International Journal of Management, IT & Engineering
Vol. 13 Issue 12, December 2023
ISSN: 2249-0558 Impact Factor: 7.119
Journal Homepage: http://www.ijmra.us, Email: editorijmie@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gate as well as in Cabell's Directories of Publishing Opportunities, U.S.A

Image Similarity Search Using AWS OpenSearch Service

Prateek Sharma

Abstract

This article provides a practical implementation of image similarity search using AWS OpenSearch Service, employing the Kaggle celebrity dataset as a case study. Image similarity search is animportantuse case in several applications like e-commerce, search engines, and digital document storage. Image similarity search involves identifying images resembling a reference image from a large dataset. This process requires extracting and transforming image features into vector embeddings and storing them in a vector database for efficient retrieval. The article outlines the steps of feature extraction, particularly using the ResNet50 model, and provides the details on storage and retrieval process in AWS OpenSearch Service. Key results from the use of the celebrity face image dataset highlight the robust capabilities of AWS OpenSearch Service in executing image similarity searches with high recall.

Keywords:

Image Similarity Search,
AWS OpenSearch Service,
Vector Embeddings,
Feature Extraction,
ANN.

Author correspondence:

Prateek Sharma,
Masters in Computer Engineering
Email: sharmaprateek10@gmail.com

1. Introduction

Images play a key role in today's applications, which handle a vast amount of image data, ranging from social media and e-commerce sites to search engines. For example, e-commerce websites have several high-quality images of products,search engines keep a large collection of images linked to web documents, and digital document storage services often contain many images. With so many applications dealing with extensive image data, it isan important use case to be able to find images that are similar to each other based on what users are looking for. For example, e-commerce sites can enhance shopping experience by showing customers products similar to what they're currently viewing. Likewise, on a search engine, users might want to find all images similar toones they've uploaded. Handling these kinds of image-related searches efficiently is crucial in today's digital world.

To enable image similarity search, applications use machine learning models to extract meaningful and contextual information from images. This information is transformed into embeddings, or vectors, which are then stored in a vector database. Vector databasesare designed for efficient storage and querying based on vector data. They support algorithms like approximate nearest neighbor (ANN) search, which are crucial for finding the closest matches to a queried vector. ANN search is particularly useful for image similarity searches within large datasets, as it efficiently identifies the most similar images.This paper providesan overview of how image similarity search workswith AWS OpenSearch Service and includes an example implementation using the Kaggle celebrity dataset.

2. Image Similarity Search

In image similarity search, the main goal is to locate images that closely resemble a user-provided reference image from a large collection. This process begins with feature extraction, where distinct characteristics of each image are converted into dense vector embeddings. These embeddings encapsulate crucial elements of the images in a numerical form.Once extracted, these embeddings are stored in a vector database. Vector database is designed for efficient storage and retrieval of high-dimensional vector data, facilitating quick and

effective search operations. When a search query is initiated, the reference image goes through the same feature extraction process to generate its embedding. This embedding is then used as the query vector for similarity search. Vector databasessupport various ANNalgorithms for handling large-scale datasets. These algorithms help the system to quickly go through the large number of image embeddings and identify those most similar to the query embedding.

The subsequent sections provide basic overview of feature extraction, the mechanics of embedding storage, and the methodologies employed in similarity query searches.

Figure 1. *Offline First Reference Architecture*

2.1 Feature Extraction

Feature extraction is an important step in image recognition, as it involves retrieving relevant information and context to construct feature vectors or embeddings. These embeddings are crucial for achieving high accuracy in image similarity searches. There are various techniques for image feature extraction, including Convolutional Neural Networks (CNNs), grayscale features, edge detection, autoencoders, Scale-Invariant Feature Transforms (SIFT), and Local Binary Patterns (LBP). CNNs are often the preferred choice for feature extraction due to their proficiency in detecting local image features like edges and textures, as well as their ability to learn complex hierarchical patterns, such as object parts. Moreover, the availability of several pre-trained CNN models, such as VGG, ResNet, and Xception, which have been trained on extensive image datasets, makes them highly effective and readily usable for various feature extraction tasks.

2.2Storage and Retrieval

Embeddings derived from the feature extraction process are high-dimensional vectors. Vector databases are specialized in storing these embeddings, facilitating fast and precise similarity searches. They simplify the management of vector data by prioritizing semantic or contextual closeness over exact matches, making them ideal for tasks requiring a deeper understanding of content.

Vector databases employ various algorithms to index and query these vector embeddings. These algorithms support ANN searches, employing techniques like hashing, quantization, or graph-based searches. While ANN is less exact than the K Nearest Neighbors (KNN) algorithm, it's more efficient for handling large datasets with high-dimensional vectors, striking a balance between computational intensity and accuracy. When a query for similar vectors is received, the vector database compares the query vector against its indexed vectors to identify the nearest neighbors. There are multiple approaches to constructing vector indexes, including flat indexing, Locality Sensitive Hashing (LSH), Inverted File (IVF) Indexes, and Hierarchical Navigable Small Worlds (HNSW) indexes. Each of these methods offers unique advantages in optimizing search accuracy and speed.

3. Implementation Using AWS OpenSearch Service

This section provides details on how image similarity search can be implemented using AWS OpenSearch service as a vector database.

3.1 Dataset

For this article, 'Celebrity Face Image Dataset' from Kaggle [4] is used. It's a collection of 1800 total images from 18 different famous Hollywood celebrities and contains 100 images for each celebrity. This dataset was chosen for its simplicity and variety of images, providing an ideal basis for demonstrating the effectiveness of image similarity search algorithms. For more information on the dataset, please refer to [1].Out of 1800 images, 80% (1440 images) will be stored in the vector database and the remaining 20% (360 images) will be used for doing similarity search queries.

3.2Feature Extraction

For feature extraction, we have used ResNet50 model which is widely used CNN for image classification tasks. This model is initially loaded with weights that have been pre-trained on the ImageNet dataset.We replaced its top layer with a Global Average Pooling 2D layer to reduce output dimensions while keeping important features. Next, we added a dense layer with 1024 neurons with ReLU activation to capture

complex image patterns. Finally, we concluded the architecture with a 256-neuron logistic layer using softmax activation, converting the patterns into 256-dimensional vector embeddings for each image.
This architecture combines ResNet50's strengths with our specific needs for efficient image similarity search.

3.3Storage and Similarity Search

As part of initial storage, 80% of the images from the 'Celebrity Face Image Dataset' goes through the feature extraction and generated 256-dimensional vector embeddings are stored in the AWS OpenSearch vector database using a Jupyter notebook. AWS OpenSearch service is a fully managed service which provides vector database capabilities. It provides ANN vector search capabilities by building vector indexes from two different algorithms, HNSW and IVF using Non-Metric Space Library (NMSLIB), Facebook AI Similarity Search (FAISS) and Lucene libraries. Additionally, it is highly scalable and available.

A new index, as shown in Figure 1, is created in AWS OpenSearch where each document contains“name” and “image_embedding” fields. “Name” field stores the actual celebrity’s name and “image_embedding” stores the 256-dimensional vector embedding for the image. This index uses NMSLIB engine and HNSW algorithm to support nearest neighbor search.

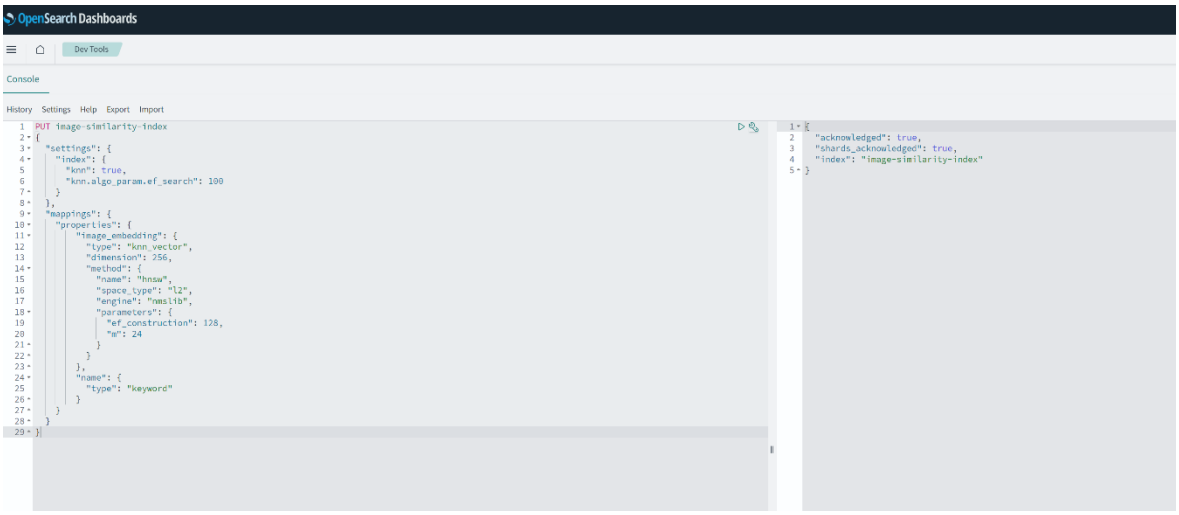

Figure 1. *ANN Search Index Example*

Figure 2 shows an example of ANN query. In this scenario, a vector embedding is generated first using the ResNet model mentioned above for feature extraction. Then this vector embedding is used for the ANN query where K=1 and size=2 which provides 1 nearest neighbor per shard and overall, two results.Both of the results returned by OpenSearch correspond to the same celebrity as the query vector.

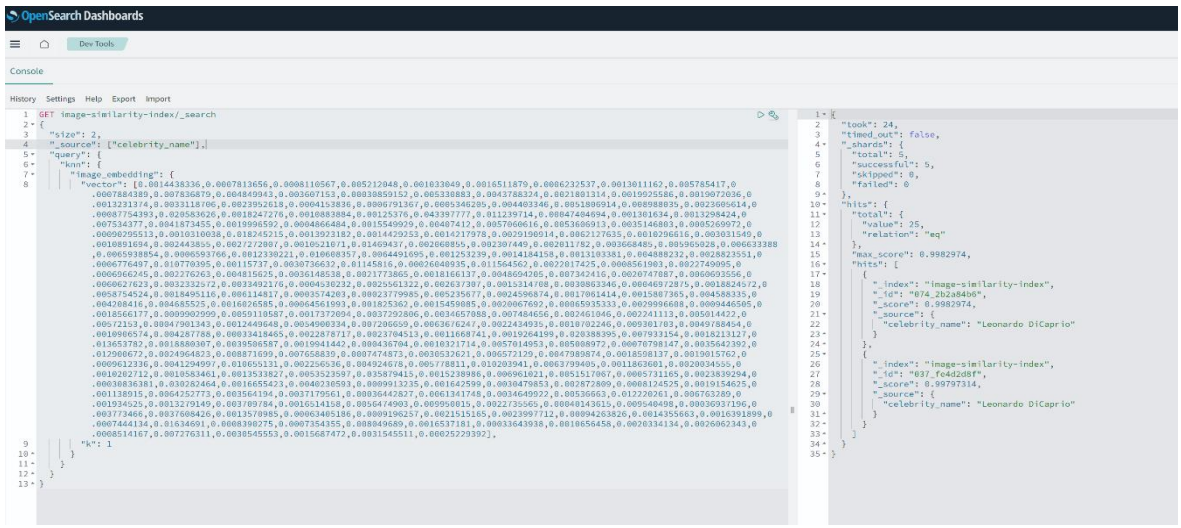

Figure 2. *ANN Similarity Search*

4. Results

After conducting the ANN similarity search on20% (360images)of the dataset, OpenSearch returned the correct celebrity name (at least one result in the response) in 81.9% (295) of the queries. For these queries, we set 'K', the number of nearest neighbors per shard, as 5, with a total response size of 10.

It is important to recognize that recall usually depeds on a variety of factors, including the number of vectors, their dimensions, the configuration of clusters, and the characteristics of the dataset. Therefore, a detailed understanding of the system's requirements regarding latency and accuracy is essential for determining the right configuration.

5. Conclusion

This article has demonstrated the practicality and effectiveness of using AWS OpenSearch Service for image similarity search. Our implementation with the Celebrity Face Image Dataset from Kaggle shows how a practical image similarity search system can be implemented using CNN based feature extraction and AWS OpenSearch Service as a vector database. The use of ANN algorithms within a scalable platform like AWS OpenSearch Service demonstrates a powerful approach to handling extensive image datasets. Future work could focus on refining the feature extraction process, exploring different neural network architectures, and experimenting with other indexing and search algorithms to further enhance the recall. Additionally, applying this methodology to diverse and larger datasets would provide more insights into its scalability and adaptability to different use cases.

References

[1] Euripides G.M. Petraki, and Christos Faloutsos., "Similarity Searching in Medical Image DataBases," IEEE Transactions on Knowledge and Data Enginnering, Vol. 9 No. 3 pp. 435-447, May/June 1997

[2] Moise, Diana & Shestakov, Denis & Guðmundsson, Gylfi & Amsaleg, Laurent, "Terabyte-scale image similarity search: Experience and best practice", Proceedings - 2013 IEEE International Conference on Big Data, Big Data 2013. 674-682. 10.1109/BigData.2013.6691637, 2013

[3] Kumar, Gaurav & Bhatia, Pradeep., "A Detailed Review of Feature Extraction in Image Processing Systems", 2014 Fourth International Conference on Advanced Computing & Communication Technologie, 10.1109/ACCT.2014.74.,2014

[4] Agarwal, V. (2020). Celebrity face image dataset. Retrieved from Vishesh Agarwal's Celebrity Face Image Dataset on Kaggle.

[5] Amazon Web Services, "Amazon OpenSearch Service's vector database capabilities explained. Retrieved from https://aws.amazon.com/blogs/big-data/amazon-opensearch-services-vector-database-capabilities-explained/", Jun 2023

[6] Lv, Q., Charikar, M., & Li, K., "Image Similarity Search with Compact Data Structures.", In CIKM '04: Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management, pp. 208-217,2004