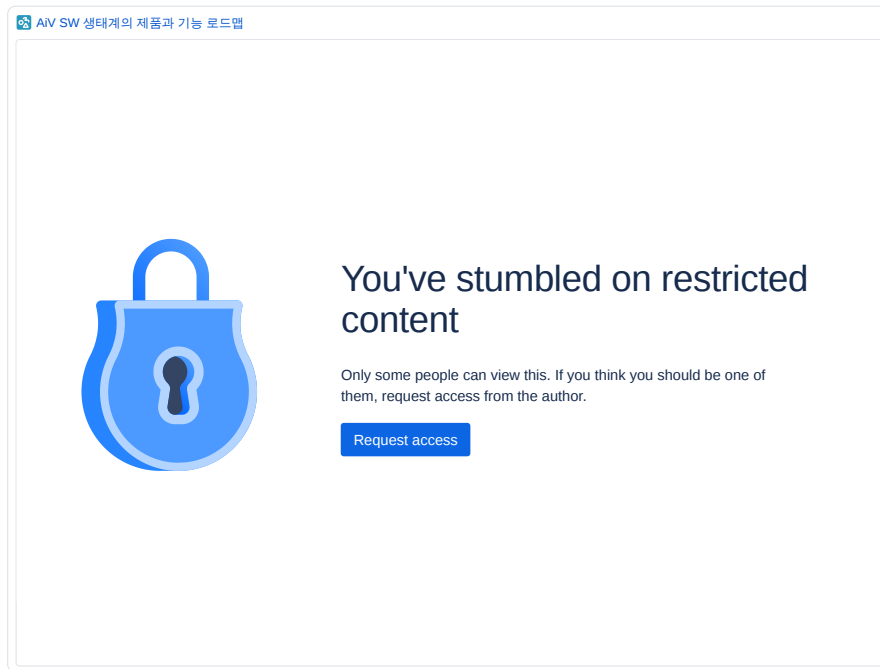


[SRS] Image embedding feature extraction & Image comparison

URS, 설계 문서, Jira 링크 [↗](#)

.



문서 정보

1. 문서 버전:
2. 작성일: 2024년 5월 16일
3. 작성자: @park.jihoon

개요

- 제품 개요: AiV 의 이미지 서비스는 다양한 AiV 서비스에서 사용하는 이미지들을 관리하는 역할을 한다.
- 목적: 이미지를 업로드 할 때 업로드 되는 이미지의 feature 를 추출하여 유사도 검색에 활용한다.
- 범위: AiVOps, AiVData, AiVision 등 다양한 곳에서 활용될 수 있도록 범용적인 interface 가 필요함.
 - 다양한 이미지 크기, 이미지 형식, 다양한 domain (금속, 플라스틱, 고무 등 다양한 재질 및 다양한 불량 유형)에 적용 가능해야 함
- 관련 문서:
 - [Embedding이란 무엇이고, 어떻게 사용하는가? - 싱클리\(Syncly\)](#)
 - [스캐터 필터\(Scatter Filter\)](#)
 - [오토 큐레이트\(Auto-Curate\)](#)
- 용어 정의:
 - Embedding 은 데이터를 저차원 벡터 공간에 매핑하는 기술입니다.
 - vector database는 벡터 검색을 지원하기 위해 벡터 데이터를 저장하고 관리하는 데이터베이스 시스템입니다.

시스템 전반적인 설명

1. 시스템 아키텍처: [AiV SW 생태계의 제품과 기능 로드맵](#) 참고
2. 주요 기능: AiV 의 이미지 서비스는 AiVOps, AiVData, AiVision 등 다양한 AiV 서비스에서 이미지를 업로드하거나 다운로드 할 수 있도록 관리하는 역할을 한다.
3. 사용자의 역할과 책임:

기능적 요구사항

- 사용 사례 (Use Cases):
 - [스캐터 필터\(Scatter Filter\)](#)
 - [오토 큐레이트\(Auto-Curate\)](#)
- 상세한 기능 명세:
 - embedding 기능
 - CPU 에서 동작 가능해야 함
 - 이미지 사이즈에 관계 없이 동작할 것
 - embedding vector 의 차원 수 제한은 없음. 다만 유사도 검색 시 성능 저하가 발생하면 안되고, 유사도가 분류될 수 있을만한 수준이어야 함
 - 유사도 검색
 - vector database 를 활용하여 이미지 유사도 검색이 가능해야 함
 - cosine similarity
 - 같은 그룹의 이미지끼리 유사도가 높고, 다른 그룹의 이미지는 유사도가 낮아 분류기의 기능을 동작해야 함
 - 시각화를 위한 2D projection
 - 고차원의 vector dataset 을 시각화하여 확인하기 위해 2d projection 을 지원해야 함

비기능적 요구사항

1. 성능 요구사항:
 - a. Embedding 관련
 - i. 1024x1024 크기 기준 50msec 이내로 동작할 것 (embedding 작업만)
 - 이미지 크기 별 embedding 시간이 분석되어야 함
 - ii. embedding vector 의 차원 수 제한은 없음. 다만 유사도 검색 시 성능 저하가 발생하면 안되고, 유사도가 분류될 수 있을만한 수준이어야 함
 - dimension 에 따른 유사도 검색 시간, 검색 정확도 분석되어야 함
 - b. 유사도 검색 관련
 - i. dataset 10,000, 100,000 각각의 데이터셋 사이즈에 따른 유사도 검색 시간이 분석되어야 함
2. 보안 요구사항:
3. 가용성 요구사항:
 - a. embedding, 유사도 검색 진행으로 인해 다른 API 요청이 blocking 되어서는 안됨
4. 사용성 요구사항:
 - a. 개발되는 language 는 상관 없으나, node js, go 환경에서 호출이 가능할 것

- b. Docker 환경에서 구동되어야 함


데이터 요구사항

- 데이터 모델:
- 데이터베이스 요구사항:
 - open source database 를 사용할 것

시스템 제약 사항

1. 하드웨어 제약: GPU 를 사용할 수 없는 환경
2. 소프트웨어 제약:
3. 기타 제약:

품질 요구사항

- 코드 품질:
 -  Python
 - AiV 의 코드 가이드라인을 따를 것
- 테스트 전략:
 - 기능 별 unit test case 를 확보하여 code coverage 60% 이상 확보할 것
- 유지보수성:
 - 유지 보수 가능한 코드 품질을 가질 것
 - 의미 있는 변수명, 함수명을 가질 것
 - 하나의 함수 길이는 스크롤 없이 확인할 수 있을 것
 - 추상화된 클래스를 상속받는 형태로 기능을 구현하여 추후 확장성을 고려할 것

프로젝트 일정 및 일정 계획

1. 프로젝트 일정: 2024년 6월 1일 ~ 2024년 7월 31일
2. 개발 단계: 초기 한 달간 embeddin 알고리즘 구현 및 검증, 이후 한 달간 서비스에 적용 및 테스트
3. 테스트 단계: 개발 단계에 포함
4. 유지보수 단계: 유지보수는 기존 팀원들이 이어서 진행

비용 추정

- 개발 비용: 1인 개발자 2개월
- 운영 비용:

위험 관리

1. 주요 프로젝트 위험: 없음
2. 대응 계획: