734863, 0.12638813257217407}}, {'_id':
ObjectId('66baa33b0fac82afdaf31455'), 'text': '[Event "44th Olympiad"]'[Site "Chennai
IND"]'[Date "2022.08.06"]'[EventDate "2022.07.29"]'[Round "8.2"]'[Result "0-1"]'[White
"Fabiano Caruana"]'[Black "D Gukesh"]'[ECO "B31"]'[WhiteElo "2783"]'[BlackElo
"2684"]'[PlyCount "90"]1. e4 c5 2. Nf3 Nc6 3. Bb5 g6 4. O-O Bg7 5. Bxc6 bxc6 6. Re1 Qc7
7. h3 d6 8. e5 dxe5 9. d3 c4 10. Nc3 cxd3 11. cxd3 Nh6 12. Nxe5 Nf5 13. Bf4 Qb7 14. Na4 f6
15. Nf3 O-O 16. d4 g5 17. Bh2 h5 18. Re4 Qd7 19. Qc2 Rf7 20. Rae1 Bf8 21. Qe2 Qd5 22.
Nc3 Qd7 23. Qc4 Qb7 24. b4 e6 25. Rb1 Qd7 26. Rbe1 Qb7 27. Rb1 Qd7 28. a3 a5 29. Na4
Qd8 30. bxa5 Rxa5 31. Nc5 Qd5 32. Qe2 Rxa3 33. Rd1 Rfa7 34. g4 hxg4 35. hxg4 Nh6 36.
Bg3 e5 37. Nxe5 fxe5 38. Rxe5 Bxg4 39. Qd2 Qf3 40. Rxc5+ Rg7 41. Re1 Bh3 42. Bd6 Bxd6
43. Rxc5+ Kxc5 44. Qg5+ Kh7 45. Ne4 Qxe4 0-1', 'vector': [0.25676921010017395,
0.3684017062187195, -0.0675523579120636, -0.13758254051208496,
-0.020031895488500595, -0.0904623419046402, 0.0921136811375618,
-0.022578971460461617, -0.18462443351745605, -0.15940609574317932,
0.05559650808572769, -0.49572527408599854, 0.12740203738212585,
0.1631368100643158, -0.08947768807411194, -0.030200395733118057,
-0.005774440243840218, -0.3682301640510559, -0.08266659826040268,
-0.11545002460479736, -0.14053846895694733, -0.13760408759117126,
-0.21334539353847504, 0.33555400371551514, 0.05343148112297058,
0.07255784422159195, -0.12466897815465927, 0.1448519229888916,
-0.21918949484825134, 0.2090909779071808, 0.14475402235984802,
0.33113735914230347, 0.2004091441631317, 0.601283848285675, -0.030534517019987106,
0.01252157986164093, -0.41501396894454956, 0.4271993041038513,
0.14966115355491638, 0.030239608138799667, 0.3914386034011841,
-0.1440102756023407, 0.21564054489135742, 0.16804024577140808, 0.3268129527568817,
0.2033984512090683, 0.09364618360996246, 0.13810674846172333,
-0.34627023339271545, 0.04835741966962814, -0.011996932327747345,
-0.10801706463098526, -0.25867950916290283, -0.33037620782852173,
-0.07739835977554321, -0.2068314105272293, 0.1066032275557518,
-0.21686874330043793, 0.16713474690914154, -0.12191925197839737,
-0.21565747261047363, -0.19280463457107544, -0.5534811615943909,
0.12506690621376038, 0.022411201149225235, -0.26241055130958557,
0.26136693358421326, 0.04500066861510277, 0.320032000541687, 0.09029071033000946,
0.3629619777202606, 0.10028451681137085, -0.08646930754184723,
-0.0987626239657402, 0.02086782082915306, 0.4158373475074768,

0.0024812649935483932, 0.16969077289104462, -0.12222876399755478,
-0.4057689309120178, 0.0687265396118164, -0.1464519500732422, 0.17233285307884216,
0.004894801881164312, 0.33839160203933716, -0.04284348338842392,
0.15841305255889893, 0.18772299587726593, 0.2936466336250305, -0.1817990243434906,
0.3758854866027832, 0.41307052969932556, 0.15360288321971893, 0.01841653883457184,
-0.28068870306015015, 0.2961997389793396, 0.2562965452671051, -0.2946784496307373,
0.19654995203018188, 0.22924986481666565, -0.059133049100637436,
-0.009312735870480537, -0.3014965057373047, 0.052527155727148056,
-0.1688065379858017, -0.23378899693489075, -0.16099925339221954,
-0.29959413409233093, -0.038450174033641815, 0.5204411745071411,
-0.17979475855827332, 0.02447064220905304, -0.11278299987316132,
-0.07317369431257248, -0.18061476945877075, 0.2593151926994324,
0.1325322687625885, 0.22779230773448944, -0.43408605456352234,
0.13811877369880676, -0.2379610687494278, 0.07031349837779999, 0.1607258915901184,
-0.05792419612407684, -0.06561121344566345, 0.27652788162231445,
0.2067352533340454, 0.1208416223526001, -0.1477757841348648, -0.49225571751594543,
-0.12409134209156036, -0.160504549741745, -0.03943774849176407,
-0.19633209705352783, -0.12324284017086029, -0.29399624466896057,
0.24101099371910095, -0.2661573886871338, -0.13057184219360352,
-0.14968352019786835, 0.2821231782436371, -0.3699065148830414, 0.2960594892501831,
-0.06082282215356827, 0.21616032719612122, 0.07950626313686371,
0.18559718132019043, 0.006826990284025669, 0.21739661693572998,
-0.3061313331127167, -0.05907703563570976, 0.06314060091972351,
-0.08095888048410416, 0.6477255821228027, 0.22445401549339294,
-0.12321607023477554, 0.10059930384159088, 0.17842794954776764,
0.2235502302646637, 0.011025051586329937, 0.006173405796289444,
0.058209046721458435, 0.05044759809970856, 0.10405946522951126,
-0.028717638924717903, -0.019656695425510406, 0.13581201434135437,
-0.3376008868217468, 0.057381290942430496, 0.1978534758090973,
-0.3592773377895355, 0.12254948914051056, 0.014595209620893002,
0.1464691162109375, -0.16533035039901733, -0.3295324444770813,
0.39375025033950806, 0.2447398155927658, -0.22963492572307587,
0.06512489914894104, -0.1995488703250885, -0.21642205119132996,
0.1449514627456665, -0.08352657407522202, -0.12657159566879272,
-0.1273988038301468, -0.11784583330154419, 0.1833769679069519, 0.1905369758605957,
0.010059649124741554, -0.43495047092437744, 0.01690659299492836,
-0.17846082150936127, -0.005634509027004242, -0.04858462139964104,
-0.04497154802083969, 0.1459648758172989, -0.3697238862514496, 0.222673237323761,
0.07848533242940903, -0.018482841551303864, 0.005747657269239426,
-0.08415468782186508, -0.18099409341812134, 0.4890187680721283,
-0.1452580690383911, 0.02026546746492386, -0.3319655656814575,
0.039648376405239105, 0.26616257429122925, -0.05450829863548279,
-0.1538776457309723, 0.2702990770339966, -0.29293307662010193, 0.2823335826396942,
0.2382546067237854, 0.01658373698592186, -0.16925425827503204, -0.505719780921936,
-0.15132957696914673, -0.02140376903116703, -0.21081556379795074,
-0.047739021480083466, -0.30410221219062805, -0.12070415169000626,

0.1202181726694107, -0.164866641163826, -0.1714385449886322, 0.4012249708175659,
-0.23395627737045288, -0.10593324899673462, 0.1417076736688614, 0.369968980550766,
-0.1298271119594574, 0.050537243485450745, -0.10593335330486298,
0.07973413169384003, 0.05694453418254852, -0.07974183559417725,
0.2373766452074051, 0.08229482173919678, 0.11486362665891647,
-0.21026210486888885, 0.16293208301067352, -0.05240969359874725,
0.10621799528598785, -0.4303640127182007, -0.10953852534294128,
-0.3635134696960449, 0.12919406592845917, 0.27418240904808044,
-0.22440284490585327, -0.012716777622699738, 0.18371674418449402,
-0.15450778603553772, -0.20596365630626678, -0.22599245607852936,
0.24088135361671448, 0.029106542468070984, 0.0552048534154892, 0.3484498858451843,
0.0635066106915474, -0.18966418504714966, -0.03407985717058182,
0.1507532298564911, 0.038936056196689606, -0.20263813436031342,
0.3454669713973999, 0.059616412967443466, -0.03397607430815697,
-0.02748372033238411, -0.011481966823339462, -0.08075281232595444,
-0.23410393297672272, -0.1821928322315216, -0.1647372841835022,
0.026137351989746094, 0.2700935900211334, 0.24663494527339935,
0.08179929852485657, 0.20001177489757538, -0.2770777940750122,
-0.2678859829902649, -0.1467176377773285, -0.2301921844482422, -0.3590080738067627,
0.02207949385046959, -0.07481954246759415, -0.16293755173683167,
0.024698585271835327, 0.2898348271846771, 0.5281148552894592,
-0.14910991489887238, -0.28454703092575073, 0.1421714872121811, 0.2514115273952484,
-0.3738243877887726, -0.05891595035791397, 0.08894309401512146,
-0.1829991191625595, 0.06351259350776672, 0.6171760559082031, 0.15187621116638184,
0.09787294268608093, -0.22165170311927795, 0.11385196447372437,
-0.0007826359942555428, 0.25524455308914185, 0.012847906909883022,
-0.01758967712521553, 0.08683662116527557, -0.2341025173664093,
0.3360705077648163, 0.3839215040206909, -0.06258121132850647, 0.20131489634513855,
0.11170941591262817, 0.15246911346912384, 0.1426677405834198, 0.0532216876745224,
0.23750311136245728, -0.06989337503910065, -0.0009787213057279587,
0.27246835827827454, 0.07436159998178482, 0.3137883245944977,
-0.45626524090766907, -0.35826462507247925, 0.3506911098957062,
0.05788286030292511, 0.021451570093631744, 0.07856057584285736,
0.07023493945598602, -0.10176709294319153, -0.257584810256958,
-0.27836835384368896, 0.029042446985840797, 0.26204076409339905,
-0.21164681017398834, -0.2462030053138733, -0.31066256761550903,
0.0780605897307396, 0.5764673948287964, -0.21239875257015228, -0.1766335666179657,
-0.07297447323799133, -0.39991575479507446, 0.04234905540943146,
-0.08066636323928833, 0.026264943182468414, -0.257893443107605,
0.03186364844441414, 0.135464608669281, -0.43274569511413574,
-0.08893562853336334, -0.005918249487876892, -0.2798762917518616,
-0.046707846224308014, 0.40870940685272217, 0.05005709081888199,
0.0713406354188919, -0.04500493407249451, 0.13647747039794922, 0.5021792054176331,
0.47571903467178345, -0.09938538074493408, -0.2745327353477478,
-0.13317061960697174, -0.01288524828851223, -0.32911399006843567,
-0.3840866982936859, 0.08607897907495499, 0.2317235767841339, -0.3453854024410248,

0.17557017505168915, 0.11270424723625183, 0.117856964468956, 0.02073364332318306,
0.07264436781406403, 0.006047704257071018, 0.08456499129533768,
-0.4776216447353363, -0.36919140815734863, 0.12638813257217407]], {'_id':
Objectid('66aecbce7b2ef667496feea'), 'text': '[Event "44th Olympiad"]\n[Site "Chennai
IND"]\n[Date "2022.08.06"]\n[EventDate "2022.07.29"]\n[Round "8.2"]\n[Result
"0-1"]\n[White "Fabiano Caruana"]\n[Black "D Gukesh"]\n[ECO "B31"]\n[WhiteElo
"2783"]\n[BlackElo "2684"]\n[PlyCount "90"]\n1. e4 c5 2. Nf3 Nc6 3. Bb5 g6 4. O-O Bg7 5.
Bxc6 bxc6 6. Re1 Qc7 7. h3 \n d6 8. e5 \n dxe5 9. d3 c4 10. Nc3 cxd3 11. cxd3 Nh6 12. Nxe5
Nf5 13. Bf4 Qb7 14. \n Na4 f6 15. \n Nf3 O-O 16. d4 g5 17. Bh2 h5 18. Re4 Qd7 19. Qc2 Rf7
20. Rae1 Bf8 21. \n Qe2 Qd5 \n 22. Nc3 Qd7 23. Qc4 Qb7 24. b4 e6 25. Rb1 Qd7 26. Rbe1
Qb7 27. Rb1 \n Qd7 28. a3 a5 \n 29. Na4 Qd8 30. bxa5 Rxa5 31. Nc5 Qd5 32. Qe2 Rxa3 33.
Rd1 Rfa7 34. \n g4 hxg4 35. \n hxg4 Nh6 36. Bg3 e5 37. Nxe5 fxe5 38. Rxe5 Bxg4 39. Qd2
Qf3 40. Rxg5+ \n Rg7 41. \n Re1 Bh3 42. Bd6 Bxd6 43. Rxg7+ Kxg7 44. Qg5+ Kh7 45. Ne4
Qxe4 0-1 \n', 'vector': [0.25676941871643066, 0.36840152740478516,
-0.06755205988883972, -0.13758254051208496, -0.02003183402121067,
-0.09046251326799393, 0.09211383759975433, -0.022578919306397438,
-0.18462447822093964, -0.15940634906291962, 0.055596478283405304,
-0.4957253634929657, 0.12740199267864227, 0.16313669085502625,
-0.0894775241613388, -0.030200373381376266, -0.005774196237325668,
-0.3682301640510559, -0.08266651630401611, -0.11545021831989288,
-0.14053860306739807, -0.13760419189929962, -0.21334531903266907,
0.33555400371551514, 0.05343184620141983, 0.07255764305591583,
-0.12466898560523987, 0.14485210180282593, -0.21918949484825134,
0.20909158885478973, 0.14475420117378235, 0.33113712072372437, 0.2004089504480362,
0.6012837886810303, -0.030534591525793076, 0.012521801516413689,
-0.4150140881538391, 0.42719927430152893, 0.14966106414794922,
0.030239522457122803, 0.3914388120174408, -0.14401018619537354,
0.21564055979251862, 0.1680404394865036, 0.32681289315223694, 0.2033984363079071,
0.09364601969718933, 0.13810648024082184, -0.3462704122066498,
0.04835730791091919, -0.011996924877166748, -0.10801713168621063,
-0.2586795389652252, -0.33037593960762024, -0.07739819586277008,
-0.2068314105272293, 0.10660317540168762, -0.21686899662017822,
0.1671348214149475, -0.12191908061504364, -0.2156577706336975,
-0.19280490279197693, -0.5534809827804565, 0.1250668168067932,
0.02241101674735546, -0.262410432100296, 0.2613668143749237, 0.04500100016593933,
0.3200322091579437, 0.0902908518910408, 0.3629620373249054, 0.10028429329395294,
-0.08646906167268753, -0.09876222908496857, 0.02086758054792881,
0.41583701968193054, 0.0024811429902911186, 0.16969065368175507,
-0.12222875654697418, -0.40576910972595215, 0.06872650980949402,
-0.14645186066627502, 0.1723330020904541, 0.004894989542663097,
0.3383915424346924, -0.042843401432037354, 0.1584128588438034,
0.18772301077842712, 0.2936463952064514, -0.18179874122142792,
0.37588560581207275, 0.4130706191062927, 0.15360291302204132, 0.01841646246612072,
-0.28068825602531433, 0.2961996793746948, 0.25629645586013794,
-0.29467833042144775, 0.19654987752437592, 0.2292499542236328,
-0.059133175760507584, -0.009312686510384083, -0.3014964461326599,

0.0525270514190197, -0.16880634427070618, -0.2337891161441803, -0.160999596118927,
-0.2995942234992981, -0.03845013305544853, 0.5204411149024963, -0.1797948032617569,
0.024470683187246323, -0.11278301477432251, -0.07317347824573517,
-0.18061476945877075, 0.2593151926994324, 0.13253232836723328,
0.22779253125190735, -0.4340859055519104, 0.13811905682086945,
-0.2379612773656845, 0.07031363993883133, 0.16072610020637512,
-0.057924363762140274, -0.06561155617237091, 0.27652788162231445,
0.20673540234565735, 0.12084148824214935, -0.14777597784996033,
-0.49225571751594543, -0.12409155815839767, -0.1605042815208435,
-0.03943762183189392, -0.19633205235004425, -0.12324262410402298,
-0.29399630427360535, 0.24101096391677856, -0.266157329082489,
-0.13057169318199158, -0.1496836543083191, 0.2821232080459595,
-0.36990633606910706, 0.2960594892501831, -0.060822803527116776,
0.21616026759147644, 0.07950606942176819, 0.18559715151786804,
0.006826832890510559, 0.21739646792411804, -0.30613118410110474,
-0.05907691642642021, 0.06314059346914291, -0.08095913380384445,
0.6477255821228027, 0.22445397078990936, -0.1232159435749054, 0.10059922933578491,
0.1784278154373169, 0.22355031967163086, 0.01102500967681408, 0.0061735431663692,
0.05820915848016739, 0.05044769495725632, 0.10405950248241425,
-0.028717534616589546, -0.01965683326125145, 0.13581177592277527,
-0.33760079741477966, 0.05738142877817154, 0.19785305857658386,
-0.3592774271965027, 0.12254936993122101, 0.014594880864024162,
0.14646922051906586, -0.1653304398059845, -0.32953259348869324,
0.39375030994415283, 0.24473994970321655, -0.22963473200798035,
0.06512471288442612, -0.19954876601696014, -0.21642211079597473,
0.1449512243270874, -0.08352641761302948, -0.12657177448272705,
-0.12739866971969604, -0.11784601956605911, 0.18337681889533997,
0.1905369758605957, 0.010059226304292679, -0.4349506199359894,
0.016906704753637314, -0.1784607172012329, -0.005634728819131851,
-0.04858435317873955, -0.044971488416194916, 0.14596495032310486,
-0.36972370743751526, 0.22267340123653412, 0.07848537713289261,
-0.01848289556801319, 0.005747741553932428, -0.0841546505689621,
-0.18099388480186462, 0.48901835083961487, -0.1452580690383911,
0.020265180617570877, -0.3319654166698456, 0.03964856266975403,
0.2661627531051636, -0.05450839176774025, -0.15387745201587677,
0.27029940485954285, -0.2929326891899109, 0.2823336124420166, 0.238254576921463,
0.016583573073148727, -0.16925401985645294, -0.5057194828987122,
-0.15132951736450195, -0.021403789520263672, -0.21081548929214478,
-0.047739386558532715, -0.3041018843650818, -0.1207042783498764,
0.12021839618682861, -0.16486674547195435, -0.17143869400024414,
0.4012249708175659, -0.23395641148090363, -0.10593325644731522,
0.14170777797698975, 0.36996883153915405, -0.12982714176177979,
0.05053696036338806, -0.1059333086013794, 0.07973413169384003,
0.05694451928138733, -0.07974210381507874, 0.23737677931785583,
0.08229488879442215, 0.11486360430717468, -0.21026228368282318,
0.16293203830718994, -0.052409734576940536, 0.10621794313192368,

-0.4303639233112335, -0.10953831672668457, -0.3635135293006897, 0.1291937232017517,
0.2741827070713043, -0.22440284490585327, -0.01271701231598854,
0.18371666967868805, -0.15450789034366608, -0.20596344769001007,
-0.22599229216575623, 0.24088135361671448, 0.029106400907039642,
0.05520486831665039, 0.3484497666358948, 0.06350648403167725,
-0.18966424465179443, -0.03407985344529152, 0.15075311064720154,
0.03893580287694931, -0.20263823866844177, 0.34546715021133423,
0.05961637943983078, -0.03397616744041443, -0.02748371660709381,
-0.01148190163075924, -0.08075294643640518, -0.23410385847091675,
-0.18219289183616638, -0.164737269282341, 0.026137501001358032, 0.270093709230423,
0.24663500487804413, 0.08179921656847, 0.20001184940338135, -0.27707812190055847,
-0.26788583397865295, -0.1467176079750061, -0.23019221425056458,
-0.3590080738067627, 0.022079531103372574, -0.07481943815946579,
-0.16293753683567047, 0.024698644876480103, 0.2898348569869995,
0.5281146764755249, -0.1491101086139679, -0.2845468819141388, 0.14217154681682587,
0.2514115571975708, -0.37382441759109497, -0.05891627445816994,
0.08894307166337967, -0.18299880623817444, 0.06351262331008911,
0.6171760559082031, 0.15187618136405945, 0.09787297248840332,
-0.22165189683437347, 0.11385177075862885, -0.0007827095687389374,
0.25524449348449707, 0.012847725301980972, -0.017589963972568512,
0.08683646470308304, -0.2341023087501526, 0.33607029914855957, 0.3839212954044342,
-0.06258101761341095, 0.20131483674049377, 0.11170955747365952,
0.15246909856796265, 0.142667755484581, 0.05322186276316643, 0.2375032603740692,
-0.06989350914955139, -0.000978674739599228, 0.2724681496620178,
0.07436161488294601, 0.31378844380378723, -0.4562651216983795,
-0.35826432704925537, 0.35069090127944946, 0.05788275972008705,
0.021451346576213837, 0.0785604938864708, 0.0702347606420517,
-0.10176704078912735, -0.2575848698616028, -0.27836859226226807,
0.02904258295893669, 0.26204103231430054, -0.21164673566818237,
-0.24620310962200165, -0.3106624484062195, 0.07806029915809631,
0.5764673948287964, -0.21239866316318512, -0.1766340732574463,
-0.07297438383102417, -0.399915486574173, 0.04234914854168892,
-0.08066609501838684, 0.02626468427479267, -0.25789332389831543,
0.03186359256505966, 0.13546472787857056, -0.43274563550949097,
-0.08893579989671707, -0.005917975679039955, -0.2798762321472168,
-0.04670803248882294, 0.40870973467826843, 0.050057366490364075,
0.07134051620960236, -0.04500512033700943, 0.13647764921188354,
0.5021789073944092, 0.475719153881073, -0.09938511997461319, -0.27453285455703735,
-0.133170485496521, -0.01288534700870514, -0.32911381125450134,
-0.3840865194797516, 0.08607891201972961, 0.23172339797019958,
-0.34538543224334717, 0.17556992173194885, 0.11270421743392944,
0.11785666644573212, 0.020733866840600967, 0.07264429330825806,
0.006047719623893499, 0.08456521481275558, -0.47762173414230347,
-0.3691912889480591, 0.1263880729675293]]]

Here is the cosine similarity code:

Cosine similarity function

def cosine_similarity(vector_1: list, vector_2: list) -> float:

"""

Returns cosine similarity value

Parameters:

vector_1 (list): User query

vector_2 (list): Game collection

Returns:

float: cosine similarity value

"""

if vector_1 and vector_2:

return sum(a * b for a, b in zip(vector_1, vector_2)) / (sum(a * a for a in vector_1) ** 0.5 *
sum(b * b for b in vector_2) ** 0.5)

Function to retrieve relevant documents from MongoDB

def retrieve_relevant_docs(query: list, collection: Collection) -> list:

"""

Retrieve relevant document based on the user query

Parameters:

query (list): User query

collection (pymongo.collection.Collection): game collection

Returns:

list: document relevant to the user query

"""

if query and collection is not None:

model = get_sentence_transformer_model()

if model:

query_vector = model.encode(query).tolist()

docs = list(collection.find())

if docs and query_vector:

relevant_docs = sorted(docs, key=lambda doc: cosine_similarity(query_vector,
doc[DOCUMENT_VECTOR]),
reverse=True)[:5]

return relevant_docs

10.11 Generate response to the user query

OpenAI's GPT-3.5 model is used to generate natural language responses to user queries by processing the user's question along with related game data. Here is the code snippet:

```
if relevant_docs:
    # Generate response
    nextmove_response = generate_response(user_query, relevant_docs)
    assert nextmove_response is not None, "nextmove_response not set."
    print("\n nextmove_response: ", nextmove_response)
```

```
nextmove_response: After the moves 1. d4 Nf6 2. c4 e6 3. Nf3 d5 4. g3 Be7 5. Bg2 O-O 6. O-O c6, World Chess Champion Liren Ding could continue with 7. Qc2. This move prepares for further development of pieces and allows the bishop on g2 to control important central squares. Subsequently, White can look to follow up with moves like Nbd2, Ree1, or cxd5, continuing to build up their position harmoniously.
```

10.12 Save chat history

The following code saves the chat history into the chat_history collection:

```
# Save chat history to MongoDB
save_chat_history(user_query, nextmove_response, chat_history_collection)
print("\n Chat saved into MongoDB collection as embedding!")
```

```
Chat saved into MongoDB collection as embedding!
```

10.13 Verify the number of records inserted into the collections

The following is a book-keeping step to verify the number of records in the games and chat_history collection:

```
# Count the number of documents in the collection
games_count = games_collection.count_documents({})
print(f'Number of documents in the games collection: {games_count}')

chat_history_count = chat_history_collection.count_documents({})
print(f'Number of documents in the chat_history collection: {chat_history_count}')
```

```
Number of documents in the games collection: 4
Number of documents in the chat_history collection: 6
```

Here is the github of this project:

[https://github.com/datariders/NextMOVE/blob/main/src/mongodb_atlas_vector_search_tutorial.i
pynb](https://github.com/datariders/NextMOVE/blob/main/src/mongodb_atlas_vector_search_tutorial.ipynb)

11. Screenshots for using NextMOVE project via Streamlit application

11.1 Step 1



NextMOVE: Chess training assistant

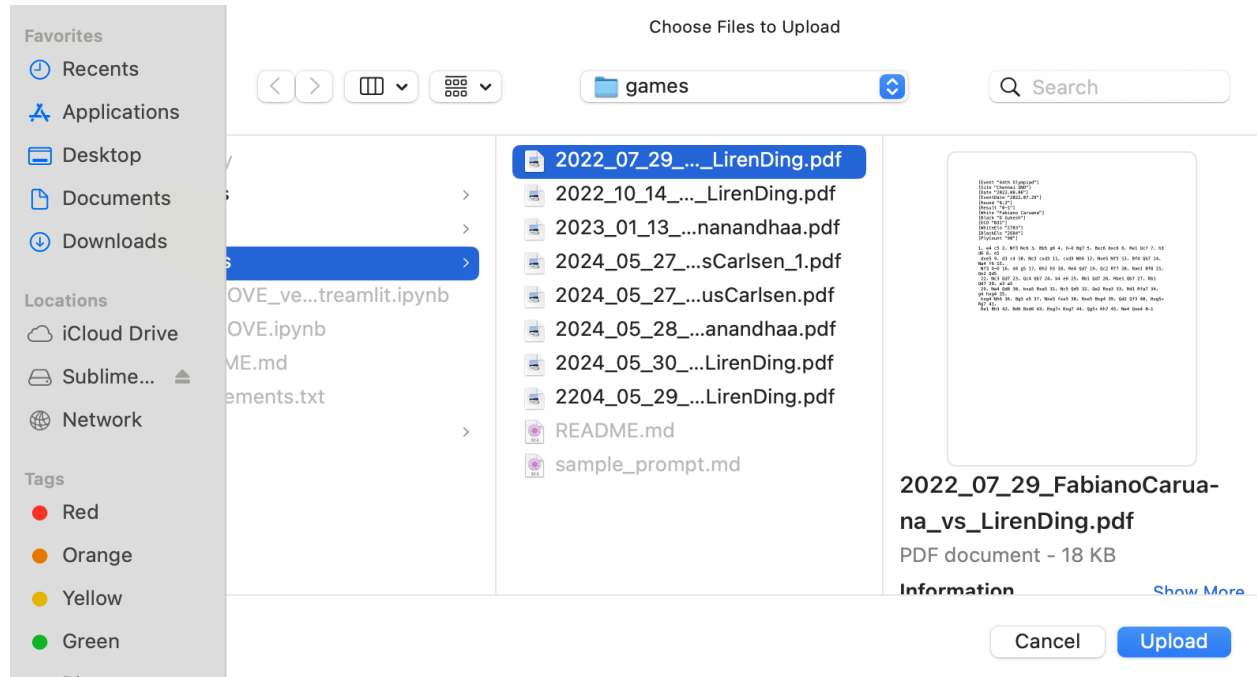
Upload Chess games (pdf)



Drag and drop file here
Limit 200MB per file • PDF

Browse files

11.2 Step 2



11.3 Step 3

```
Extracted game_text:
[Event "44th Olympiad"]
[Site "Chennai IND"]
[Date "2022.08.06"]
[EventDate "2022.07.29"]
[Round "8.2"]
[Result "0-1"]
[White "Fabiano Caruana"]
[Black "D Gukesh"]
[ECO "B31"]
[WhiteElo "2783"]
[BlackElo "2684"]
[PlyCount "90"]
1. e4 c5 2. Nf3 Nc6 3. Bb5 g6 4. O-O Bg7 5. Bxc6 bxc6 6. Re1 Qc7 7. h3
d6 8. e5
  dxe5 9. d3 c4 10. Nc3 cxd3 11. cxd3 Nh6 12. Nxe5 Nf5 13. Bf4 Qb7 14.
Na4 f6 15.
  Nf3 O-O 16. d4 g5 17. Bh2 h5 18. Re4 Qd7 19. Qc2 Rf7 20. Rae1 Bf8 21.
Qe2 Qd5
  22. Nc3 Qd7 23. Qc4 Qb7 24. b4 e6 25. Rb1 Qd7 26. Rbe1 Qb7 27. Rb1
Qd7 28. a3 a5
  29. Na4 Qd8 30. bxa5 Rxa5 31. Nc5 Qd5 32. Qe2 Rxa3 33. Rd1 Rfa7 34.
g4 hxg4 35.
  hxg4 Nh6 36. Bg3 e5 37. Nxe5 fxe5 38. Rxe5 Bxg4 39. Qd2 Qf3 40. Rxc5+
Rg7 41.
  Re1 Bh3 42. Bd6 Bxd6 43. Rxc7+ Kxc7 44. Qg5+ Kh7 45. Ne4 Qxe4 0-1
```

11.4 Step 4

Vectorized game_embedding:

```
[ 0.25676942  0.36840153 -0.06755206 -0.13758254 -0.02003183 -0.09046251
 0.09211384 -0.02257892 -0.18462448 -0.15940635  0.05559648 -0.49572536
 0.127402    0.16313669 -0.08947752 -0.03020037 -0.0057742  -0.36823016
-0.08266652 -0.11545022 -0.1405386  -0.13760419 -0.21334532  0.335554
 0.05343185  0.07255764 -0.12466899  0.1448521  -0.2191895  0.20909159
 0.1447542   0.33113712  0.20040895  0.6012838  -0.03053459  0.0125218
-0.4150141   0.42719927  0.14966106  0.03023952  0.3914388  -0.14401019
 0.21564056  0.16804044  0.3268129   0.20339844  0.09364602  0.13810648
-0.3462704   0.04835731 -0.01199692 -0.10801713 -0.25867954 -0.33037594
-0.0773982  -0.20683141  0.10660318 -0.216869   0.16713482 -0.12191908
-0.21565777 -0.1928049  -0.553481   0.12506682  0.02241102 -0.26241043
 0.2613668   0.045001   0.3200322  0.09029085  0.36296204  0.10028429
-0.08646906 -0.09876223  0.02086758  0.41583702  0.00248114  0.16969065
-0.12222876 -0.4057691   0.06872651 -0.14645186  0.172333   0.00489499
 0.33839154 -0.0428434   0.15841286  0.18772301  0.2936464  -0.18179874
 0.3758856   0.41307062  0.15360291  0.01841646 -0.28068826  0.29619968
 0.25629646 -0.29467833  0.19654988  0.22924995 -0.05913318 -0.00931269
-0.30149645  0.05252705 -0.16880634 -0.23378912 -0.1609996  -0.29959422
-0.03845013  0.5204411  -0.1797948   0.02447068 -0.11278301 -0.07317348
-0.18061477  0.2593152   0.13253233  0.22779253 -0.4340859   0.13811906
-0.23796128  0.07031364  0.1607261  -0.05792436 -0.06561156  0.27652788
 0.2067354   0.12084149 -0.14777598 -0.49225572 -0.12409156 -0.16050428
-0.03943762 -0.19633205 -0.12324262 -0.2939963   0.24101096 -0.26615733
-0.1305717  -0.14968365  0.2821232  -0.36990634  0.2960595  -0.0608228
 0.21616027  0.07950607  0.18559715  0.00682683  0.21739647 -0.30613118
-0.05907692  0.06314059 -0.08095913  0.6477256   0.22445397 -0.12321594
 0.10059923  0.17842782  0.22355032  0.01102501  0.00617354  0.05820916
 0.05044769  0.1040595  -0.02871753 -0.01965683  0.13581178 -0.3376008
 0.05738143  0.19785306 -0.35927743  0.12254937  0.01459488  0.14646922
-0.16533044 -0.3295326   0.3937503   0.24473995 -0.22963473  0.06512471
-0.19954877 -0.21642211  0.14495122 -0.08352642 -0.12657177 -0.12739867
-0.11784602  0.18337682  0.19053698  0.01005923 -0.43495062  0.0169067
-0.17846072 -0.00563473 -0.04858435 -0.04497149  0.14596495 -0.3697237
 0.2226734   0.07848538 -0.0184829   0.00574774 -0.08415465 -0.18099388
 0.48901835 -0.14525807  0.02026518 -0.33196542  0.03964856  0.26616275
-0.05450839 -0.15387745  0.2702994  -0.2929327   0.2823336   0.23825458]
```

0.01658357 -0.16925402 -0.5057195 -0.15132952 -0.02140379 -0.21081549
-0.04773939 -0.30410188 -0.12070428 0.1202184 -0.16486675 -0.1714387
0.40122497 -0.23395641 -0.10593326 0.14170778 0.36996883 -0.12982714
0.05053696 -0.10593331 0.07973413 0.05694452 -0.0797421 0.23737678
0.08229489 0.1148636 -0.21026228 0.16293204 -0.05240973 0.10621794
-0.43036392 -0.10953832 -0.36351353 0.12919372 0.2741827 -0.22440284
-0.01271701 0.18371667 -0.15450789 -0.20596345 -0.22599229 0.24088135
0.0291064 0.05520487 0.34844977 0.06350648 -0.18966424 -0.03407985
0.15075311 0.0389358 -0.20263824 0.34546715 0.05961638 -0.03397617
-0.02748372 -0.0114819 -0.08075295 -0.23410386 -0.18219289 -0.16473727
0.0261375 0.2700937 0.246635 0.08179922 0.20001185 -0.27707812
-0.26788583 -0.14671761 -0.23019221 -0.35900807 0.02207953 -0.07481944
-0.16293754 0.02469864 0.28983486 0.5281147 -0.14911011 -0.28454688
0.14217155 0.25141156 -0.37382442 -0.05891627 0.08894307 -0.1829988
0.06351262 0.61717606 0.15187618 0.09787297 -0.2216519 0.11385177
-0.00078271 0.2552445 0.01284773 -0.01758996 0.08683646 -0.23410231
0.3360703 0.3839213 -0.06258102 0.20131484 0.11170956 0.1524691
0.14266776 0.05322186 0.23750326 -0.06989351 -0.00097867 0.27246815
0.07436161 0.31378844 -0.45626512 -0.35826433 0.3506909 0.05788276
0.02145135 0.07856049 0.07023476 -0.10176704 -0.25758487 -0.2783686
0.02904258 0.26204103 -0.21164674 -0.24620311 -0.31066245 0.0780603
0.5764674 -0.21239866 -0.17663407 -0.07297438 -0.3999155 0.04234915
-0.0806661 0.02626468 -0.25789332 0.03186359 0.13546473 -0.43274564
-0.0889358 -0.00591798 -0.27987623 -0.04670803 0.40870973 0.05005737
0.07134052 -0.04500512 0.13647765 0.5021789 0.47571915 -0.09938512
-0.27453285 -0.13317049 -0.01288535 -0.3291138 -0.38408652 0.08607891
0.2317234 -0.34538543 0.17556992 0.11270422 0.11785667 0.02073387
0.07264429 0.00604772 0.08456521 -0.47762173 -0.3691913 0.12638807]

11.5 Step 5



NextMOVE: Chess training assistant

Upload Chess games (pdf)



Drag and drop file here

Limit 200MB per file • PDF

Browse files



2022_07_29_FabianoCaruana_vs_LirenDing.pdf 17.8KB



Enter your query:

11.6 Step 6



NextMOVE: Chess training assistant

Upload Chess games (pdf)



Drag and drop file here

Limit 200MB per file • PDF

Browse files



2022_07_29_FabianoCaruana_vs_LirenDing.pdf 17.8KB



Enter your query:

What would Liren Ding play after 1. d4 Nf6 2. c4 e6 3. Nf3 d5 4. g3 Be7 5. Bg2 O-O 6. O-O c6

Press Enter to apply

11.7 Step 7

user_query: What would Liren Ding play after 1. d4 Nf6 2. c4 e6 3. Nf3 d5 4. g3 Be7 5. Bg2 O-O 6. O-O c6

11.8 Step 8

nextmove_response: After 6. O-O c6, Liren Ding would likely continue with 7. Qc2, as seen in the game between Ding Liren and Magnus Carlsen.

Chat saved into MongoDB collection as embedding!

11.9 Step 9

Enter your query:

What would Liren Ding play after 1. d4 Nf6 2. c4 e6 3. Nf3 d5 4. g3 Be7 5. Bg2 O-O 6. O-O c6

NextMOVE response:

After 6. O-O c6, Liren Ding would likely continue with 7. Qc2, as seen in the game between Ding Liren and Magnus Carlsen.

12. Code base

The github code base is here:

<https://github.com/datariders/NextMOVE/tree/main>

13. Conclusion

This tutorial demonstrates how to use MongoDB tech stack for building a RAG - a Chess training assistant using OpenAI, Sentence transformer, and Streamlit.

MongoDB is used as a database system for storing, retrieving, and managing data related to chess games and chat history. It provides the necessary infrastructure for persisting vectorized game data and chat interactions, enabling the application to retrieve relevant information based on user queries and maintain a record of user interactions. MongoDB's flexibility and scalability make it well-suited for handling the unstructured and semi-structured data involved in this application.

OpenAI's GPT-3.5 model is used to generate natural language responses to user queries within the chess training assistant application. By processing the user's question along with related game data, GPT-3.5 provides intelligent, context-aware advice., enhancing the overall functionality and interactivity of the application.