

추억 사진관

CV - 14

발표자 : 신건희

팀원 : 김우진, 신건희, 신중현, 이종휘

INDEX

문제 정의 및 주제 제안

- 프로젝트 설명
 - 문제 정의 및 방향성
-

데이터셋 & 평가 메트릭

- 데이터 셋
 - 평가 메트릭
-

사용 모델 선정 및 Fine-tuning

- 모델 선정
 - 모델링 & fine-tuning
-

Product serving

- service architecture
-

시연 영상

- Result & Conclusion

팀원 & 역할



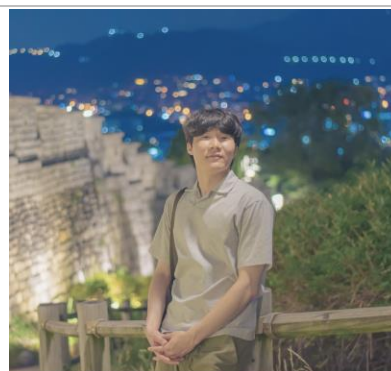
김우진

- Data Crawling
- AWS S3 서버구축
- mongoDB를 통한 db구축
- Flask api와 node.js 연결 구축
- Backend
- Frontend



신건희

- Data Crawling
- 모델 서치 & 모델링
- Train 파이프 라인 구현
- 평가 Metric 구현



신중현

- Data Crawling
- 모델 서치 & 모델링
- Pretrained & 경량화 탐색
- Frontend



이종휘

- Data Crawling
- 모델 서치 & 모델링
- 모델 실험 및 평가
- Pretrained model fine-tuning

1. 프로젝트 소개

1. 프로젝트 소개
2. 문제 정의
3. 방향성 소개

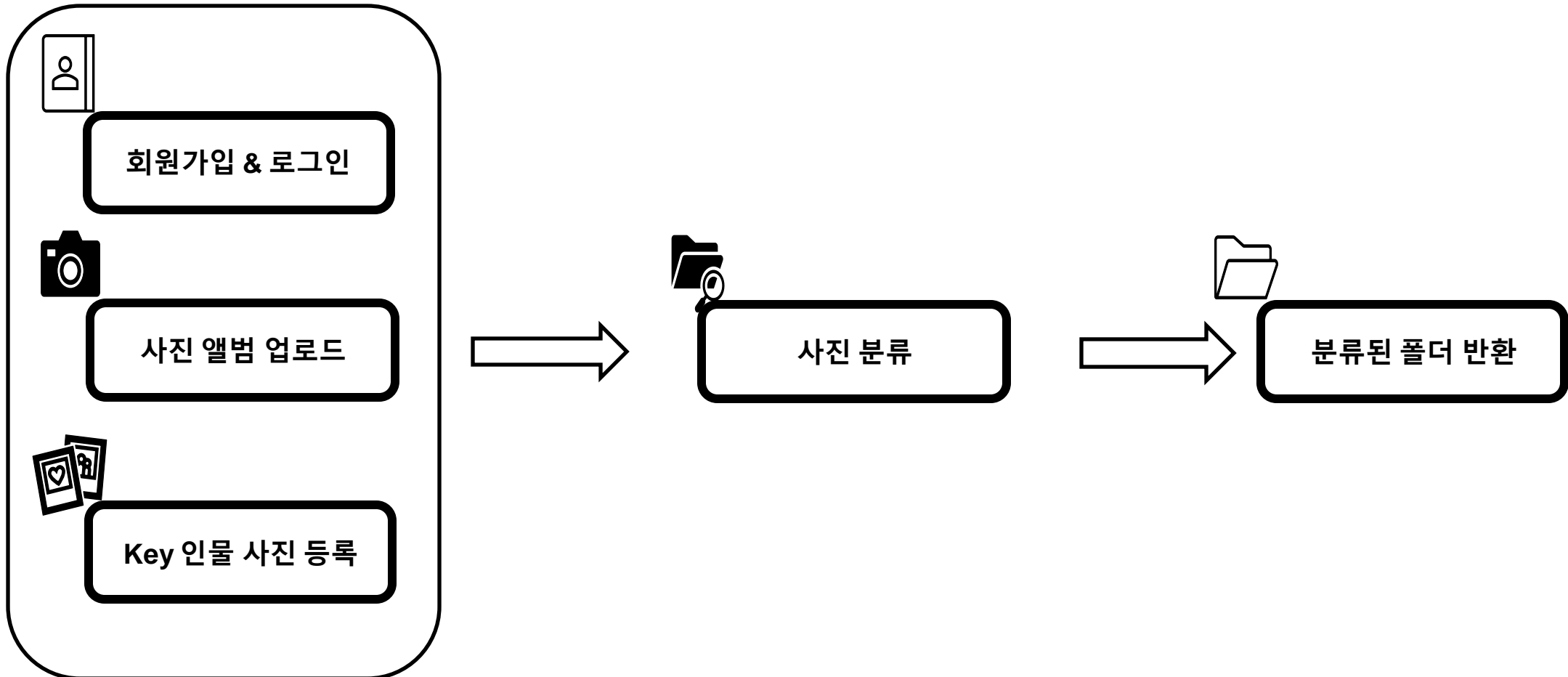
추억 사진관

추억사진관은 당신의 사진앨범에서 **당신이 원하는 인물이 있는 사진**만을 구별해주는 서비스입니다.



추억 사진관

추억사진관은 당신의 사진앨범에서 **당신이 원하는 인물이 있는 사진**만을 구별해주는 서비스입니다.



추억 사진관

추억사진관은 당신의 사진앨범에서 **당신이 원하는 인물이 있는 사진**만을 구별해주는 서비스입니다.

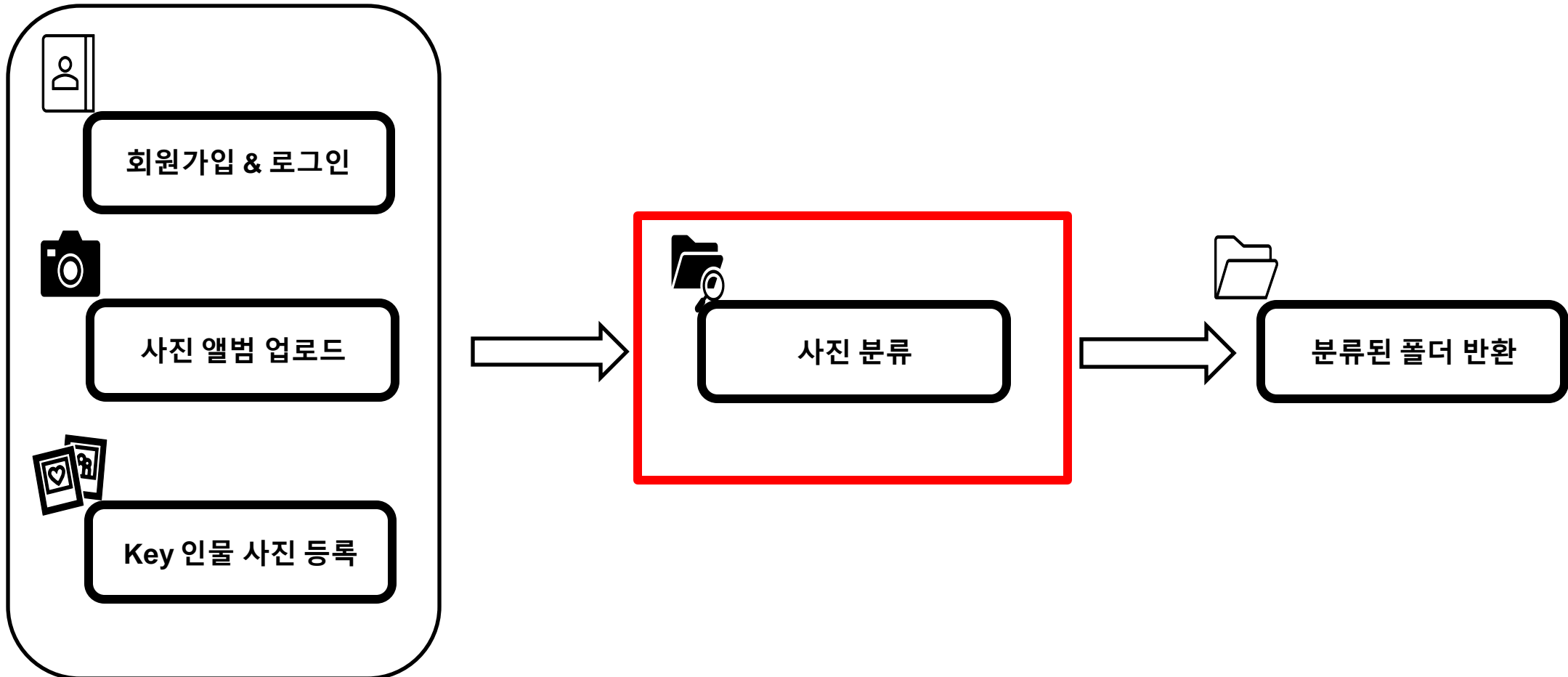
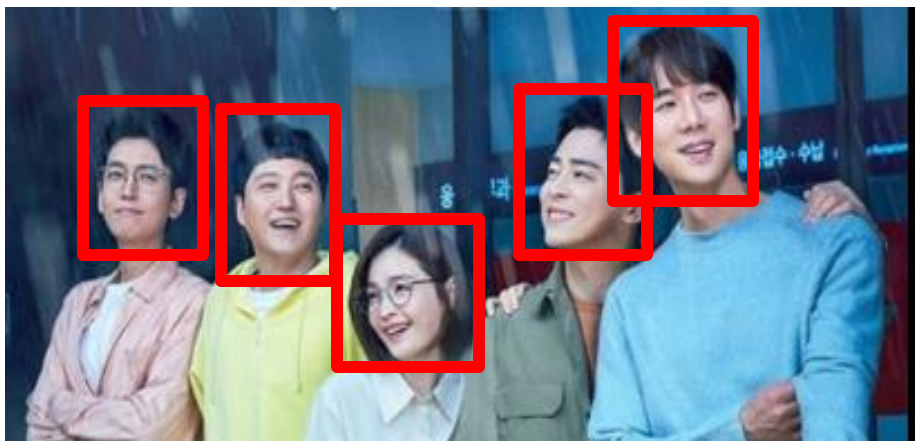


사진 분류 과정

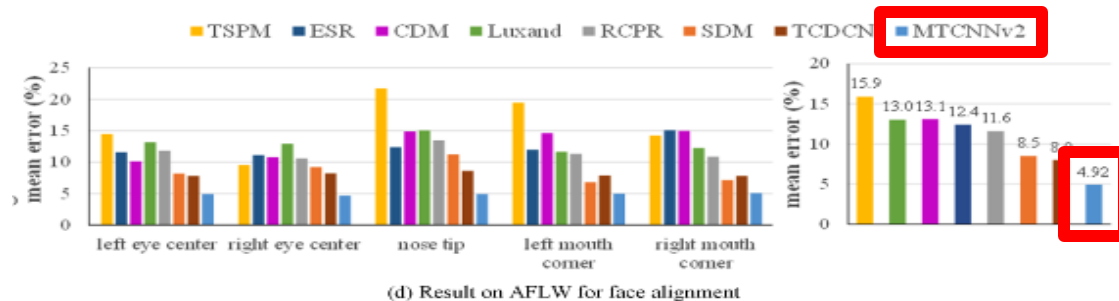
Detection – MTCNN

Face Detection 모델로 얼굴의 영역을 Detection합니다.
Face Detection 모델은 MTCNN 모델을 사용했습니다.



MTCNN 선정 이유

높은 정확도



빠른 inference-time

- Pictures containing 10 frontal faces:

Image size	Total pixels	Process time	FPS
474x224	106,176	0.185 seconds	5.4
736x348	256,128	0.290 seconds	3.4
2100x994	2,087,400	1.286 seconds	0.7

사진 분류 과정

Detection – MTCNN

Face Detection 모델으로 얼굴의 영역을 Detection합니다.
Face Detection 모델은 MTCNN 모델을 사용했습니다.



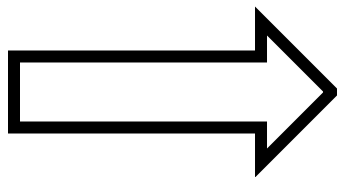
사진 분류 과정

Feature Extraction

Face Detection 모델을 통해 얻은 **얼굴 이미지**의 **임베딩 벡터**를 추출합니다.

VggFace2 데이터셋으로 Pre-train 된 InceptionResnet-50 모델을 Fine-Tuning 하여 사용했습니다.

얼굴 이미지



임베딩 벡터 추출

[122.28, 213.07, 16.45, 35.83, ... 124],
[72.86, 21.3, 55.16, 124.35, ... 121.11],
[7.78, 32.11, 16.55, 174.22, ... 187.21],
[66.322, 83.21, 6.12, 35.34, ... 44.132],
[193.11, 213.5, 6.14, 52.23, ... 133.23]

사진 분류 과정

Verify

추출된 임베딩 벡터를 분류를 원하는 인물의 임베딩 벡터와 **코사인유사도**를 계산하여 **같은 인물인지** 판단합니다.



Is in



문제 정의

VggFace2 Dataset



Source domain

Korean Celeb Dataset



Target domain



문제 정의

사전 조사 결과 얼굴 인식에 East Asian이 상대적으로, 낮은 정확도를 가짐

Backbone	Dataset	MR-ALL	African	Caucasian	South Asian	East Asian	Link(onnx)
R100	Casia	42.735	39.666	53.933	47.807	21.572	GDrive
R100	MS1MV2	80.725	79.117	87.176	85.501	55.807	GDrive
R18	MS1MV3	68.326	62.613	75.125	70.213	43.859	GDrive
R34	MS1MV3	77.365	71.644	83.291	80.084	53.712	GDrive
R50	MS1MV3	80.533	75.488	86.115	84.305	57.352	GDrive
R100	MS1MV3	84.312	81.083	89.040	88.082	62.193	GDrive
R18	Glnt360K	72.074	68.230	80.575	75.852	47.831	GDrive
R34	Glnt360K	83.015	79.907	88.620	86.815	60.604	GDrive
R50	Glnt360K	87.077	85.272	91.617	90.541	66.813	GDrive
R100	Glnt360K	90.659	89.488	94.285	93.434	72.528	GDrive

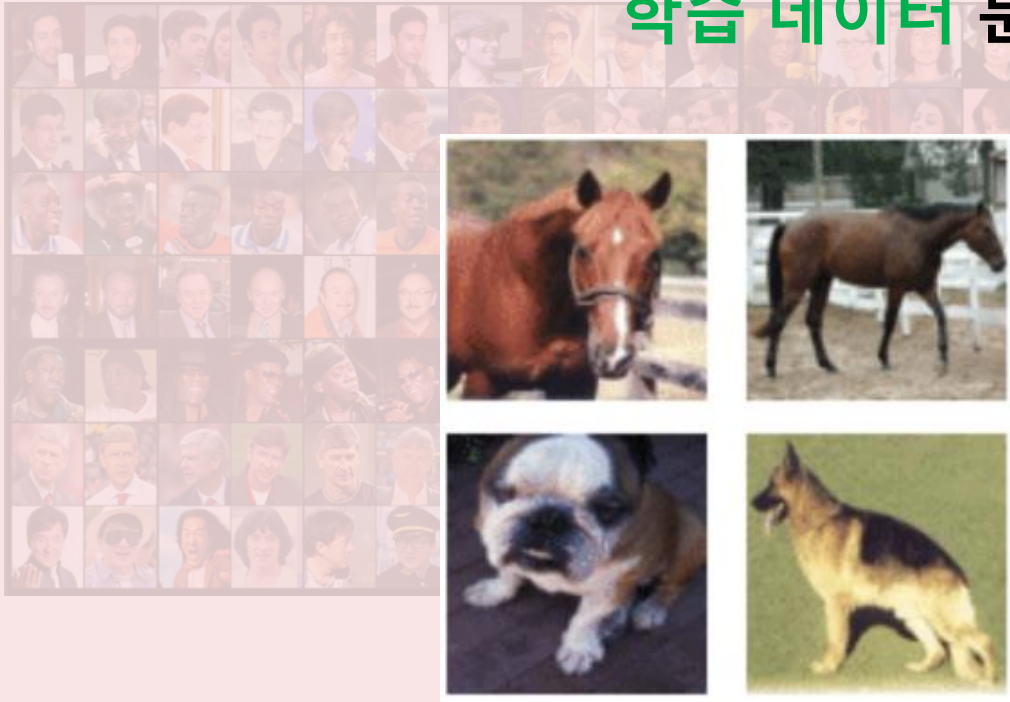
문제 정의

Domain Shift

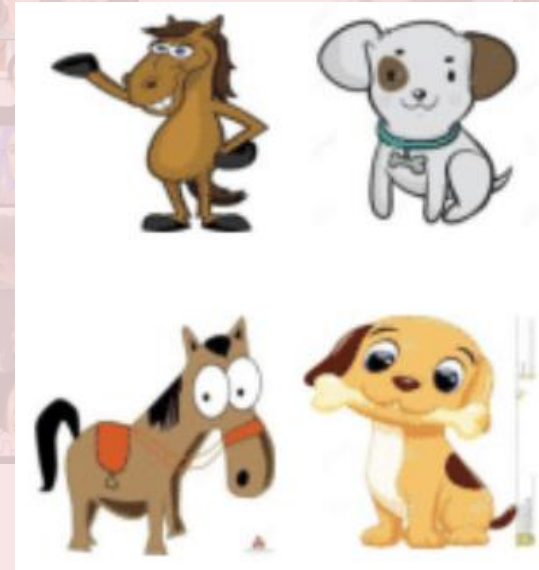
VggFace2 Dataset

학습 데이터 분포와 테스트 데이터 분포가 다름

한국의 Celeb Dataset

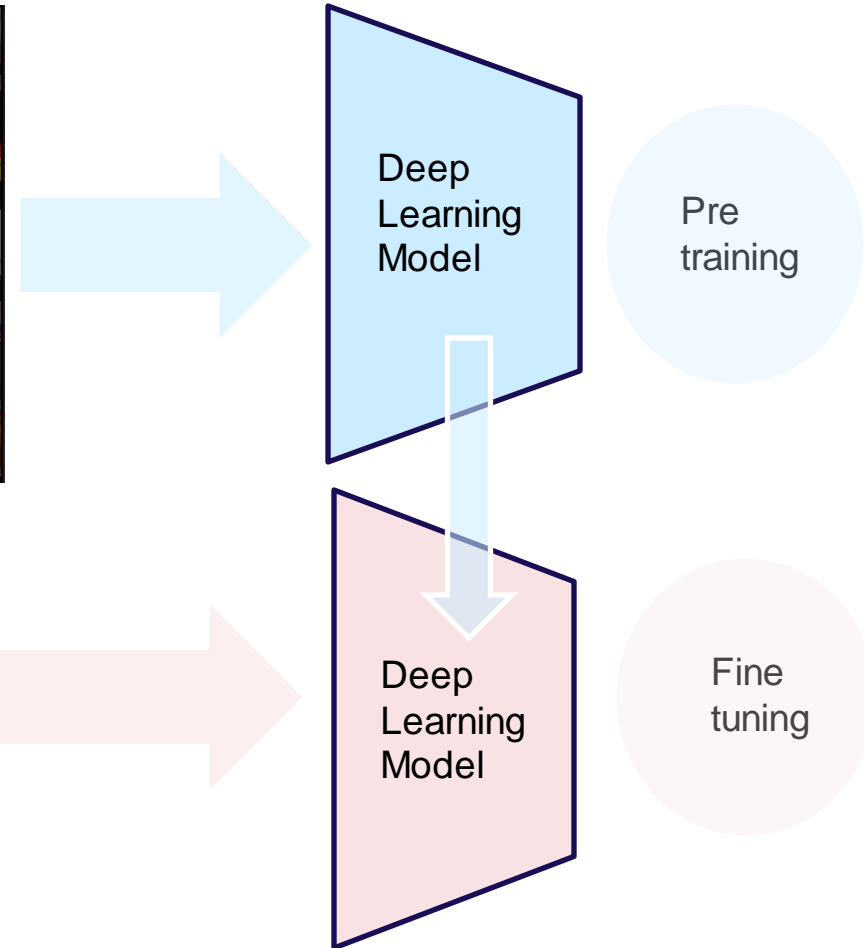


Training set



Test set

Domain Generalization



PROJECT TIMELINE



방향성

DATA

Domain 적합
데이터를 crawling &
labeling

모델 탐색

관련 SOTA모델 서칭
모델 특징 분석

Metric

Domain일반화 & 성능↑
적합한 Metric 선정

Train

적합한 데이터로 Fine-tuning
Hyper Parameter-tuning

2.Dataset & Metric

1. Dataset
2. Loss & Distance
3. Metric

한국인 Celeb Data Crawling

- 100명의 label
- 각 20장씩 총 2000장

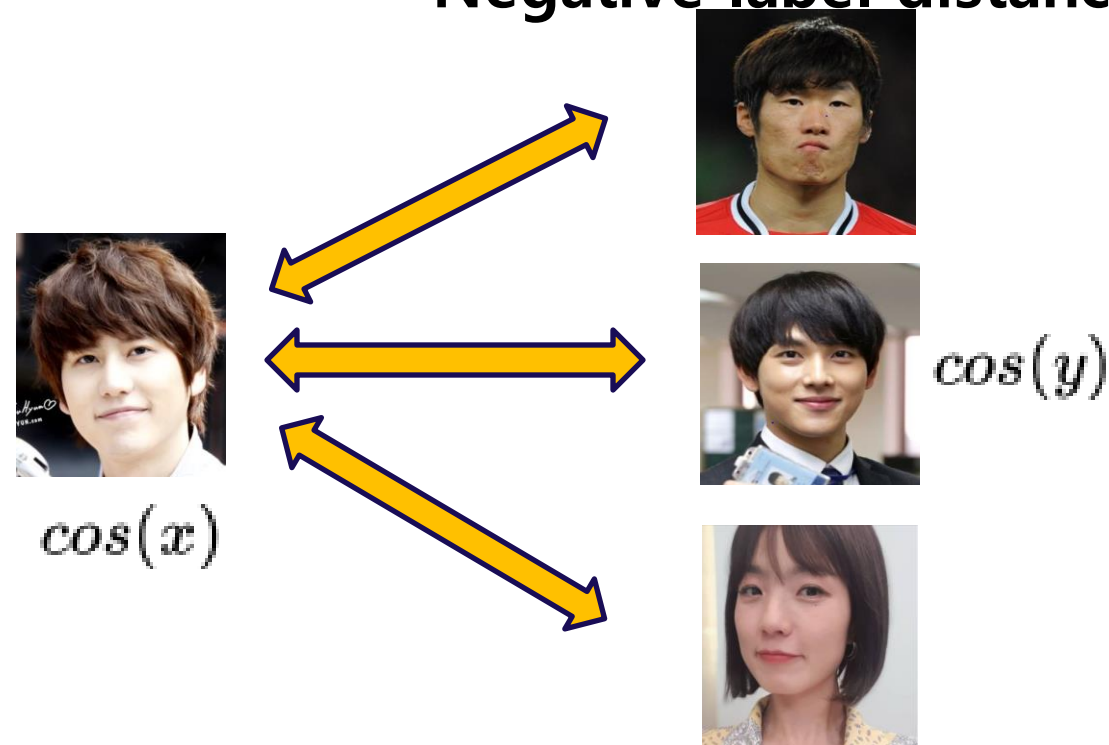
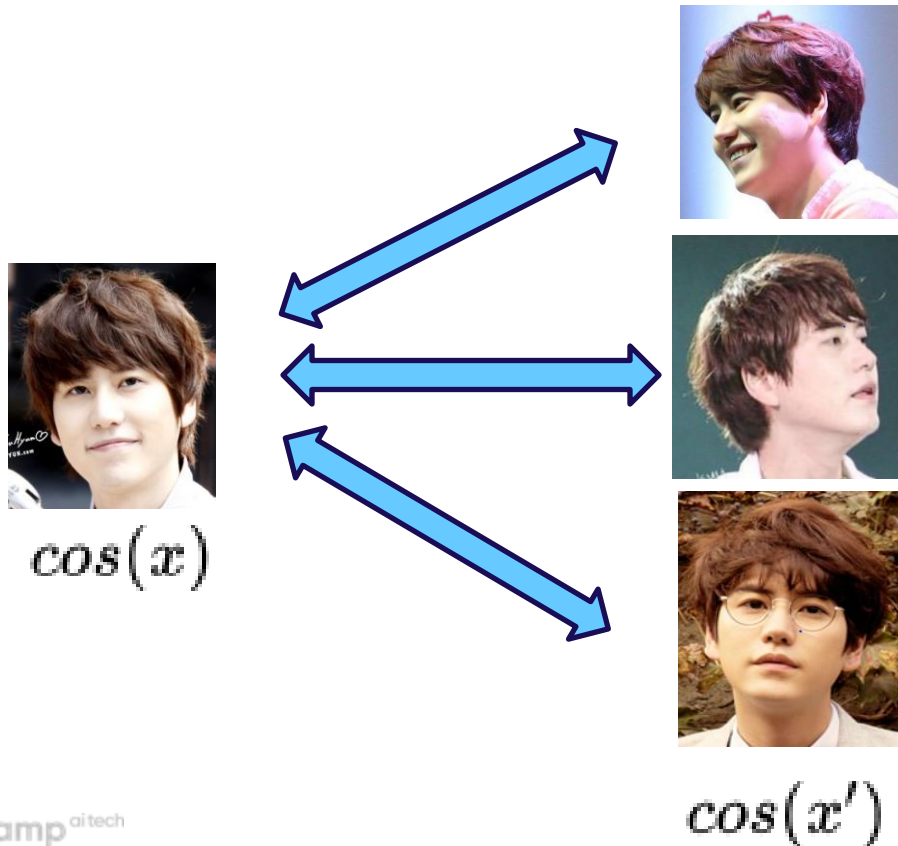


Distance (Cosine Similarity)



Positive label distance

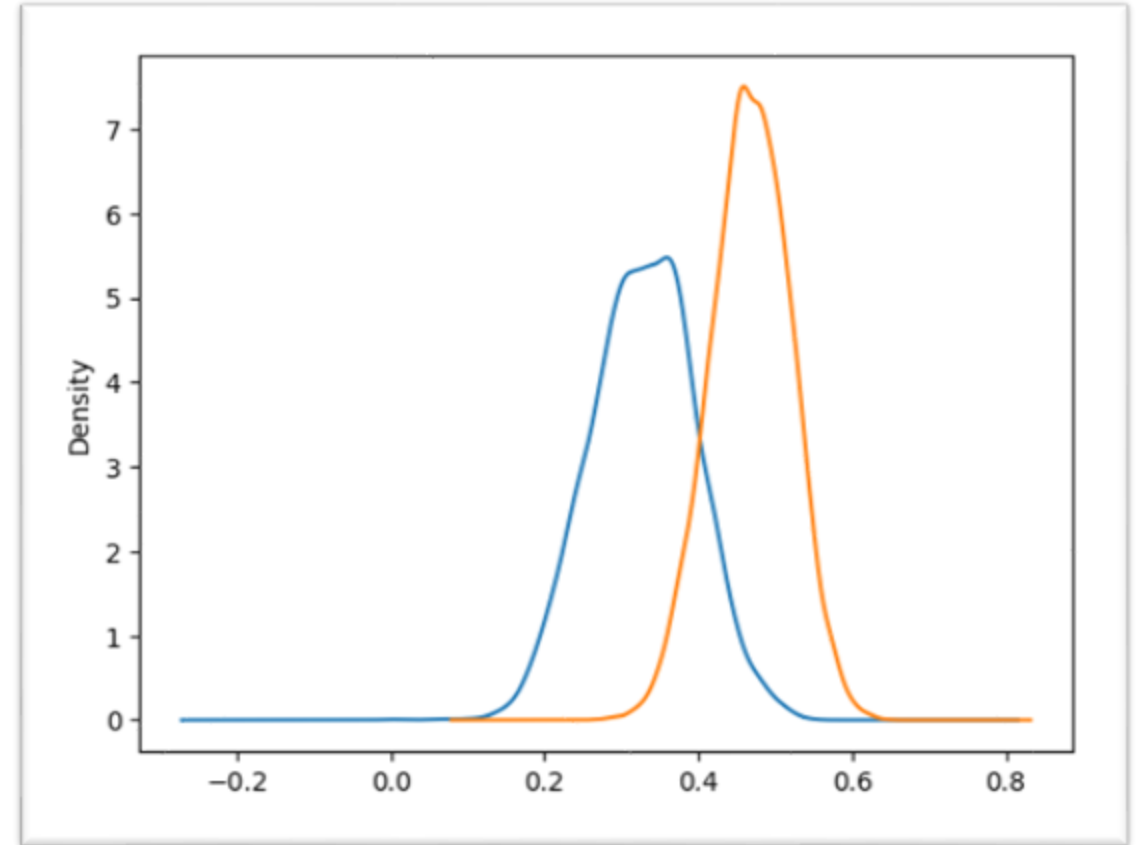
$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|}$$

Negative label distance



Metric

- KDE(Kernel Density Estimation)
- Positive distance 
- Negative distance 
- Threshold : acc 최고점으로 설정



3. 사용 모델 선정 및 fine-tuning

1. pre-trained Model
2. Fine-tuning
3. Result

Pre-trained Model

Arcface(Resnet 18) - MS1MV3	
Acc	0.5485
Recall	0.6102
F1	0.4579
Precision	0.3664

Arcface(mobilenet) - face emore	
Acc	0.5321
Recall	0.5906
F1	0.4410
Precision	0.3519

Facenet(Inception) - VGGface2	
Acc	0.8810
Recall	0.8382
F1	0.8262
Precision	0.8096

- 모델 선정 Test dataset :
- korean celeb 500장
- Acc 기준으로 pretrained model 선택
- 최종 선택한 모델
- **Facenet**(InceptionResnetV1)

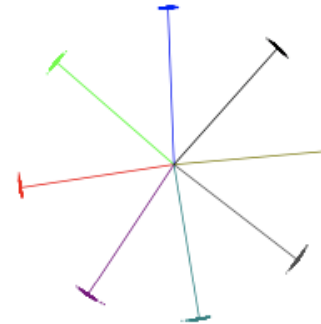
Arcface(Additional Angular margin loss)

$$Loss = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{s(\cos(\theta_{y_i} + m))}}{e^{s(\cos(\theta_{y_i} + m))} + \sum_{j=1, j \neq y_i}^n e^{s \cos \theta_j}}$$

softmax



arcface



Fine-tuning Result


- Validation data (500장) // Train data (1500장)

Pre-trained-model	
Acc	0.8810
Recall	0.8382
F1	0.8262
Precision	0.8096



Fine-tuned model	
Acc	0.9292
Recall	0.8739
F1	0.8911
Precision	0.9090

FINE TUNING

LEARNING RATE		OPTIMIZER		LAYER FREEZE		RESIZE		FINAL			
lr	Acc	Optimizer	Acc	freeze	Acc	Resize	Acc	Fine-tuned model			
1e-4	0.8945	SGD	0.8943	No-freeze	0.8945	112	0.8943	Acc	0.9292		
5e-5	0.8815	Adam	0.8234	Bn	0.8942	160	0.9292	Recall	0.8739		
1e-5	0.6652			conv	0.8628			F1	0.8911		
5e-6	0.5662							Precision	0.9090		
1e-6	0.4531										
1e - 4		SGD		Non-Freeze layer		Resize : 160					

3. Product Serving

1. Service Architecture

Service 포인트



Microservices

마이크로 서비스 아키텍처 패턴

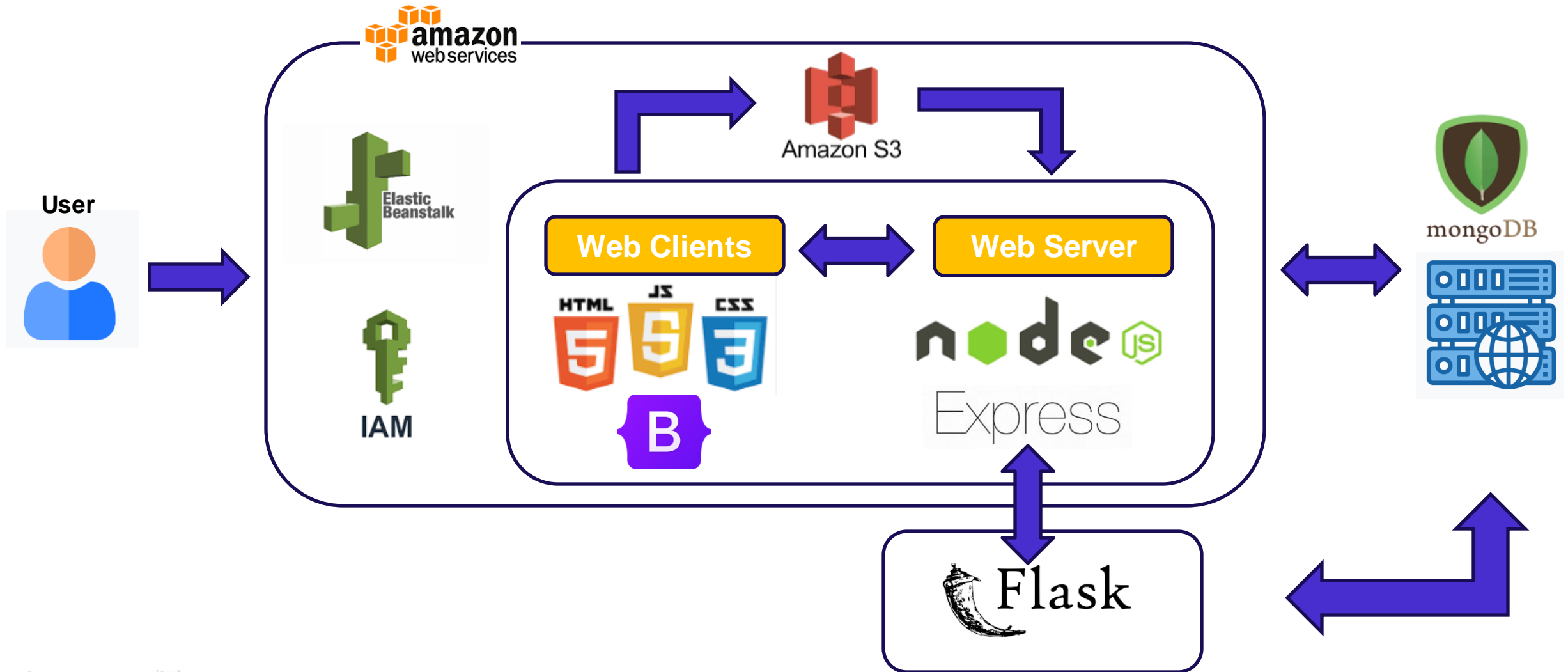
Node.js : 사용자와 상호작용

Flask : 모델 탑재, 이미지 검색

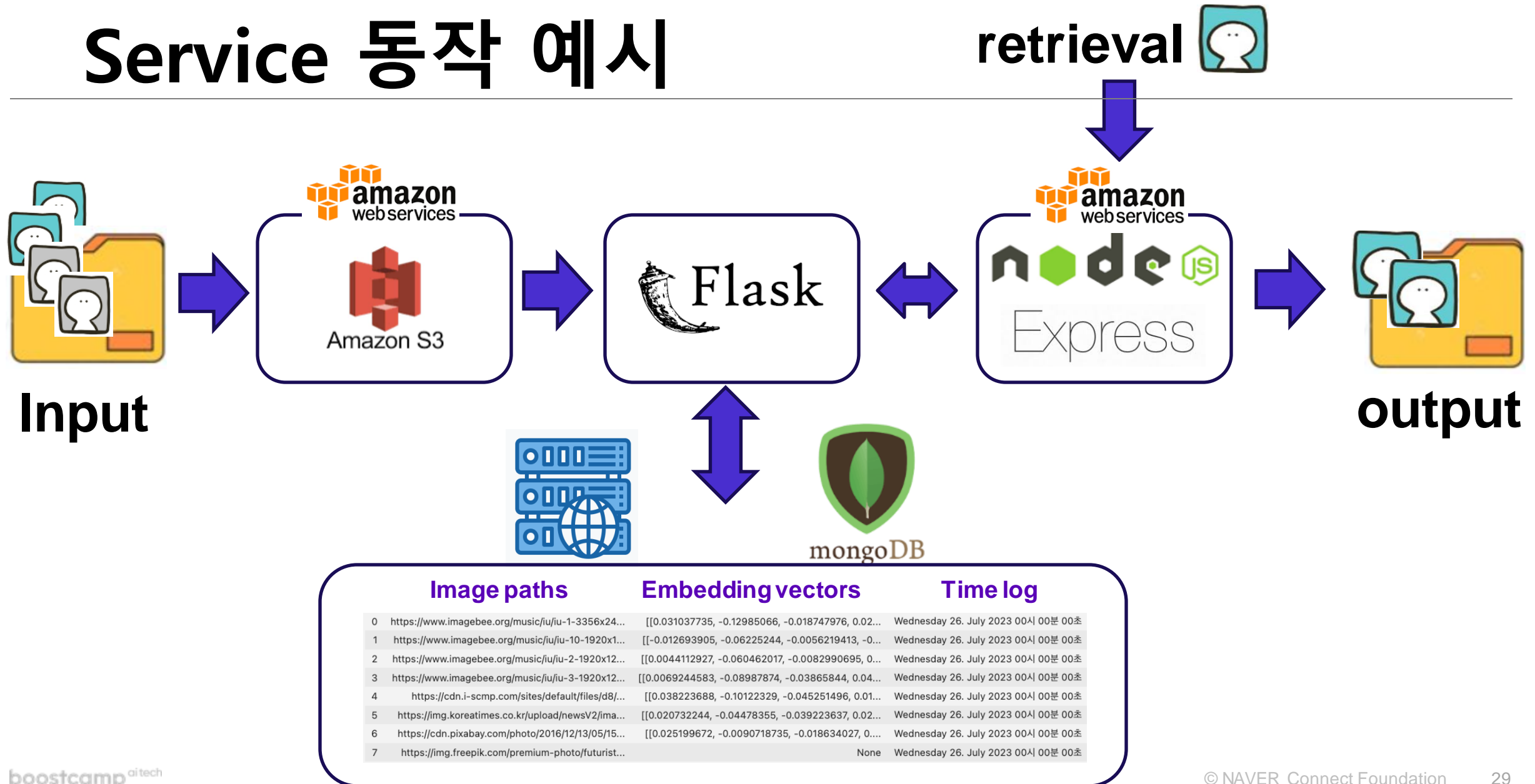


이미지 업로드 시
Embedding vector 계산 후,
DB에 저장

Service Architecture



Service 동작 예시



4. 시연 영상

1. Result/ Conclusion

Result & Conclusion

모델링

- ☒ 88%성능의 모델을 Fine-tuning 하여 93%의 성능 달성. (목표 : 95%)
- ☒ 다른 여러 pretrained 모델 비교 및 채
- ☐ 모델 경량화

데이터

- ☒ 적절한 데이터 셋을 수집하였다.
- ☐ 데이터의 수의 부족을 Domain Adaptation의 DANN 방법으로 해결

Front End & Back End

- ☒ Node.js, mongoDB를 사용해 구
- ☒ 적절한 서비스 평가 메트릭을 고안
- ☒ 마이크로 서비스 아키텍처를 통한 서비스 배포
- ☐ 휴대폰으로 활용이 예상되는 서비스라 어플 개발이 필요

후속 개발

지속적인 재학습

- ☐ 고객 앨범 데이터셋 확보
고객의 동의를 받은 데이터를 학습에 활용
- ☐ 주기적 모델 재학습을 통한 일반화 성능 향상
ex) 고객 평가의 잃은 별점 n개 누적시마다 재학습
- ☐ 모델 경량화(pruning, quantization)를 통한 빠른 추론

클라우드 연동 & 배포

- ☐ 클라우드를 연동하여 고객이 직접 사진을 업로드 하는 불편을 제거
- ☐ docker를 활용한 환경관리 및 배포

기능 확장

- ☐ 인물 + 사물, 동물 등 검색 기능 확장
- ☐ Chat-GPT API 연결하여 추천 분류 태그 얻기
- ☐ Fine-tuning model 활용하여 추가 기능 확장
(주민등록증 사진 신뢰도, 보정 정도 확인하는 서비스)

어플리케이션 서비스 개발

- ☐ 웹 서비스 → 어플리케이션 서비스 전환을 통한 사용자 편의성 향상
- ☐ IOS, Android 어플 개발

End of Document

Thank You.