elastic | The Search AI Company

KNN-Based Image Search with Elasticsearch

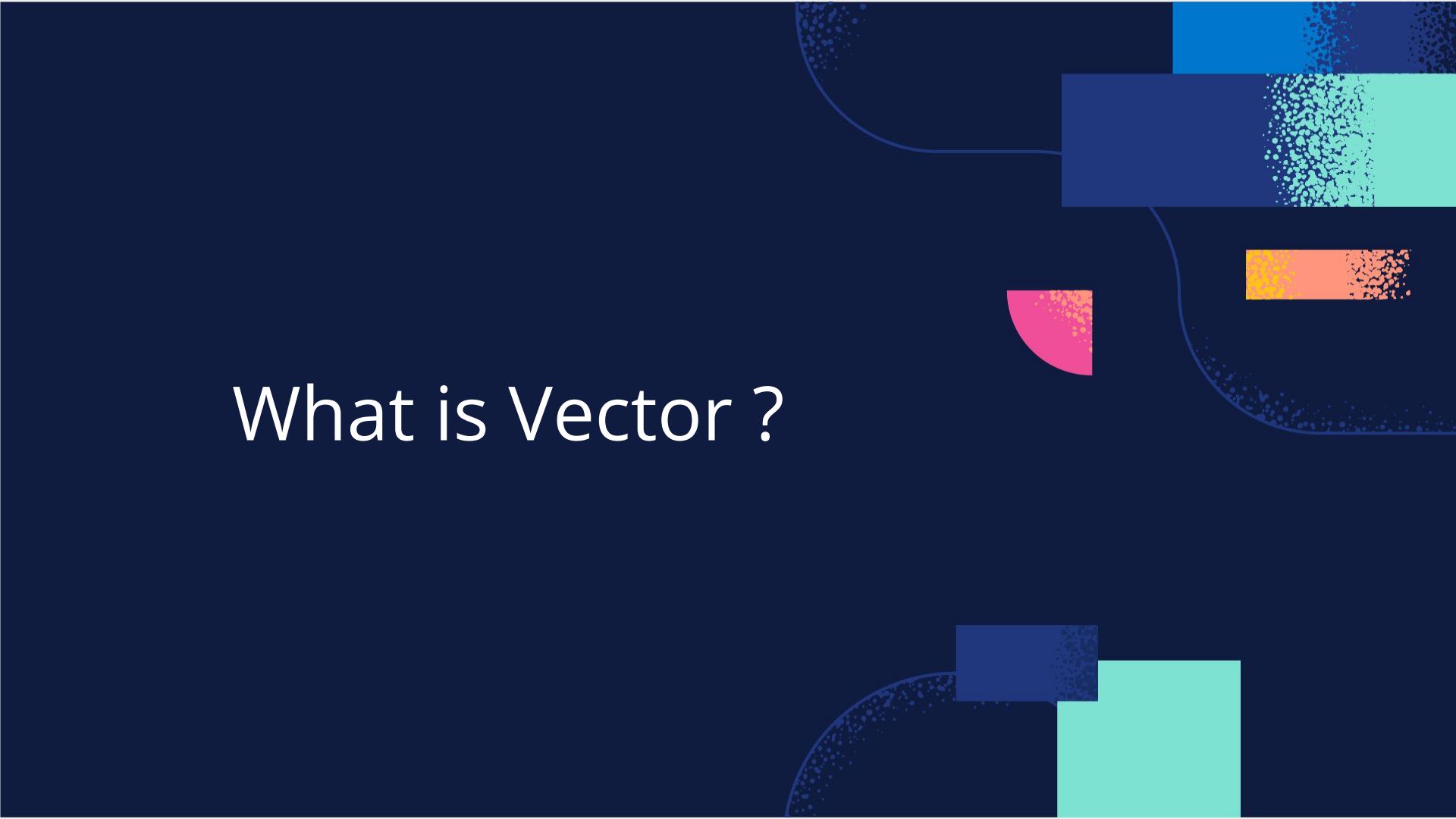
Leveraging vector embeddings for efficient image similarity search

NAVIN BALAJI RK

Introduction

- Problem: Searching for similar images is complex using traditional methods.
- Goal: Use image embeddings + KNN to search similar images efficiently.
- Tools: Javascript, CLIP, Elasticsearch, KNN vector search

What is Vector ?

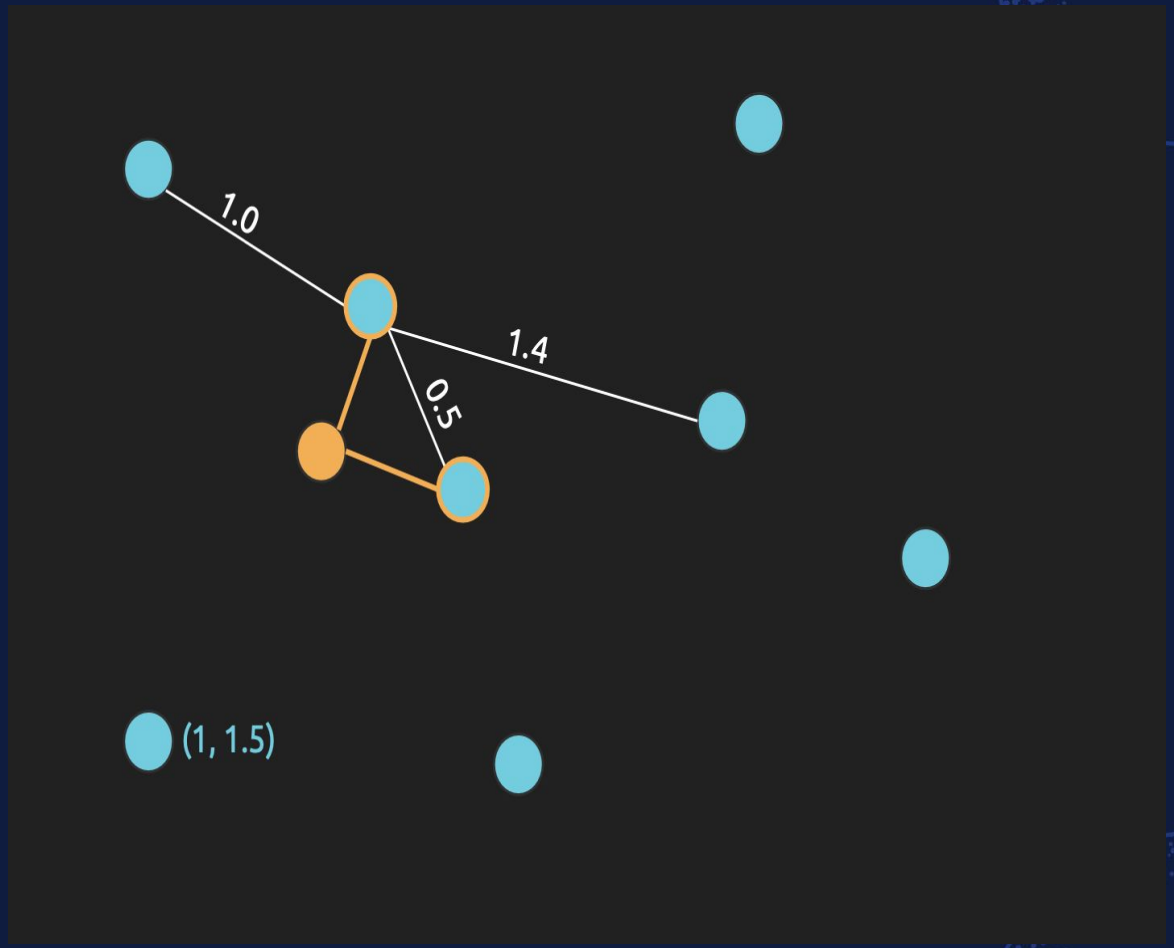


Vector embeddings are a way to convert words and sentences and other data like images into numbers
that capture their meaning and relationship

Different type of Vectors

1. Dense Vector
2. Sparse Vector

What is KNN?


1.0
1.4
0.5
(1, 1.5)

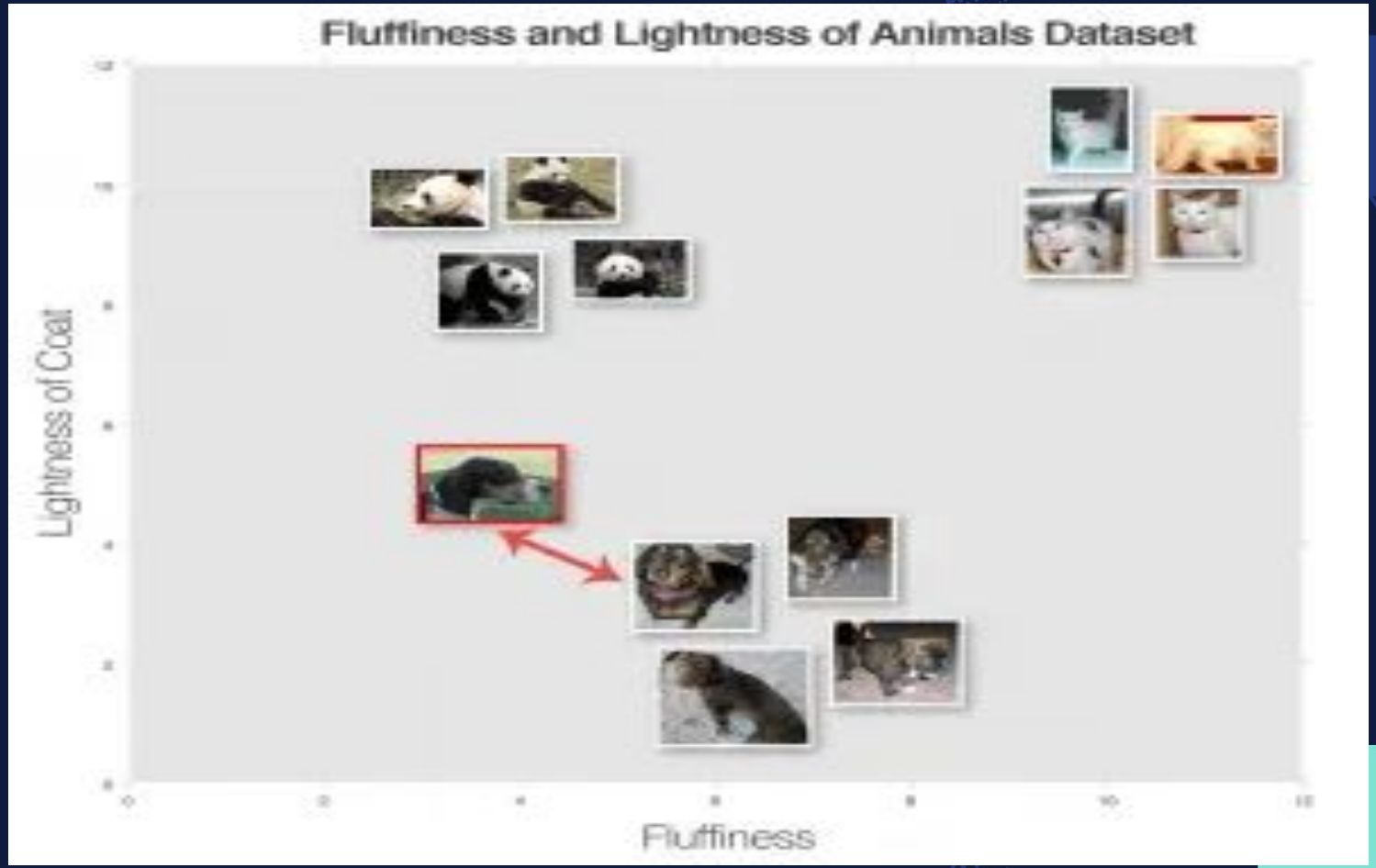
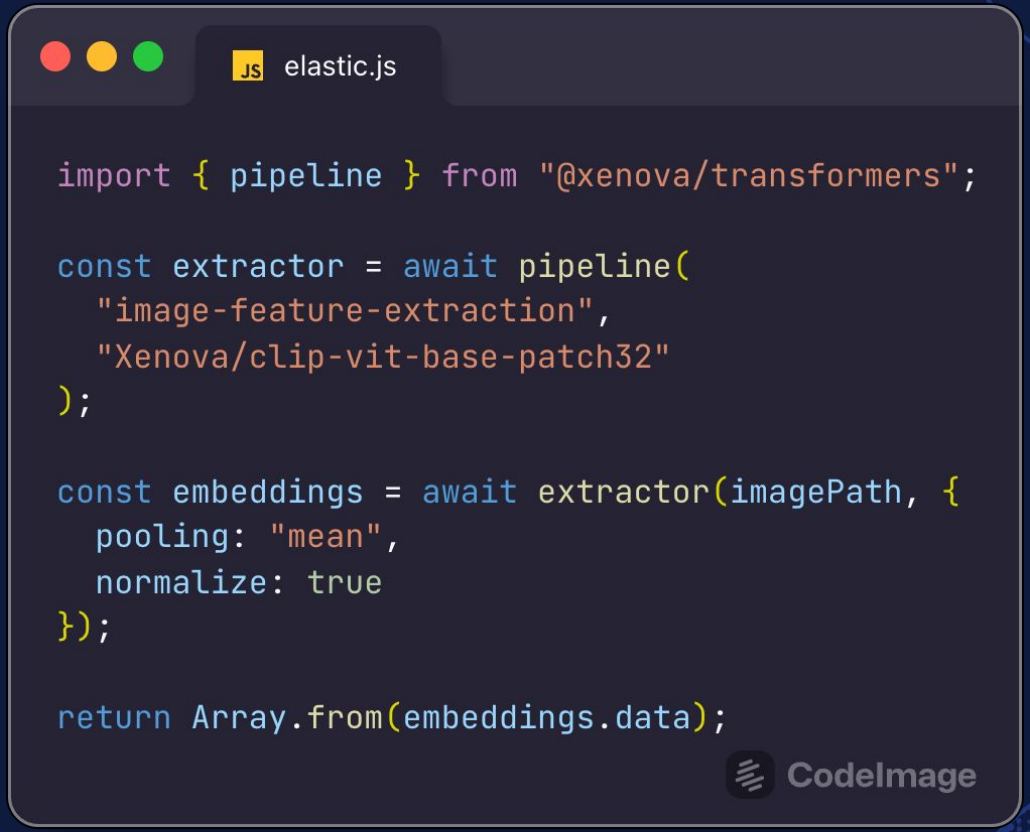

Fluffiness and Lightness of Animals Dataset
Lightness of Coat
Fluffiness

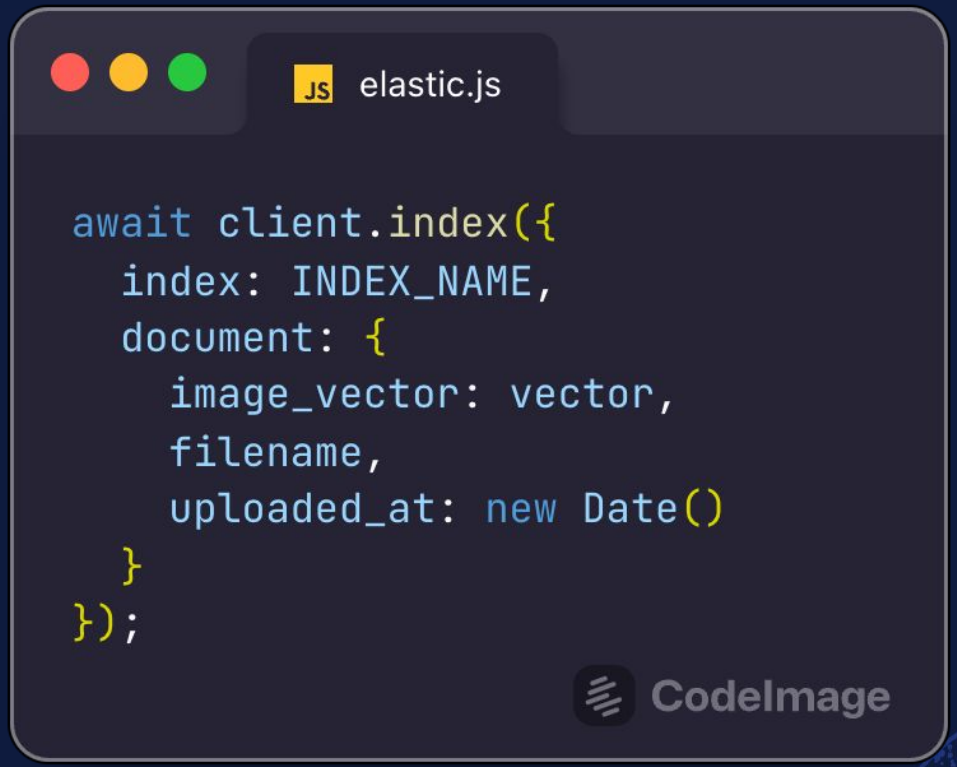
Image Vectorization

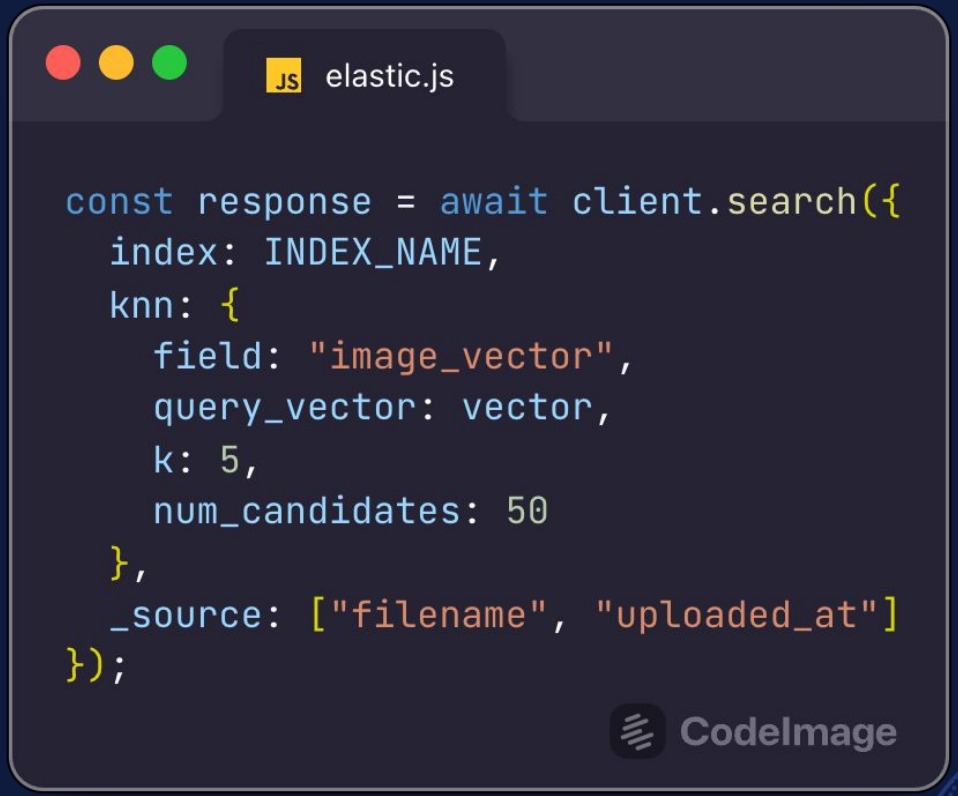
- Use pre-trained models like OpenAI CLIP or Xenova/clip-vit-base-patch32
- Convert images to 512-D


elastic.js
import { Client } from "@elastic/elasticsearch";
const client = new Client({
node: process.env.ELASTIC_NODE,
auth: {
username: process.env.ELASTIC_USERNAME,
password: process.env.ELASTIC_PASSWORD
}
});
CodeImage


elastic.js
await client.indices.create({
index: INDEX_NAME,
body: {
mappings: {
properties: {
image_vector: {
type: "dense_vector",
dims: 512,
index: true,
similarity: "cosine"
},
filename: { type: "keyword" },
uploaded_at: { type: "date" }
}
}
}
});
CodeImage


elastic.js
import { pipeline } from "@xenova/transformers";
const extractor = await pipeline(
"image-feature-extraction",
"Xenova/clip-vit-base-patch32"
);
const embeddings = await extractor(imagePath, {
pooling: "mean",
normalize: true
});
return Array.from(embeddings.data);
CodeImage


elastic.js
await client.index({
index: INDEX_NAME,
document: {
image_vector: vector,
filename,
uploaded_at: new Date()
}
});
CodeImage


elastic.js
const response = await client.search({
index: INDEX_NAME,
knn: {
field: "image_vector",
query_vector: vector,
k: 5,
num_candidates: 50
},
_source: ["filename", "uploaded_at"]
});
CodeImage

Elasticsearch Plugin for Nearest Neighbor Search

The `knn` search option accepts a number of parameters that configure the search:

- `field`: the field in the index to search. The field must have a `dense_vector` type.
- `query_vector`: the embedding to search for. This should be an embedding generated from the search text.
- `num_candidates`: the number of candidate documents to consider from each shard. Elasticsearch retrieves this many candidates from each shard, combines them into a single list and then finds the closest "k" to return as results.
- `k`: the number of results to return. This number has a direct effect on performance, so it should be kept as small as possible. The value passed in this option must be less than `num_candidates`.

Resources

- Dense vector field type | Elasticsearch Guide [8.17] | Elastic
- k-nearest neighbor (kNN) search | Elasticsearch Guide [8.17] | Elastic
- Knn query | Elasticsearch Guide [8.17] | Elastic
- https://www.elastic.co/search-labs/blog/implementing-image-search-with-elasticsearch
- https://huggingface.co/Xenova/clip-vit-base-patch32
- https://alexklibisz.github.io/elastiknn

Questions?

Thank You

More about me