**Tìm hiểu về Elasticsearch, MTCNN, FaceAPI và RetinaFace**

1. **Elasticsearch**
2. Định nghĩa, tổng quan

* Là một công cụ tìm kiếm dựa trên nền tảng Apache Lucene. Nó cung cấp một bộ máy tìm kiếm dạng phân tán, có đầy đủ công cụ với một giao diện web HTTP có hỗ trợ dữ liệu JSON.
* Là 1 hệ thống tìm kiếm và phân tích dữ liệu phân tán mã nguồn mở, được thiết kế để lưu trữ, tìm kiếm và phân tích các tài liệu có cấu trúc hoặc không cấu trúc với tốc độ cao.
* Sử dụng cơ chế ngược (inverted index) để tối ưu hóa hiệu suất tìm kiếm và cho phép truy vấn phức tạp trên lượng dữ liệu lớn.
* Hỗ trợ nhiều tính năng: full-text search, autocomplete, facetting, aggregation và analytics.
* Có thể tích hợp với nhiều công nghệ khác nhau: Logstash, Kibana, Beats và các công nghệ clould computing khác để xây dựng các ứng dụng phân tích dữ liệu, quản lý log và giám sát hệ thống.

1. Ưu, nhược điểm:
2. Ưu điểm:
   * Tìm kiếm dữ liệu mạnh mẽ, nhanh chóng.
   * Có khả năng phân tích dữ liệu.
   * Hỗ trợ tìm kiếm mờ (từ khóa tìm kiếm có thể bị sai lỗi chính tả hay không đúng cú pháp thì vẫn có khả năng trả về kết quả tốt)
   * Khả năng mở rộng theo chiều ngang tuyệt vời
   * Hỗ trợ nhiều ngôn ngữ: Java, PHP, Javascript. Rupy, .NET, Python
3. Nhược điểm:
   * Với những nhiệm vụ khác ngoài search, elasticsearch kém thế hơn so với những database khác.
   * Không có ràng buộc giữa các dữ liệu nên không đảm bảo tính toàn vẹn dữ liệu.
   * Không thích hợp với những hệ thống thường xuyên cập nhật dữ liệu
   * Tốn kém cho việc đánh index dữ liệu.

* Thường được dùng như một công cụ tìm kiếm nhiều hơn là dùng để lưu trữ dữ liệu.

1. Tìm kiếm vecto trong Elasticsearch

* Sử dụng phương pháp “nearest neighbor search” (tìm kiếm láng giềng gần nhất) bằng cách sử dụng plugin Elasticsearch Vector Scoring. Plugin này cho phép lưu trữ và tìm kiếm các vecto bằng cách sử dụng các thuật toán như consine similarity hoặc Euclidean distance để tìm ra các vecto giống nhau nhất.
* Có thể sử dụng các công cụ khác như Annoy hoặc FAISS để kết hợp với Elasticsearch để tìm kiếm vecto nhanh hơn.
* Các bước thực hiện:

+ Lưu trữ các vecto cần thiết vào Elasticsearch

+ Tạo 1 chỉ mục cho các vector đó và sử dụng plugin Elasticearch Vector Scoring để lưu trữ chúng

+ Sử dụng câu truy vấn tìm kiểm để tìm ra các vector gần giống nhất với vector đầu vào. Có thể sử dụng thuật toán cosine similarity hoặc Euclidean distance để tính toán độ tương đồng giữa các vector.

+ Sắp xếp kết quả theo độ tương đồng và lấy ra kết quả cuối cùng

1. **MTCNN**
2. Định nghĩa

* MTCNN là viết tắt của “Multi-task Cascaded Convolutional Networks”, là 1 mô hình deeplearning được sử dụng cho face detection (phát hiện khuôn mặt) và face recognition (nhận diện khuôn mặt)
* MTCNN sử dụng chuỗi các convolutional neural networks (CNNs), mỗi mạng CNN đóng vai trò phát hiện và xác định khuôn mặt trong 1 vùng quan tâm của hình ảnh => chuyển qua các bộ phận khác để nhận diện và xác minh danh tính khuôn mặt.

1. Ưu, nhược điểm:
2. Ưu điểm:

* Hiệu suất cao: khả năng phát hiện khuôn mặt với độ chính xác rất cao, thậm chí còn hoạt động tốt trên các bức ảnh có độ nghiêng, xoay hoặc không đồng nhất.
* Tính đa nhiệm: Ngoài việc phát hiện khuôn mặt, MTCNN còn có thể trích xuất đặc trưng khuôn mặt, giúp cho việc nhận diện khuôn mặt trở nên dễ dàng hơn.
* Khả năng phát hiện được nhiều khuôn mặt trong 1 bức ảnh => xử lý ảnh nhanh chóng và tiết kiệm thời gian.

1. Nhược điểm:

* Yêu cầu tài nguyên tính toán cao -> có thể gây ra việc chậm hoặc không ổn định cho hệ thống
* Không hiệu quả khi áp dụng trên ảnh với độ phân giải thấp
* Độ nhạy cảm với vùng ánh sáng: MTCNN có thể bị ảnh hưởng bởi vùng ánh sáng trên ảnh -> khả năng phát hiện khuôn mặt có thể bị giảm xuống.

1. Các phương thức được cung cấp bởi thư viện dùng MTCNN để tiếp cận:

* MTCNN.detect\_faces(image): Phát hiện khuôn mặt trong ảnh và trả về các bounding box (hộp giới hạn) của các khuôn mặt đó.
* MTCNN.extract\_face(image, bbox, save\_path=None): Trích xuất khuôn mặt từ ảnh ban đầu theo bounding box được xác định bởi phương thức detect\_faces và lưu vào đường dẫn được chỉ định bởi save\_path.
* MTCNN(): Khởi tạo 1 đối tượng MTCNN để sử dụng cho phát hiện và trích xuất khuôn mặt.

1. **FaceAPI**
2. Khái niệm:

* FaceAPI là 1 công cụ phân tích khuôn mặt do Microsoft cung cấp. Nó sử dụng các thuật toán học sâu để phát hiện và phân tích các đặc trưng của khuôn mặt: vị trí, giới tính, tuổi, biểu cảm, điểm đặc biệt (như mắt, mũi, miệng) và nhiều hơn nữa.
* FaceAPI có thể được sử dụng để phát hiện khuôn mặt trong hình ảnh hoặc đoạn video, xác định các thuộc tính của khuôn mặt và theo dõi phân tích biểu cảm của khuôn mặt trong thời gian thực.

1. Ưu, nhược điểm:
2. Ưu điểm:

* Có khả năng phát hiện và phân tích các đặc trưng khuôn mặt như vị trí, tuổi, giới tính, biểu cảm, với độ chính xác cao.
* Cung cấp API để tích hợp vào các ứng dụng của người dùng.
* Được cung cấp bởi Microsoft, là một tập đoàn công nghệ có uy tín lớn.

1. Nhược điểm:

* Đòi hỏi dữ liệu đầu vào là hình ảnh hoặc video có chất lượng tốt để đạt được độ chính xác cao.
* Tốc độ xử lý có thể chậm đối với ảnh hoặc video lớn vì đòi ỏi sử dụng các thuật toán học sâu phức tạp.
* Việc sử dụng FaceAPI đòi hỏi phải trả phí cho việc sử dụng.

1. Các hàm chính của FaceAPI

* face.detectWithUrl(): phát hiện khuôn mặt trong hình ảnh được lưu trữ trên một URL
* face.detectWithStream(): Phát hiện khuôn mặt trong hình ảnh được lưu trữ trên sever.
* face.detectWithBuffer(): phát hiện khuôn mặt trong hình ảnh được lưu trữ trên máy tính của người dùng
* face.identify(): nhận dạng danh tính của 1 người dựa trên thông tin khuôn mặt
* face.verify(): xác nhận xem 2 hình ảnh là của 1 người hay không
* face.findSimilar(): tìm kiếm các khuôn mặt tương tự trong CSDL
* face.group(): phân nhóm các khuôn mặt tương tự trong CSDL
* face.landmarks(): truy xuất vị trí các điểm đặc trưng trên khuôn mặt: mắt, mũi, miệng
* face.pose(): xác định hướng và góc xoay của khuôn mặt
* face.expressions(): phân tích biểu cảm khuôn mặt

1. **RetinaFace**
2. Tổng quan:

* RetinaFace là một mô hình các lớp mạng neural được sử dụng để phát hiện khuôn mặt trong ảnh. Nó được xây dựng trên kiến trúc RetinaNet, là một trong những kiến trúc mạng neural hiệu quả nhất để giải quyết vấn đề object detection.
* RetinaFace có thể cùng lúc phát hiện vị trí khuôn mặt và các điểm đặc trưng của khuôn mặt, bao gồm vị trí của mắt, mũi và miệng.
* RetinaFace cung cấp độ chính xác cao trong việc xác định vị trí và điểm đặc trưng của khuôn mặt, đặc biệt là cho các khuôn mặt nhỏ hoặc cách xa camera.

1. Ứng dụng của RetinanFace:

* Nhận diện khuôn mặt trong ảnh hoặc video surveillance
* Chỉnh sửa và tạo ảnh selfie đẹp
* Xác định được người dùng trong các dịch vụ trực tuyến
* Tăng cường an ninh tại các nơi công cộng

1. Ưu, nhược điểm:
2. Ưu điểm:

* Độ chính xác cao: khả năng phát hiện khuôn mặt với độ chính xác rất cao, bao gồm cả các khuôn mặt nhỏ hoặc ở khoảng cách xa.
* Tốc độ xử lý nhanh
* Đa nhiệm: Không chỉ phát hiện vị trí khuôn mặt mà còn phát hiện được các điểm đặc trưng của khuôn mặt, giúp cho việc xác định chính xác danh tính người dùng hơn.
* Có khả năng phát hiện khuôn mặt theo nhóm

1. Nhược điểm:

* Yêu cầu phần cứng mạnh
* Đòi hỏi dữ liệu huấn luyện lớn
* Phụ thuộc vào độ phân giải ảnh: cần 1 độ phân giải ảnh cao để đạt được độ chính xác cao.

1. Sử dụng RetinaFace để nhận diện khuôn mặt

* Cài đặt RetinaFace
* Tải ảnh cần nhận diện khuôn mặt
* Sử dụng phương thức RetinaFace.detect\_faces để phát hiện tất cả các khuôn mặt có trong ảnh và trả về một danh sách các bounding boxes chứa khuôn mặt.