

Facelog

얼굴 감정 분류, AI SNS 감정 일기 서비스

3조

김성혜, 박주성, 이구협, 이승후

⌚ https://github.com/zooodung/Face_Image_Emotion_Classification/

Facelog

목차

01 프로젝트 개요

1. 프로젝트 배경
2. 프로젝트 목표
3. 프로젝트 적용기술

02 EDA 및 전처리

1. EDA
2. 전처리

03 모델 및 API

1. 모델 학습
2. 모델 비교
3. 모델 선정
4. API

04 서비스

1. 서비스 소개
2. 시스템 구성
3. 개발 환경
4. 기능
 - 4.1 핵심 기능
 - 4.2 부가 기능

05 서비스 구현

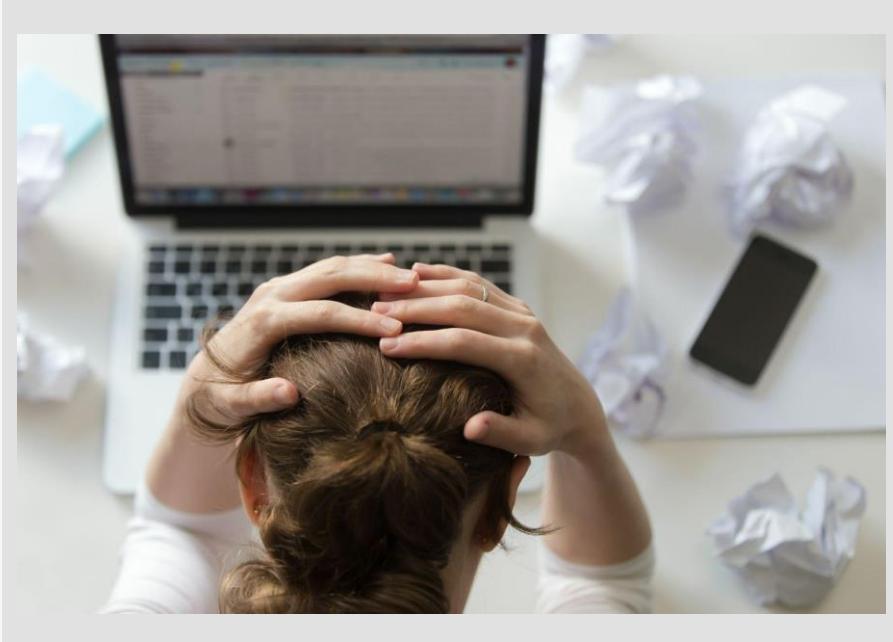
1. 화면 구성
2. 동작 로직
3. 시연 동영상
4. 시연

06 마무리

1. 문제점 및 개선방안
2. 제안사항
3. 참고문헌

프로젝트 개요

프로젝트 배경



01 현대인의 우울증 심화

- 우울증을 유발하는 현대사회
- 운동, 복용제, 광선치료, 명상 등으로 치료
- 외향적 활동보다 내향적 자기 성찰 필요

02 SNS와 소셜 블랙아웃

- SNS상의 인간관계에 대한 회의감 증대
- 지나친 SNS 관리로 인한 피로감 극대화
- 보여주기식 SNS가 아닌 자연스러운 공간 필요

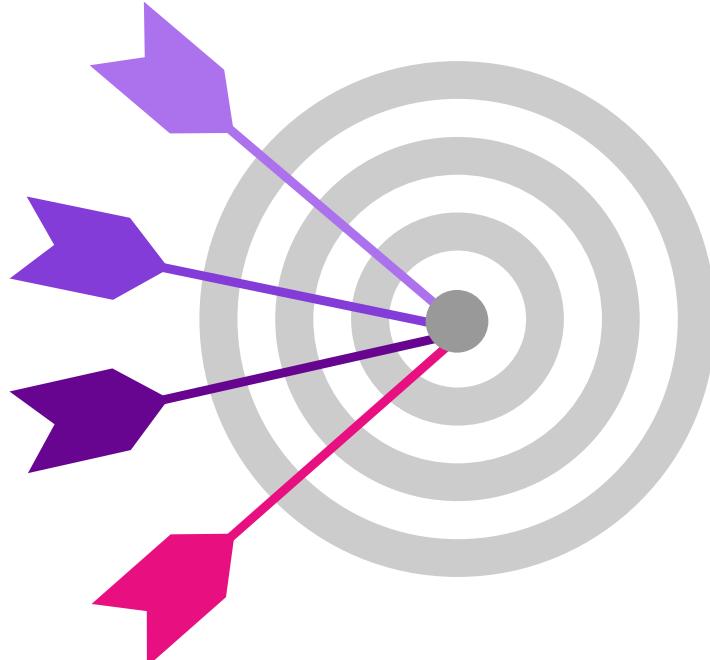
프로젝트 목표

AI를 통한 분석

AI를 이용한 대화

감정일기 제공

심리적 안정



AI 모델을 통해 감정 기록

얼굴 사진과 말, 또는 글을 통해 사용자의 기분을 정확히 파악

LM을 통한 대화 제공

언어 모델을 이용하여 사용자에게 적절한 문장을 제시

날짜별 감정일기 기록

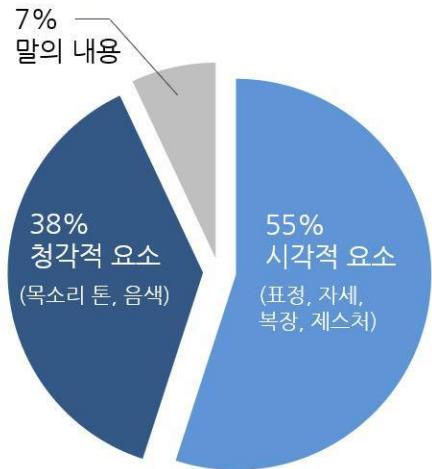
기록된 감정을 날짜별로 정리하여 보기 좋게 표시

대화를 통한 심리적 안정

언어 모델을 통해 실제와 같은 대화를 제공하여 심리적 안정 제공

프로젝트 목표

메라비언의 법칙
The Law of Mehrabian



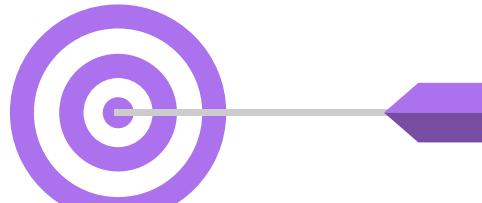
〈앨버트 메라비언〉

감정에서, 시각적 요소는 **매우 중요!**

따라서, 안면 감정 인식을 통해 말의 내용 뿐만이 아니라 사용자의 감정을 정확하게 파악하고 기록할 수 있음.

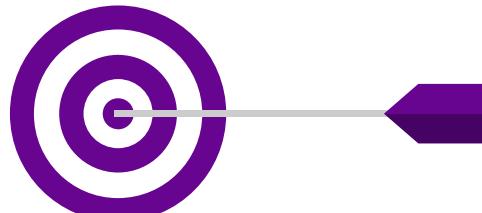
프로젝트 적용기술

EDA



확보한 데이터를 정제하여 더 높은 정확도를 확보

이미지 분류



VGG19, ResNet, SVM, ViT 등 다양한 ML/DL 모델을 이용하여 감정분류

LM/TTS/STT



언어 모델과 TTS/STT를 활용하여 이질감을 감소시키고 실제 대화처럼 느껴지게 서비스 제공

EDA 및 전처리

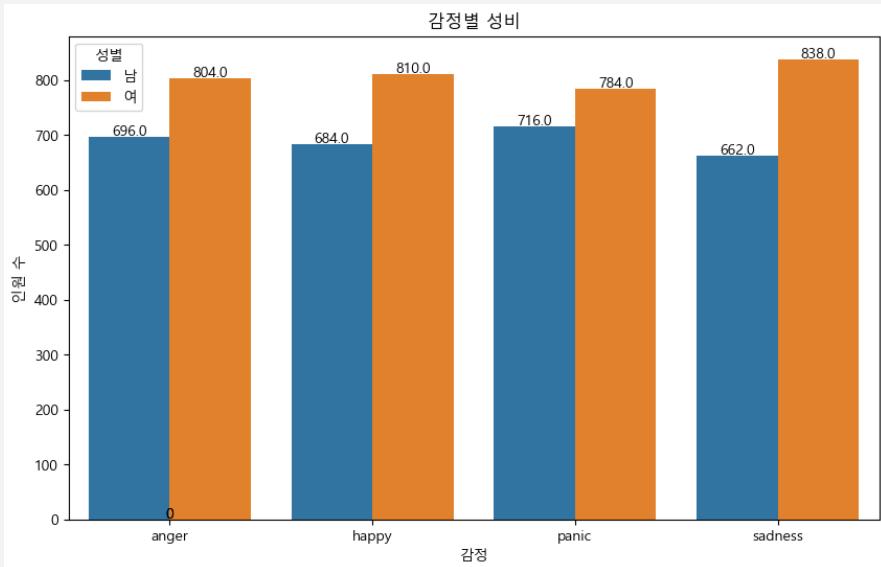
EDA

□ Label Data Feature

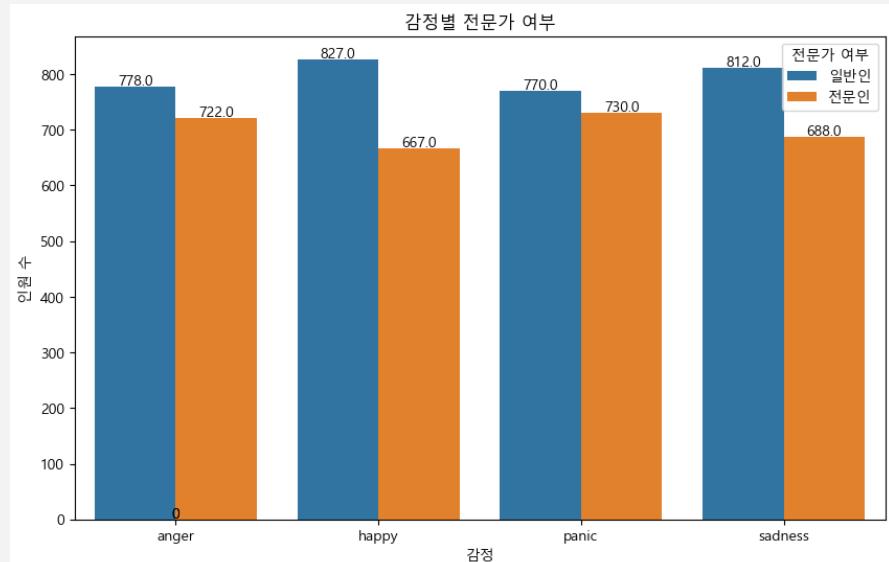
```
{  
    "filename": "6oj439e3fbcc52759fb3093035b7c0ecc55c93543dae63ec1162f5f6bfc6e4811.jpg",  
    "gender": "남",  
    "age": 30,  
    "isProf": "일반인",  
    "faceExp_uploader": "분노",  
    "bg_uploader": "교통/이동수단(엘리베이터 포함)",  
    "annot_A.boxes maxX": 2073.254604999996,  
    "annot_A.boxes maxY": 1486.204442,  
    "annot_A.boxes minX": 1338.220104999999,  
    "annot_A.boxes minY": 539.506641999999,  
    "annot_A.faceExp": "분노",  
    "annot_A.bg": "교통/이동수단(엘리베이터 포함)",  
    "annot_B.boxes maxX": 2074.663604999997,  
    "annot_B.boxes maxY": 1490.044442,  
    "annot_B.boxes minX": 1341.270105,  
    "annot_B.boxes minY": 528.132642,  
    "annot_B.faceExp": "분노",  
    "annot_B.bg": "교통/이동수단(엘리베이터 포함)",  
    "annot_C.boxes maxX": 2072.188605,  
    "annot_C.boxes maxY": 1487.918442,  
    "annot_C.boxes minX": 1340.085104999999,  
    "annot_C.boxes minY": 525.650642,  
    "annot_C.faceExp": "분노",  
    "annot_C.bg": "교통/이동수단(엘리베이터 포함)",  
    "emotion": "anger"  
},
```

- 'filename' - 이미지 파일 이름
- 'gender' - 성별
- 'age' - 나이
- 'isProf' - 신분
- 'faceExp_uploader' - 감정
- 'bg_uploader' - 배경
- 'annot_A', 'annot_B', 'annot_C' - 3명의 어노테이터가 얼굴의 X,Y 좌표/감정/배경을 특정

EDA



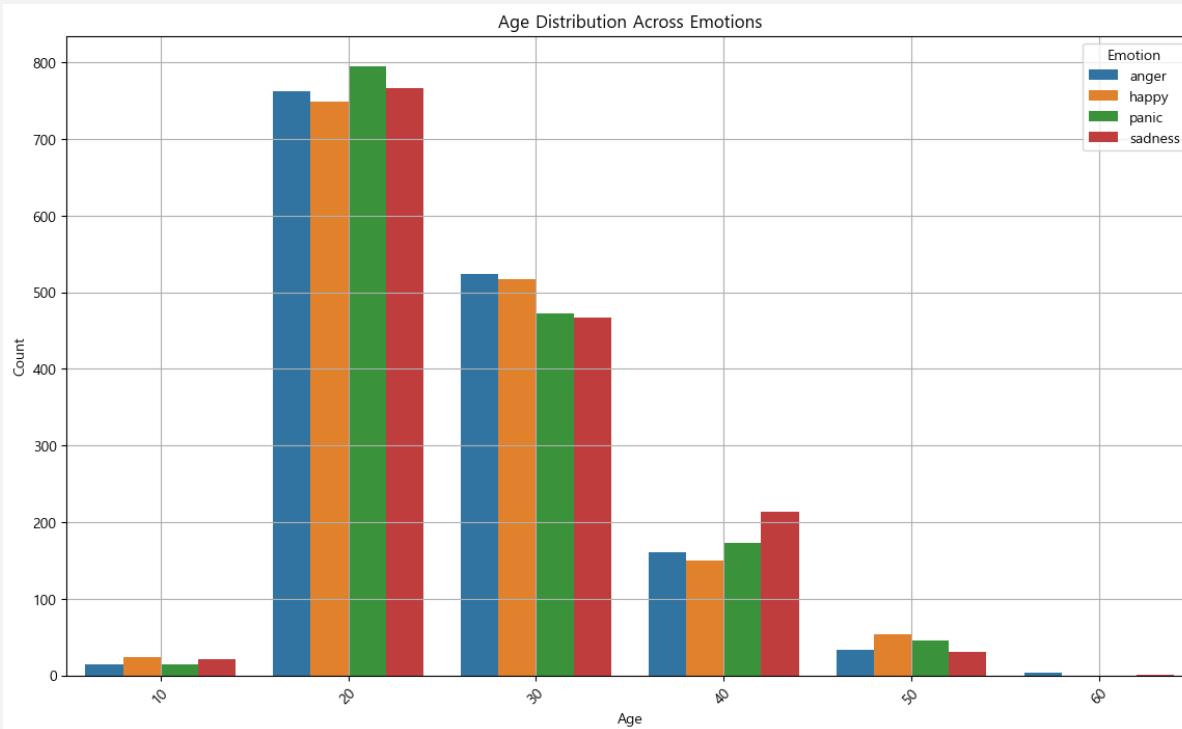
Emotion별 남/여 비율



Emotion별 전문가/일반인 비율

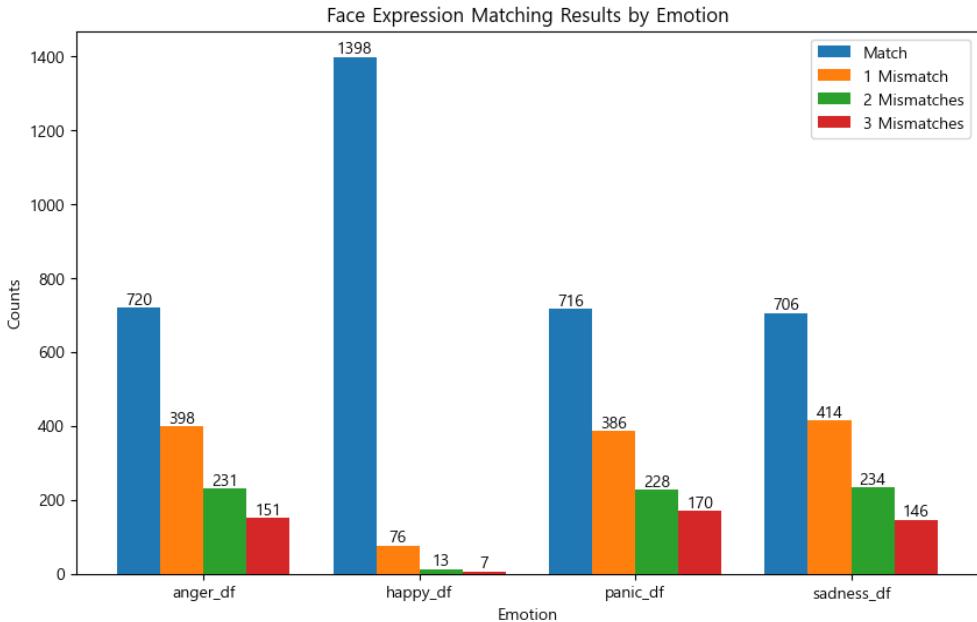
EDA

□ EMOTION 별 Age 세대 분포



EDA

□ Label Data 검증



- 3 Mismatch, 2 Mismatch에 해당하는 Data를 Outliers로 판단
- 2개 이상 동일한 FaceExp로 분류된 이미지를 학습 데이터로 선정

anger : 1118
happy : 1474
panic : 1102
sadness : 1120

각 Emotion의 상위 1102개 데이터 활용

전처리

With Label, Segment Data

Train Data / Validation Data

- FACE 이외 요소 Masking
- Label Data 활용 Face Crop
- MTCNN 활용 특징점 추출
- 좌/우 눈 기준 수평 회전
- 눈 사이 거리 스케일링
- 코끝 기준 이미지 중심이동
- 224x224 Resizing
- 0부터 1사이로 정규화

Without Label, Segment Data

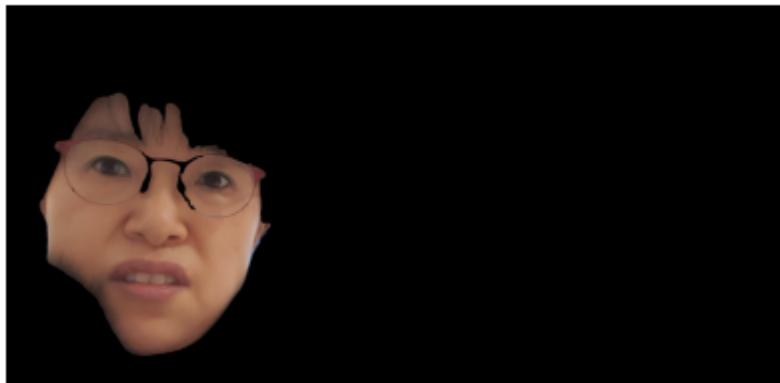
TEST Data / Web Input Data

- Dlib 활용 Face Crop
- MTCNN 활용 특징점 추출
- 좌/우 눈 기준 수평 회전
- 눈 사이 거리 스케일링
- 코끝 기준 이미지 중심이동
- 224x224 Resizing
- 0부터 1사이로 정규화

전처리

With Label, Segment Data
1. Masking 2.Crop

mask



crop



- segmentation 정보를 활용하여 얼굴 부분을 제외한 요소 masking 처리
- 얼굴 인식 정확도 향상
- 배경 및 다른 물체가 얼굴 인식 및 학습에 영향을 끼치는 것 방지

- `annot_A.boxes`
`annot_B.boxes`
`annot_C.boxes`
- 어노테이터의 얼굴 라벨링 정보를 활용해 이미지 크롭
- 얼굴 인식 속도 향상

전처리

Without Label, Segment Data
1.Crop

origin



→ 원본 이미지
(슈퍼주니어 신동)



crop



- Dlib 라이브러리 활용
- 얼굴의 68가지 특징점 추출
- 특징점 기반 얼굴 영역 Padding 적용
- Crop

전처리 MTCNN 활용 특징점 추출

Keypoints



- MTCNN 라이브러리 활용
- 얼굴의 5가지 특징점 추출
- 얼굴 인식 정확도 향상

Keypoints



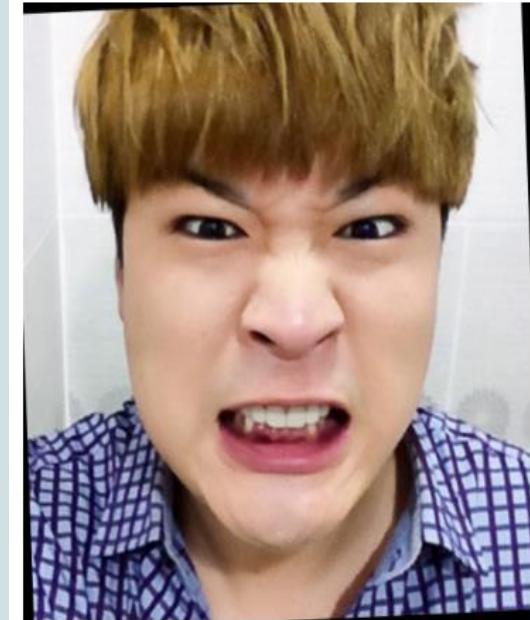
전처리 좌/우 눈 기준 수평 회전

rotate



- 왼쪽 눈, 오른쪽 눈 정보를 활용해 얼굴의 기울기 계산
- 눈을 기준으로 하여 수평으로 이미지 회전
- 얼굴 인식 정확도 향상

rotate



전처리

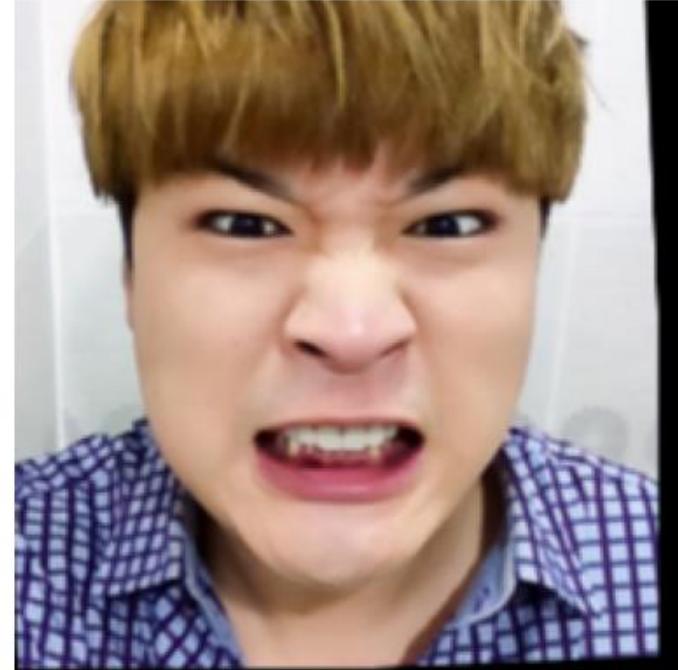
눈 사이 거리 스케일링
코 끝 기준 이미지 중심 이동
224x224 Resizing

normalize



- Dlib 라이브러리 활용
- 얼굴 68가지 특징점 추출 후 이를 기준으로 이미지 스케일링, 이동
- 눈 사이의 거리 정규화
- 코 끝을 중심으로 이미지 이동
- 이미지 크기 조절(224x224)
- 얼굴 크기나 모양에 관계 없이 일관된 특징 추출

normalize



모델 및 API

모델 학습

사용한 모델

비 Transformer 기반	Transformer 기반
<ul style="list-style-type: none">• VGG 19• ResNet• EmotionNet• SVM• EfficientNet	<ul style="list-style-type: none">• ViT• DeiT

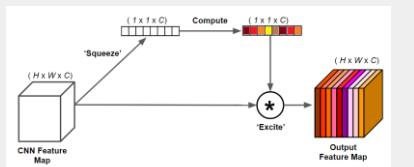
모델 학습

□ EfficientNet

주요 특징

Compound Scaling
MBConv
Swish Activation Function
Squeeze-and-Excitation

Squeeze-and-Excitation Optimization

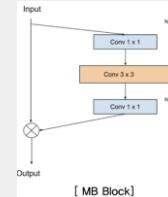


Compound Scaling

$$\begin{aligned} \text{depth: } d &= \alpha^\phi \\ \text{width: } w &= \beta^\phi \\ \text{resolution: } r &= \gamma^\phi \\ \text{s.t. } \alpha \cdot \beta^2 \cdot \gamma^2 &\approx 2 \\ \alpha \geq 1, \beta \geq 1, \gamma \geq 1 & \end{aligned}$$

MBConv

Mobile Inverted
Bottleneck
Convolution



Swish Activation Function

Swish

$$f(x) = x * \text{sigmoid}(x)$$

모델 학습

Compound Scaling

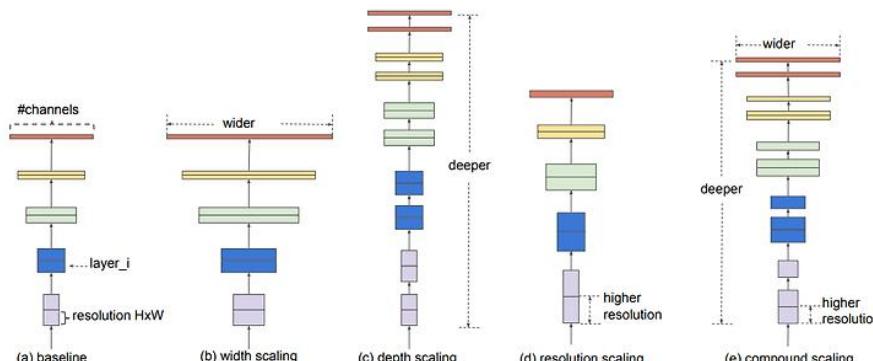


Figure 2. Model Scaling. (a) is a baseline network example; (b)-(d) are conventional scaling that only increases one dimension of network width, depth, or resolution. (e) is our proposed compound scaling method that uniformly scales all three dimensions with a fixed ratio.

Depth : Layer의 깊이

Width: Layer의 채널의 수. 네트워크 용량

Resolution: 입력 이미지의 해상도. 정보량

Compound Scaling은 이 모든걸 균형있게 확장함

$$\text{Depth } d = \alpha^\phi, \text{Width } w = \beta^\phi, \text{Resolution } r = \gamma^\phi, (1)$$

$$\text{such that } \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$$

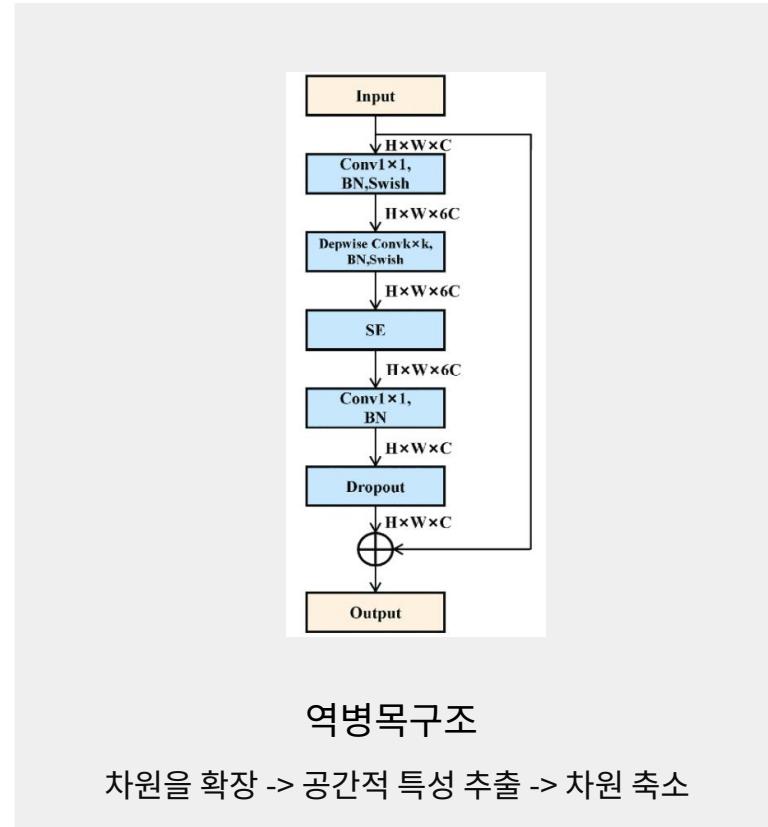
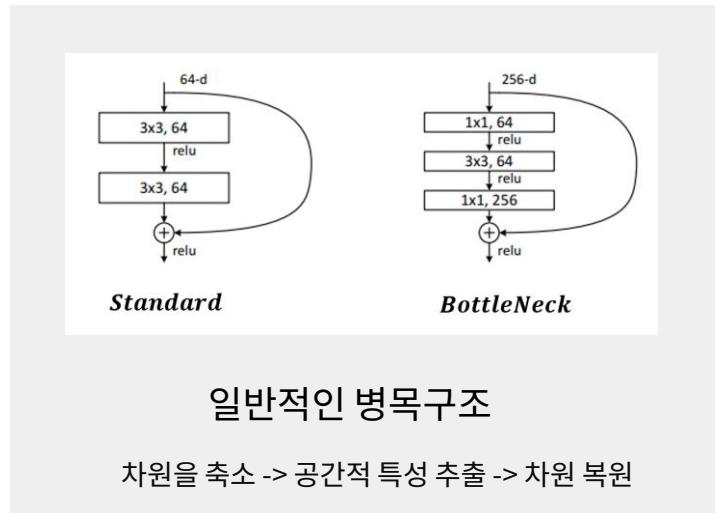
$$\alpha \geq 1, \beta \geq 1, \gamma \geq 1$$

α, β, γ 는 각각 깊이, 너비, 해상도 확장 계수이며,
 ϕ 는 전체 확장 인자

EfficientNet은 해당 제약조건을 사용함

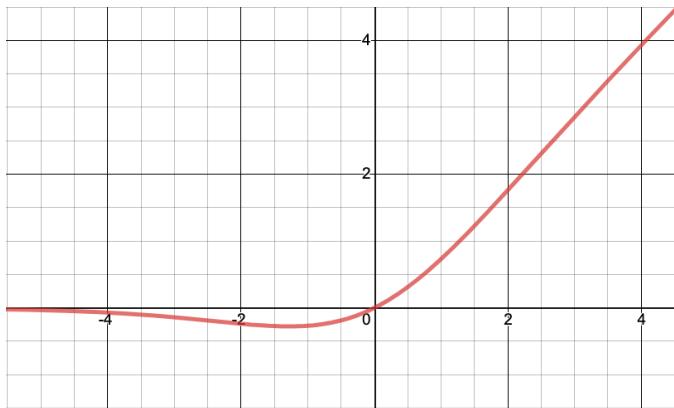
모델 학습

□ MBConv.



모델 학습

□ Swish 활성화 함수

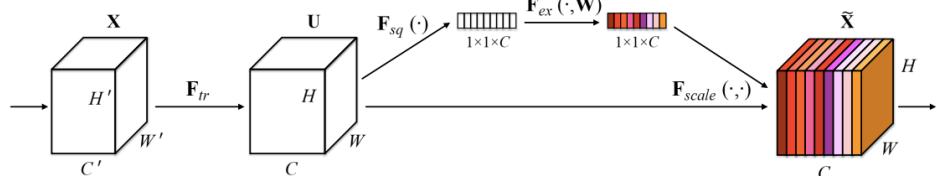


$$f(x) = x * \text{sigmoid}(x)$$

- 입력값이 연속적일때 부드럽게 변함
- ReLU와 달리 음수일때도 값을 출력
- 입력값에 비례하여 출력값을 조정 (자기 정규화)

모델 학습

□ Squeeze-and-Excitation



1. Squeeze 단계

Feature Map의 각 채널에 대해 Global Average Pooling을 수행하여 공간적 정보를 압축하여 각 채널의 Global Descriptor를 얻음

2. Excitation 단계

Global Descriptor를 두개의 Fully Connected Layer를 통해 중요도를 학습. 첫번째 레이어에서는 차원을 축소하여 비선형성을 적용한후 두번째 레이어에서 원래차원으로 확장

3. Scaling 단계

Excitation단계에서 얻은 중요도 벡터를 입력 Feature Map의 각 채널에 곱하여 더 중요한 채널을 강조하고 덜 중요한 채널을 억제합니다.

모델 학습

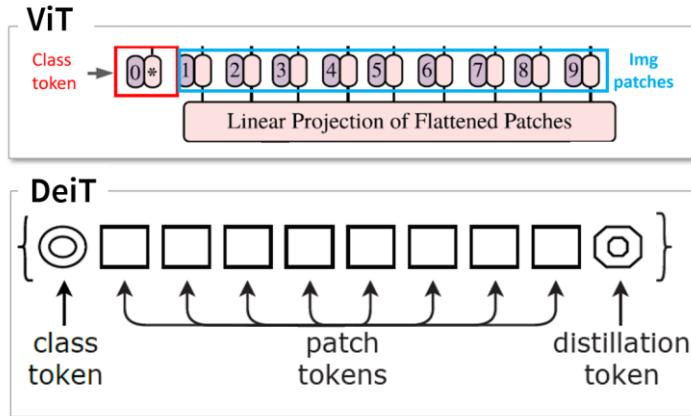
□ ViT

일반 Transformer와 ViT의 차이

공통점	차이점
<ul style="list-style-type: none">• Self-Attention 매커니즘• 포지셔널 인코딩• 임베딩 벡터로 변환	<ul style="list-style-type: none">• Transformer는 단어를 Token으로 나누는 반면에, ViT는 이미지를 Patch 단위로 나눔• Transformer는 입력 시퀀스에서 모든 토큰이 사용되는 반면에, ViT는 특별한 클래스 토큰을 추가함

모델 학습

□ Data-efficient Image Transformers



Distillation Token

교사 모델이 존재하고, 교사모델은 학생 모델에게 지식을 전이함

DeiT의 교사모델은 CNN 아키텍쳐를 가진 RegNet이며
DeiT는 더 작은 데이터셋에서 교사 모델의 출력을 학습하여
더 작은 데이터셋에서도 우수한 성능을 발휘함

지식 증류과정에서 Class Token은 True Label을 맞추게 되지만, Distillation Token은 교사 모델의 추론값과의 오차를 줄여나가며 모델의 지식을 학습함.

모델 학습

Train Set이 상당히 많지 않기 때문에, 데이터 증강을 진행

증강 사례 1

```
train_datagen = ImageDataGenerator(  
    rescale=1./255,  
    rotation_range=20,  
    width_shift_range=0.2,  
    height_shift_range=0.2,  
    shear_range=0.2,  
    zoom_range=0.2,  
    horizontal_flip=True,  
    fill_mode='nearest'  
)  
  
val_datagen = ImageDataGenerator(rescale=1./255)
```

모델 학습

증강 사례 2

```
train_datagen = ImageDataGenerator(  
    rescale=1./255,  
    rotation_range=20,  
    width_shift_range=0.2,  
    height_shift_range=0.2,  
    horizontal_flip=True,  
    brightness_range=[0.8, 1.2],  
    zoom_range=0.2,  
    preprocessing_function=lambda img: augment_face_pose(img, augmenter),  
)  
  
val_datagen = ImageDataGenerator(rescale=1./255)
```

모델 비교

Test Data의 Label Data로 검증했을때,
Test Data Set에도 일치하지 않는 결과가 존재

따라서 이들의 비율을 학습 최대치 목표로 설정

목표 Accuracy

anger : 73.21%
happy : 97.65%
panic : 82.90%
sadness : 74.64%

Counts for emotion '분노'
Zero annotation same : 28
One annotation same : 47
Two annotations same : 71
Three annotations same : 134
Sum of two, three : 205
Sum of all : 280

Counts for emotion '기쁨'
Zero annotation same : 2
One annotation same : 5
Two annotations same : 12
Three annotations same : 279
Sum of two, three : 291
Sum of all : 298

Counts for emotion '당황'
Zero annotation same : 18
One annotation same : 29
Two annotations same : 79
Three annotations same : 149
Sum of two, three : 228
Sum of all : 275

Counts for emotion '슬픔'
Zero annotation same : 29
One annotation same : 43
Two annotations same : 75
Three annotations same : 137
Sum of two, three : 212
Sum of all : 284

모델 비교

□ 학습코드 - SVM

```
# Optuna objective 함수 (SVM 하이퍼파라미터만 탐색)
def objective(trial):
    # SVM 하이퍼파라미터
    svm_c = trial.suggest_float('C', 1e-4, 1e4, log=True)
    svm_gamma = trial.suggest_categorical('gamma', ['scale', 'auto'])

    # SVM 모델 학습 및 평가
    svm = SVC(kernel='rbf', C=svm_c, gamma=svm_gamma)
    svm.fit(train_embeddings, train_labels)
    val_predictions = svm.predict(val_embeddings)
    accuracy = accuracy_score(val_labels, val_predictions)
    return accuracy

# Optuna Study 생성 및 최적화 실행
study = optuna.create_study(direction='maximize')
study.optimize(objective, n_trials=500) # 탐색 횟수 조절
```

SVM은 기초적인 머신러닝 알고리즘이기에,
Optuna를 통해 하이퍼파라미터 튜닝을 진행

모델 비교

□ 학습 코드 - EfficientNet

```
base_model = EfficientNetB0(weights='imagenet', include_top=False, input_shape=(224, 224, 3))

model = Sequential([
    base_model,
    GlobalAveragePooling2D(),
    Dropout(0.5),
    Dense(4, activation='softmax', kernel_regularizer=l2(0.01)) # L2 정규화 추가
])

model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

모델 비교

모델	Validation Loss	Validation Accuracy
VGG 19	0.38	87%
EmotionNet	0.33	88%
EfficientNet	0.27	93%
ViT b16	0.26	91%
SVM	-	83%
ResNet-1	-	88%
DeiT	0.49	91%

- 7개 모델로 학습을 진행
- 정확도가 높은 3개 모델로 Test 셋에 대한 예측을 실행

모델 비교

EfficientNet Validation 정확도

0.9372



EfficientNet Test 정확도

0.7598

- 모든 모델에서 Validation Set 보다 Test 셋에서 정확도가 떨어짐
- 또, 전처리한 이미지로 Test 셋을 예측하는 것과 이미지를 전처리 하면서 동시에 예측하는 것이 예측 결과가 다른 것을 확인

모델 비교

□ ViT b16

일괄 전처리

Anger	: 46%
Happy	: 78%
Panic	: 88%
Sadness	: 88%
Total	: 75%



개별 전처리

Anger	: 70%
Happy	: 98%
Panic	: 93%
Sadness	: 77%
Total	: 85%

모델 비교

모델	일괄 전처리 Accuracy	개별 전처리 Accuracy
ViT	73.03%	86.59%
DeiT	82.53%	83.49%
EffcientNet	75.98%	84.59%

모델 비교

Model	Top-1 Acc.	Top-5 Acc.	#Params	Ratio-to-EfficientNet	#FLOPS	Ratio-to-EfficientNet
EfficientNet-B0	77.3%	93.5%	5.3M	1x	0.39B	1x
ResNet-50 (He et al., 2016)	76.0%	93.0%	26M	4.9x	4.1B	11x
DenseNet-169 (Huang et al., 2017)	76.2%	93.2%	14M	2.6x	3.5B	8.9x
EfficientNet-B1	79.2%	94.5%	7.8M	1x	0.70B	1x
ResNet-152 (He et al., 2016)	77.8%	93.8%	60M	7.6x	11B	16x
DenseNet-264 (Huang et al., 2017)	77.9%	93.9%	34M	4.3x	6.0B	8.6x
Inception-v3 (Szegedy et al., 2016)	78.8%	94.4%	24M	3.0x	5.7B	8.1x
Xception (Chollet, 2017)	79.0%	94.5%	23M	3.0x	8.4B	12x
EfficientNet-B2	80.3%	95.0%	9.2M	1x	1.0B	1x
Inception-v4 (Szegedy et al., 2017)	80.0%	95.0%	48M	5.2x	13B	13x
Inception-resnet-v2 (Szegedy et al., 2017)	80.1%	95.1%	56M	6.1x	13B	13x
EfficientNet-B3	81.7%	95.6%	12M	1x	1.8B	1x
ResNeXt-101 (Xie et al., 2017)	80.9%	95.6%	84M	7.0x	32B	18x
PolyNet (Zhang et al., 2017)	81.3%	95.8%	92M	7.7x	35B	19x
EfficientNet-B4	83.0%	96.3%	19M	1x	4.2B	1x
SENet (Hu et al., 2018)	82.7%	96.2%	146M	7.7x	42B	10x
NASNet-A (Zoph et al., 2018)	82.7%	96.2%	89M	4.7x	24B	5.7x
AmoebaNet-A (Real et al., 2019)	82.8%	96.1%	87M	4.6x	23B	5.5x
PNASNet (Liu et al., 2018)	82.9%	96.2%	86M	4.5x	23B	6.0x
EfficientNet-B5	83.7%	96.7%	30M	1x	9.9B	1x
AmoebaNet-C (Cubuk et al., 2019)	83.5%	96.5%	155M	5.2x	41B	4.1x
EfficientNet-B6	84.2%	96.8%	43M	1x	19B	1x
EfficientNet-B7	84.4%	97.1%	66M	1x	37B	1x
GPipe (Huang et al., 2018)	84.3%	97.0%	557M	8.4x	-	-

We omit ensemble and multi-crop models (Hu et al., 2018), or models pretrained on 3.5B Instagram images (Mahajan et al., 2018).

모델 비교

□ EfficientNet B7

```
base_model = Sequential([
    base_model,
    GlobalAveragePooling2D(),
    Dropout(0.5),
    Dense(4, activation='softmax', kernel_regularizer=l2(0.01))
])
```

가장 기본적인 Layer 구조로 모델을 학습한 결과
Accuracy

Anger : 82%
Happy : 85%
Panic : 64%
Sadness: 88%

Total : 77%

모델 비교

```
model = Sequential([
    base_model,
    GlobalAveragePooling2D(),
    BatchNormalization(),
    Dense(256, kernel_regularizer=l2(0.01)),
    Activation('relu'),
    Dropout(0.5),
    BatchNormalization(),
    Dense(128, kernel_regularizer=l2(0.01)),
    Activation('relu'),
    Dropout(0.5),
    BatchNormalization(),
    Dense(64, kernel_regularizer=l2(0.01)),
    Activation('relu'),
    Dropout(0.5),
    Dense(4, activation='softmax', kernel_regularizer=l2(0.01))
])
```

Depth를 깊게 한 결과

Anger : 74%

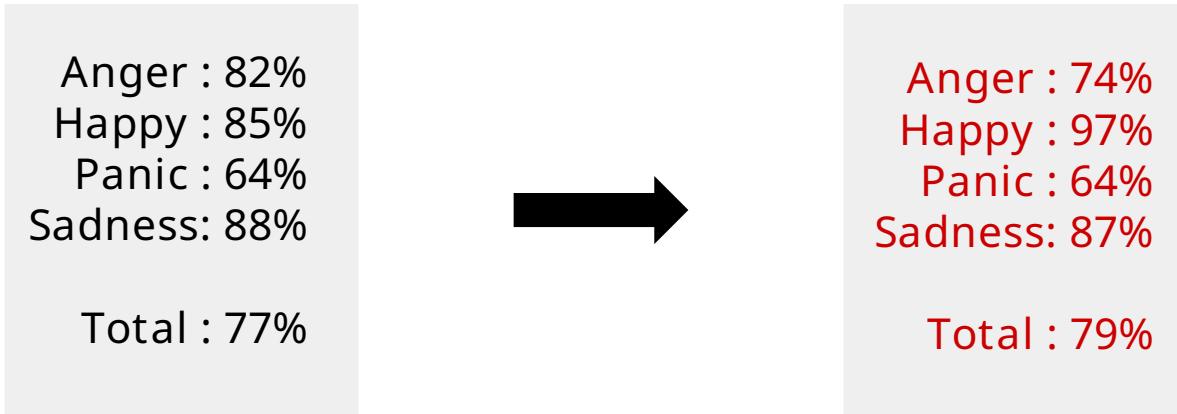
Happy : 97%

Panic : 64%

Sadness: 87%

Total : 79%

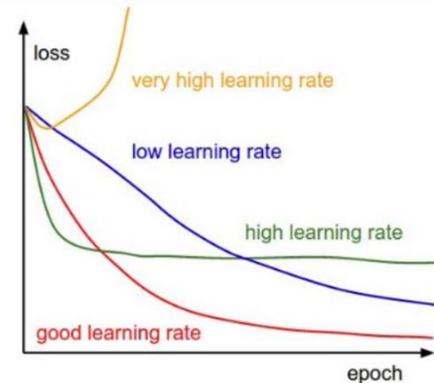
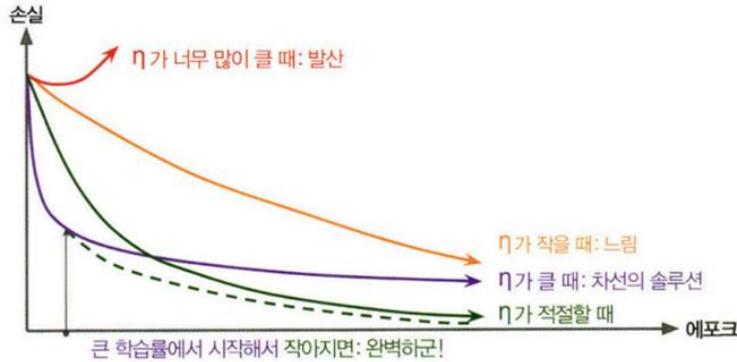
모델 비교



모델 구조가 복잡하면, 정확도가 오르지만 학습시간
이 오래걸리고 무거워짐

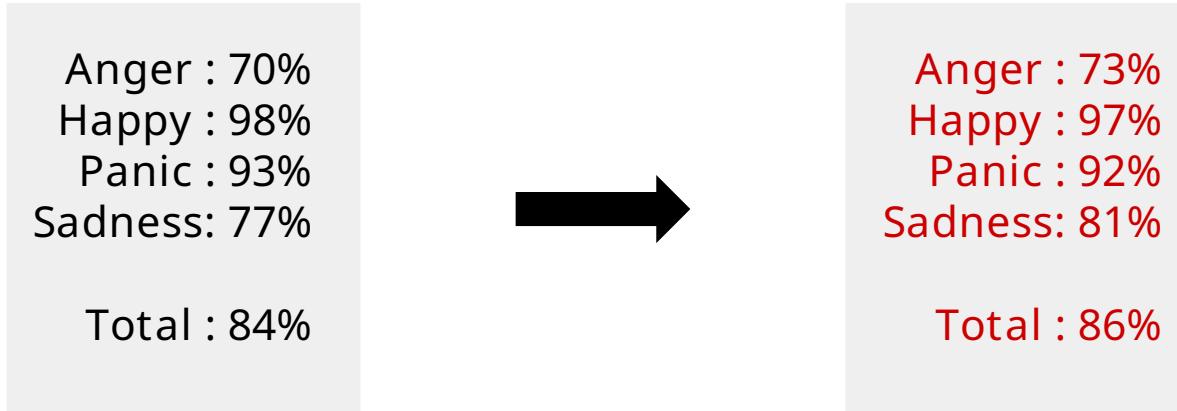
모델 비교

□ 학습률 감소를 도입



```
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=3, min_lr=0.00001)
```

모델 비교



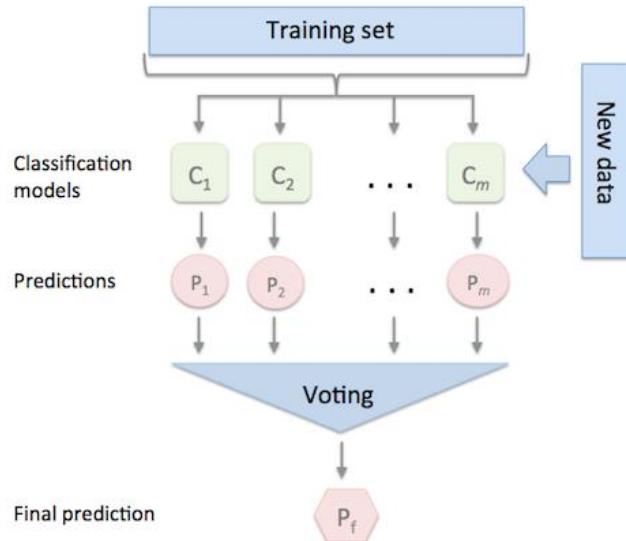
학습률 감소를 도입한 것만으로, 소폭 상승함

Depth를 늘리는것보다 유의미한 효과!

모델 비교

모델	Total	Anger	Happy	Panic	Sadness
ViT B16	86%	72%	96%	87%	90%
EfficientNet B0	86%	73%	97%	92%	81%
EfficientNet B7	84%	61%	98%	94%	82%
DeiT	84%	72%	96%	88%	79%

모델 비교



모든 모델의 결과가 특징이 적고 유사해서
보팅 양상을 도입을 시도

모델 비교

4개중에 2개가 일치하는 경우 하드보팅

```
def vote_and_classify(votes):
    vote_count = {}
    for vote in votes:
        if vote in vote_count:
            vote_count[vote] += 1
        else:
            vote_count[vote] = 1

    max_vote = max(vote_count, key=vote_count.get)
    max_count = vote_count[max_vote]
    vote_prob = max_count / len(votes)

    if max_count >= 2: #모델 4개중 2개가 같다고 판단
        return max_vote, vote_prob
```

정확도 : 75.68%

4개중에 3개가 일치하는 경우 하드 보팅.
이외에, 일부 소프트 보팅의 요소를 도입

```
def vote_and_classify(votes, confidences):
    vote_count = {}
    for vote in votes:
        if vote in vote_count:
            vote_count[vote] += 1
        else:
            vote_count[vote] = 1

    max_vote = max(vote_count, key=vote_count.get)
    max_count = vote_count[max_vote]

    if max_count >= 3: # 모델 4개 중 3개가 같다고 판단
        vote_prob = max_count / len(votes)
        return max_vote, vote_prob
    else:
        # Confidence가 가장 높은 예측을 반환
        max_confidence_index = confidences.index(max(confidences))
        return votes[max_confidence_index], max(confidences)
```

정확도 : 80.00%

Test셋의 일부를 통해 예측 해본 결과,
4개중 3개가 일치하는 경우 하드 보팅 및 일부 소프트 보팅으로 결정

모델 비교

모델	Total	Anger	Happy	Panic	Sadness
ViT B16	86%	72%	96%	87%	90%
EfficientNet B0	86%	73%	97%	92%	81%
EfficientNet B7	84%	61%	98%	94%	82%
DeiT	84%	72%	96%	88%	79%
Voting	80%	81%	92%	68%	81%

Voting 양상률이 생각보다 낮은 결과를 보여주었지만 Anger는 가장 높게 분류

모델 선정

The screenshot shows the AI Hub interface. At the top, there are navigation links: AI 데이터찾기, AI 허브소개, 참여하기, 커뮤니티, and AI 개발지원. Below this, a search bar is labeled "데이터 찾기". A card displays a thumbnail of two sad faces, with buttons for "#감정 분석" and "#표정 분석". The title of the card is "한국인 감정인식을 위한 복합 영상". Below the title, there are filters: "분야" (영상이미지), "유형" (이미지). Below the filters, it says "구축년도: 2020", "생신년월: 2023-10", "조회수: 27,304", "다운로드: 4,395", and "용량: 835.61 GB". At the bottom of the card are buttons for "다운로드" (Download) and "샘플 데이터" (Sample Data).

- 선정한 모델로 실제 서비스를 구현하기 전에, 제공받은 Test셋 이외의 데이터로도 감정 예측을 진행하고자 하였음.
- 실제 서비스는 다양한 사람들을 목표로 삼기 때문에, 실제에 가까운 데이터로 예측 수준을 평가하기 위함

기쁨, 당황, 분노, 슬픔 카테고리에서 2500장씩 총 10000장에 대해서 예측을 진행

모델 선정

```
Emotion: Anger, Average Confidence: 0.89, Accuracy: 0.81  
Emotion: Happy, Average Confidence: 0.95, Accuracy: 0.90  
Emotion: Panic, Average Confidence: 0.93, Accuracy: 0.73  
Emotion: Sadness, Average Confidence: 0.92, Accuracy: 0.78
```

Voting Ensemble Accuracy: 80.30%
Total errors: 1306

- 1만개의 데이터중 1306개의 데이터에서 얼굴 객체를 인식하는 과정에서 Error가 발생한 것으로 추정
- 전처리 Error의 횟수를 제외하고 단순히 모델만의 감정 분류 성능을 평가한다면 주어진 8694개의 데이터를 대상으로 약 92.36%로 평가 가능

```
Counts for emotion '분노'  
Zero annotation same : 828  
One annotation same : 1254  
Two annotations same : 1745  
Three annotations same : 3634  
Sum of two, three : 5379  
Sum of all : 7461  
Counts for emotion '기쁨'  
Zero annotation same : 74  
One annotation same : 106  
Two annotations same : 449  
Three annotations same : 6870  
Sum of two, three : 7319  
Sum of all : 7499  
Counts for emotion '당황'  
Zero annotation same : 697  
One annotation same : 1115  
Two annotations same : 2030  
Three annotations same : 3612  
Sum of two, three : 5642  
Sum of all : 7454  
Counts for emotion '슬픔'  
Zero annotation same : 822  
One annotation same : 1354  
Two annotations same : 1896  
Three annotations same : 3407  
Sum of two, three : 5303  
Sum of all : 7479
```

모델 선정

모델	Total	Anger	Happy	Panic	Sadness
ViT B16	86%	72%	96%	87%	90%
EfficientNet B0	86%	73%	97%	92%	81%
EfficientNet B7	84%	61%	98%	94%	82%
DeiT	84%	72%	96%	88%	79%
Voting	80%	81%	92%	68%	81%
목표치	82%	73%	97%	83%	75%

초기에 설정한 목표 성능수치와 비교하면,
EfficientNet B0와 **ViT B16**이 모두 만족스러운 수치

모델 선정

efficientnet_face_emotion_b7_new.h5	751,692KB	2024-05-30 오후 1:14	H5 파일
efficientnet_face_emotion_new.h5	47,835KB	2024-05-29 오후 7:24	H5 파일
ViT_b16.h5	335,419KB	2024-05-29 오후 9:22	H5 파일
DeiT.safetensors	335,192KB	2024-05-30 오후 1:57	SAFETENSORS 파일

결과적으로 용량, 속도, 정확도, 메모리 리소스를 모두 고려하여
EfficientNet B0을 선정

모델 선정

□ 의문점



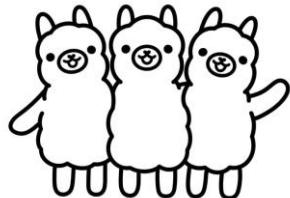
어째서 B7모델보다 B0모델이 더 성능이 좋은가?

→ B7은 파라미터가 커서 더 많은 양의 학습데이터가 필요함. 데이터가 충분하지 않아 **과적합이 발생했을 가능성이 있음.**

또한, 차이가 크지않아 학습을 다시하면 달라질 가능성이 존재

LLM 선정

Meta



ollama run llama3

Llama등 오픈소스 모델

장점 : 로컬에서 실행 가능

단점 : 제대로 사용하기 위해서는 파
인튜닝등의 작업이 필요

VS



ChatGPT

OPENAI에서 제공하는
ChatGPT 등 API

장점 : 성능이 좋고, 컴퓨팅 자원이 들지 않음

단점 : 유료

API 선정



OPENAI ChatGPT

Model	Input	Output
gpt-4o	US\$5.00 / 1M tokens	US\$15.00 / 1M tokens

VS



Claude

Sonnet

- Input: \$3 / MTok
- Output: \$15 / MTok

API 선정

“오늘은 일이 힘든 하루였어요. 할 일이 너무 많아 압도당한 기분이 들고,
모든것을 잘 처리했는지 확신이 들지 않아요.”

sonnet

업무 스트레스로 인해 힘든 하루를 보내셨군요.

하지만 지금 당신이 느끼는 부담감과 불안감은 일시적인 것일 뿐입니다. 모든 일에는 순서와 우선순위가 있습니다. 하나하나 차근차근 해결해 나가면 됩니다. 너무 압박감을 느끼지 마세요. 잠시 멈춰 서서 긍정적인 마음가짐을 갖는 것도 중요합니다. 당신의 노력을 인정하고 격려해드립니다. 곧 지나갈 어려운 시기 일 뿐이니 낙심하지 마세요. 함께 잘 해결해 나갈 수 있을 것입니다.

GPT-4o



오늘 정말 힘든 하루를 보내셨군요. 할 일이 너무 많아 압도당한 기분을 느끼셨고, 모든 것을 잘 처리했는지 확신이 들지 않아 더욱 슬프셨을 거예요. 이런 날은 누구에게나 어려울 수밖에 없죠. 당신의 슬픔에 깊이 공감해요.

이렇게 힘든 상황에서도 최선을 다한 당신이 대단하다고 생각합니다. 스스로에게 조금 더 너그러워지시고, 충분히 쉬는 시간을 가지세요. 오늘의 어려움을 이겨낸 당신을 응원합니다.

프롬프트 조정

```  
사용자의 얼굴 감정과 일기를 보고 답변을 생성하여야함.

1. 태도를 명확히 지켜서 대답해야함.
2. 추상적이고, 일관적인 답변을 하지 않아야함.
3. SNS에 달리는 친구의 댓글처럼 답변하여야함.
4. {a\_attitude}로 댓글을 달아야함.
5. {length}자 내외로 작성하여야함.

얼굴 감정 : {emotion\_kr}

일기 : {inputText}

태도 : {composition[0]}, {composition[1]}, {composition[2]}

6. 얼굴 감정과 일기에 적힌 내용이 일치하지 않으면, 그것에 대해서 물어야함.

```

프롬프트 조정

□ Temperature 조정

0.1

댓글 내용: 와, 오늘 정말 행복한 하루 보냈구나! 😊 무슨 좋은 일이 있었어?
댓글 내용: 와, 오늘 정말 행복한 하루 보냈구나! 😊 무슨 좋은 일이 있었어?

0.5

댓글 내용: 오늘 정말 행복한 하루 보냈구나! 😊 무슨 일이 있었길래 그렇게 행복했어? 좀 더 자세히 들려줘!
댓글 내용: 오늘 진짜 행복한 하루였나 보네! 😊 무슨 좋은 일 있었어? 나도 너의 행복한 기운 좀 나눠줘!

0.75

댓글 내용: 와, 네 얼굴에서 행복이 막 느껴진다! 😊 오늘 뭐 때문에 그렇게 행복한 날이었어?
댓글 내용: 와, 오늘 정말 행복한 하루 보냈구나! 😊 무슨 일 있었어? 좋은 일 있으면 나도 좀 나눠줘!

1.0

댓글 내용: 와, 오늘 정말 행복한 하루 보냈구나! 그럼 도대체 뭐가 그렇게 좋았던 거야? 궁금하네!
댓글 내용: 오늘 진짜 행복해 보여! 무슨 좋은 일 있었어? 😊 네가 이렇게 행복해하는 모습 보니까 나도 기분 좋아진다!

서비스

서비스 소개

Facelog

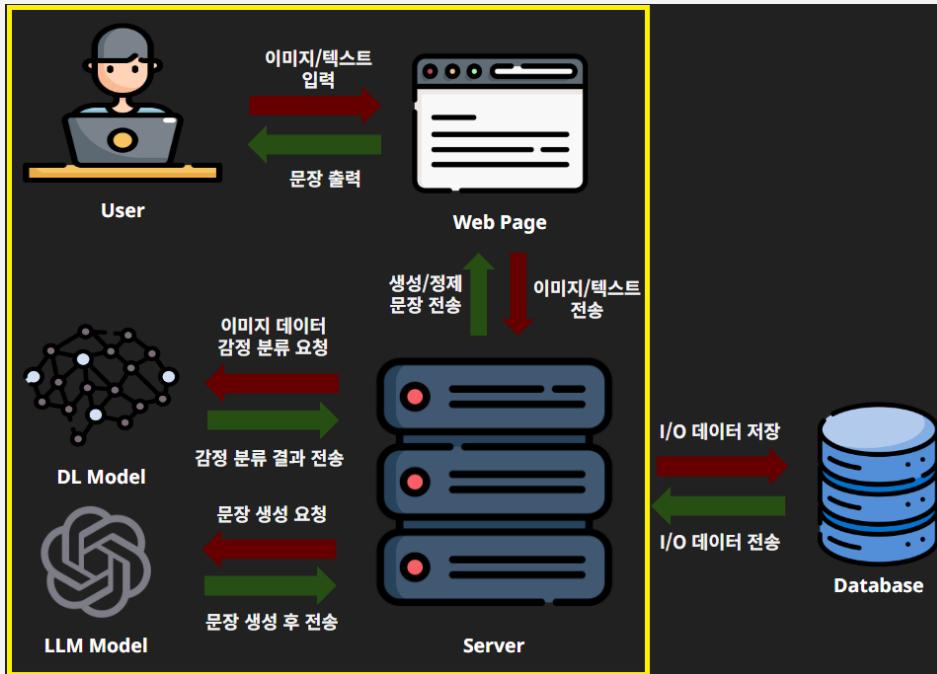


개인용 기밀성 SNS 플랫폼

- 개인용 기밀성 SNS 플랫폼의 프로토타입
- 사용자 정보, 얼굴 이미지, 텍스트, 사용자 맞춤형 프롬프트 설정을 입력받아 멀티모달로 처리 및 저장
- 수집한 데이터들을 API Prompt에 입력 및 반영하여 AI 캐릭터 'Anna'가 SNS 댓글을 생성 Web UI를 통해 출력
- 사용자가 입력한 모든 데이터와 생성된 데이터는 보호 및 저장
- 메인 프로필 페이지의 캘린더를 통해 언제든지 열람 가능

시스템 구성

● 시스템 구성도



해당하는 페이지에서

- 사용자 이름, 사용자 소개
 - 이미지, 텍스트
 - 대화 태도, 답변 구성, 답변 강도, 답변 길이
- 구조에 맞게 입력한다.

USER

입력받은 데이터를 서버로 전송 및 요청한다.

- 이미지 감정 분류 요청
- 답변 생성 요청
- 사용자 정보 업데이트

구현된 UI로 데이터를 사용자에게 출력한다.

WEB

SERVER

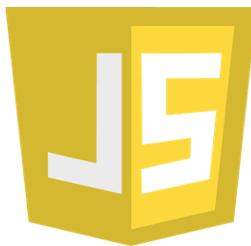
WEB으로부터 요청받은 내용을 수행한다.

개발 환경

Language



Python 3.11.9

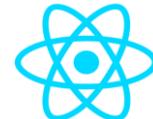


JavaScript

Framework



Tensorflow
2.15.0



React



FastAPI



Flask

web development,
one drop at a time

IDE



Jupyter Notebook



Visual Studio Code

기능 핵심 기능

핵심 기능 1

사용자 입력 이미지의 감정을 분류

- 현재 ViT_b16 모델을 단일 모델로 활용해 이미지의 감정을 분류
- 게시물 작성 페이지를 통해 사용자의 입력을 처리하고 분류된 감정은 UI를 통해 게시물 상세 페이지에서 사용자에게 출력

Emotion of the day

happy



핵심 기능 2.

분류한 감정과 사용자 입력 텍스트를 구조
에 맞추어 LLM API Prompt에 입력

```
prompt = f"""
    사용자의 얼굴 감정과 일기를 보고 답변을 생성하여야함.
    1. 태도를 명확히 지켜서 대답해야함.
    2. 추상적이고, 일관적인 답변을 하지 않아야함.
    3. SNS에 달리는 친구의 댓글처럼 답변하여야함.
    4. {a_attitude}로 댓글을 달아야함.
    5. {length}자 내외로 작성하여야함.

    얼굴 감정 : {emotion_kr}
    일기 : {inputText}
    태도 : {composition[0]}, {composition[1]}, {composition[2]}

    6. 얼굴 감정과 일기에 적힌 내용이 일치하지 않으면, 그것에 대해서 물어야함.
    7. 사용자의 이름은 '{name}'이다. 사용자의 이름을 답변에 포함해야함.
    8. 사용자의 핵심 소개는 '{introduce}'이다. 사용자의 소개를 내용에 참고해야함.

    사용자의 소개와 일기가 관련이 없으면 선택적으로 참고함.

    """
```

기능 핵심 기능

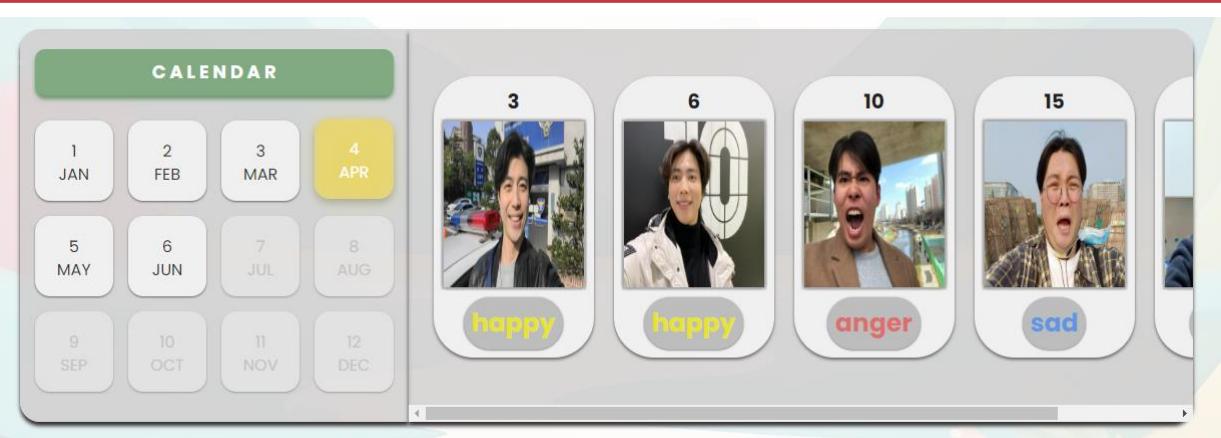
핵심 기능 3

LLM API를 통해 답변을 생성

주성아, 오늘 날씨 진짜 좋더라! 나도 선선한 바람 맞으면서 하루를 시작했어. 평범한 하루를 살아가는 중이라도 이런 좋은 날씨가 기분을 업시키는 것 같아ㅋㅋ 할일 다 마치고 말리랑 산책 가면 완전 힐링이겠다 우리 강아지도 산책 좋아하는데, 요즘은 자주 못 데리고 나가서 미안한 마음이야 ㅠㅠ 주성이는 이렇게 좋은 날씨 속에서 할 일도 척척 해내고 너무 멋지다! 근데 가끔 할 일들 마치기 어려울 때도 있잖아? 그럴 땐 좀 쉬어가면서 해도 좋아, 천천히 해도 괜찮아. 말리랑 즐거운 산책하고 오늘 하루도 기분 좋게 마무리하자! 😊

핵심 기능 4

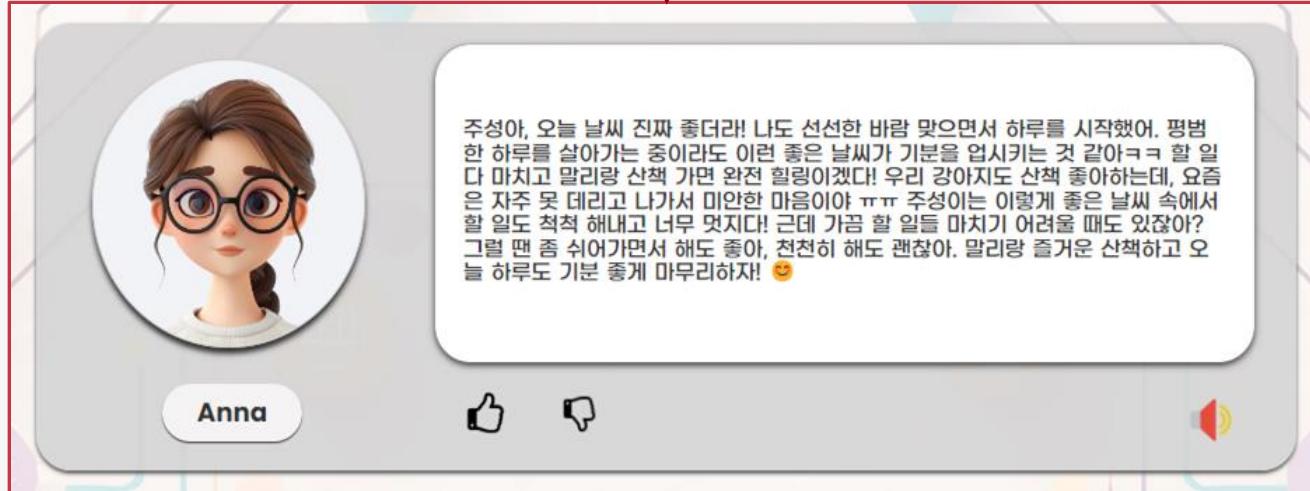
입력 및 생성한 데이터를 저장해 열람 가능하게 함



기능 핵심 기능

핵심 기능 5

출력 구조에 맞추어 SNS 댓글 형식으로 웹페이지에 출력



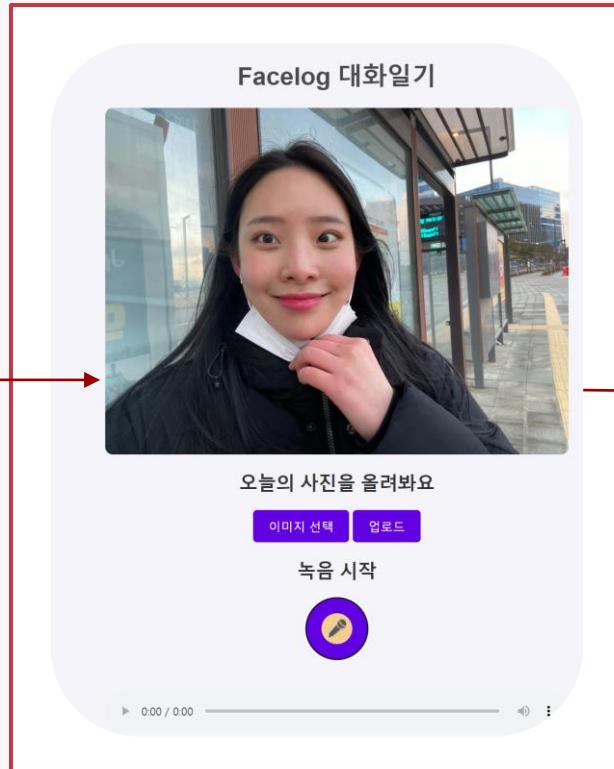
주성아, 오늘 날씨 진짜 좋더라! 나도 선선한 바람 맞으면서 하루를 시작했어. 평범한 하루를 살아가는 중이라도 이런 좋은 날씨가 기분을 업시키는 것 같아ㅋㅋ 할 일 다 마치고 말리랑 산책 가면 완전 힐링이겠다! 우리 강아지도 산책 좋아하는데, 요즘은 자주 못 데리고 나가서 미안한 마음이야ㅠㅠ 주성이는 이렇게 좋은 날씨 속에서 할 일도 척척 해내고 너무 멋지다! 근데 가끔 할 일을 마치기 어려울 때도 있잖아? 그럴 땐 좀 쉬어가면서 해도 좋아, 천천히 해도 괜찮아. 말리랑 즐거운 산책하고 오늘 하루도 기분 좋게 마무리하자! 😊

기능 핵심 기능

핵심 기능 6

사용자와 AI 상호간의 Real Time Chat APPLICATION

- 업로드 버튼 클릭시 이미지의 감정을 분류해 감정이 반영된 인사말 텍스트 출력 및 **TTS** 재생
- 마이크 버튼 클릭(시작), 클릭(중단)시 **STT**를 통해 음성을 텍스트로 변환
- 변환된 텍스트를 **PROMPT**에 입력 후 답변 텍스트 생성
- 텍스트 출력 및 **TTS** 재생



1. 이미지 입력 후 업로드 클릭

오늘 무슨 좋은 일이 있었나요? 왜 이렇게 행복해 보이시죠?

2. ‘마이크 버튼’ 클릭 후 ‘로또 3등에 당첨됐어.’ 음성 입력

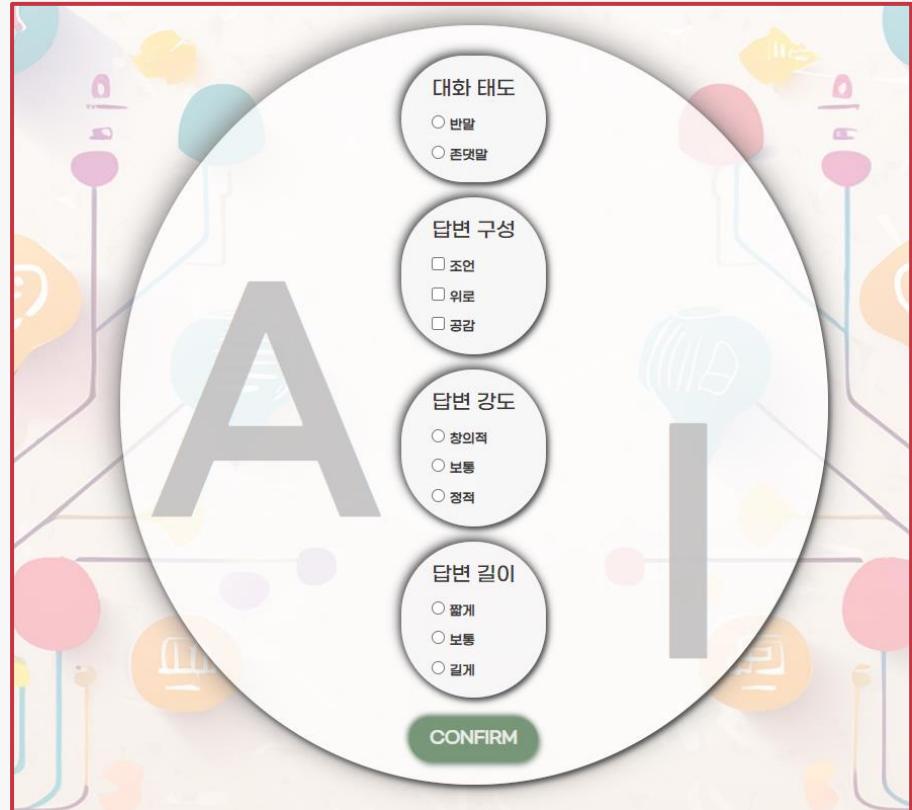
와, 정말 축하드려요! 3등에 당첨되다니 대단하네요.
그 돈으로 뭐 할 계획이세요?

기능 부가 기능

부가 기능 1

사용자 맞춤형 LLM API Prompt 설정

- 사용자는 대화 태도, 답변 강도, 답변 길이를 필수로 한가지 선택
- 답변 구성은 복수로 선택할 수 있으며 아무것도 선택하지 않을 시 조언, 위로, 공감을 모두 선택한 것으로 간주



기능 부가 기능



부가 기능 3

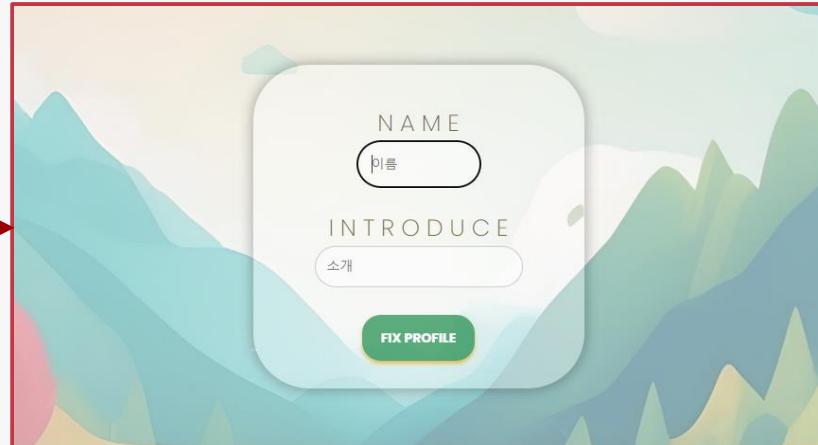
사용자 정보 수정

- 기존에 생성한 사용자의 정보에서 이름과 소개만을 Update, 작성한 게시물의 내역은 유지

부가 기능 2

사용자 생성 및 정보(이름, 소개) 입력 LLM API Prompt에 반영

- 이름은 필수적으로 입력을 요구하며 소개는 사용자 주관으로 선택적 입력 가능
- 입력한 '이름'과 '소개'는 LLM API Prompt에 반영되어 게시물 작성 텍스트에 언급하지 않아도 반영된 결과를 return함.



—

기능 부가 기능

부가 기능 1

사용자 맞춤형

LLM API Prompt 설정

부가 기능 2

사용자 생성 및 정보(이름, 소개) 입력

LLM API Prompt에 반영

부가 기능 3.

사용자 정보 수정

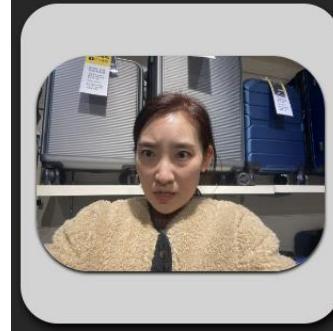
박주성

기아타이거즈의 엄청난 팬

대화 태도 : 반말, 답변 구성 : 조언, 위로, 공감
답변 강도 : 보통(0.75), 답변 길이 : 보통 (300)

부가기능 1, 2, 3의 사용자 입력 Value LLM API Prompt 반영 결과

Facelog



Emotion of the day

anger



오늘 야구 또 지네...5이닝동안 안타 0개라는게 말이 되나

2024년 6월 2일



Anna



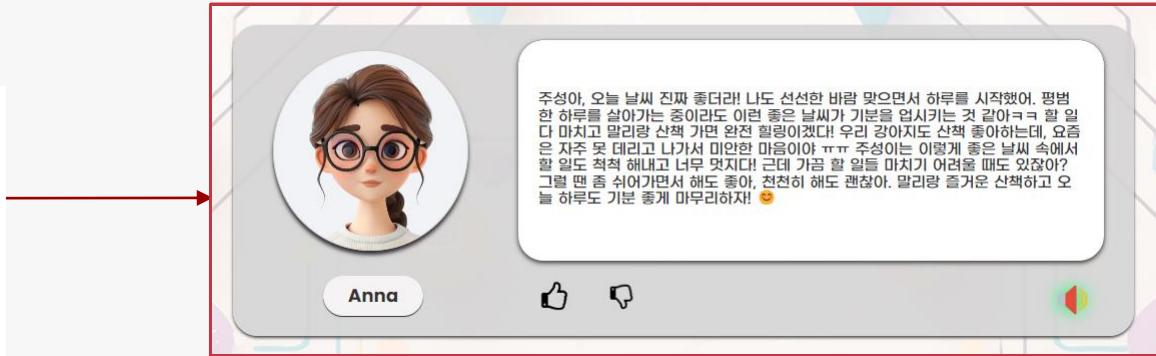
주성야, 야구 또 쳤다고? 진짜 힘빼지겠다... 😠 5이닝 동안 안타 0개라니, 그런 너 무 속상하다. 나도 기아 타이거즈 팬이라 그런지 네가 얼마나 답답할지 조금은 알 것 같아. 선수들이 컨디션이 너무 안 좋았나 보네. 그래도 우리가 여기서 포기할 수는 없잖아? 짜증도 풀고, 다음 경기 때는 더 열심히 응원해보자! 우리 타이거즈, 결국엔 이겨내고 좋은 성적 낼 거야! 그러니까 너무 속상해하지 말고, 함께 끝까지 가자고! 항상 화이팅이야, 주성야! 진짜 힘내자! 💪💪

기능 부가 기능

부가 기능 4

TTS (Text-To-Speech)

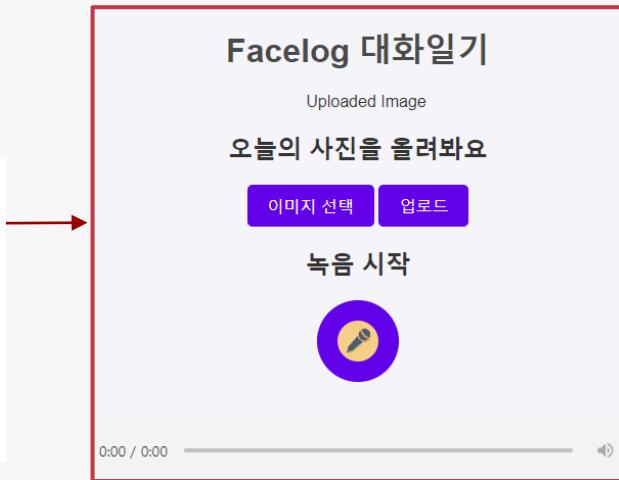
- 스피커 이미지 버튼을 클릭하면 생성된 댓글의 텍스트를 TTS를 통해 음성으로 재생



부가 기능 5

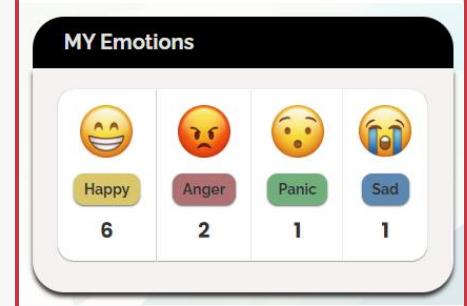
STT (Speech-To-Text)

- 스피커 이미지 버튼을 클릭하면 생성된 댓글의 텍스트를 TTS를 통해 음성으로 재생
- 게시물 작성 페이지에도 추가 예정



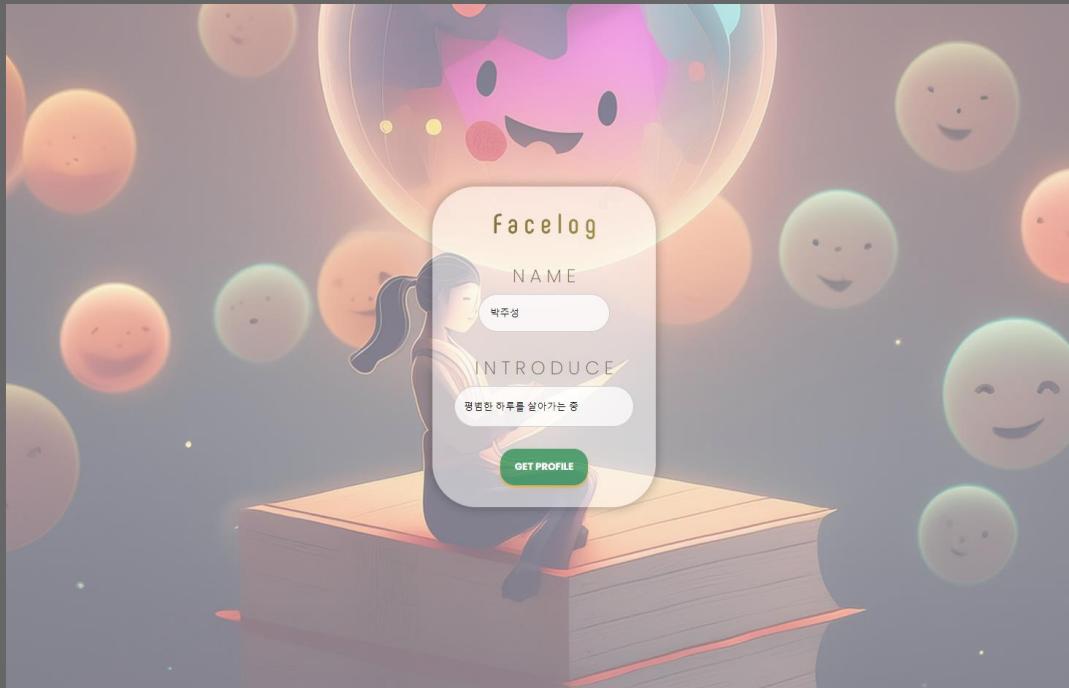
부가 기능 6

누적 감정 카운트



서비스 구현

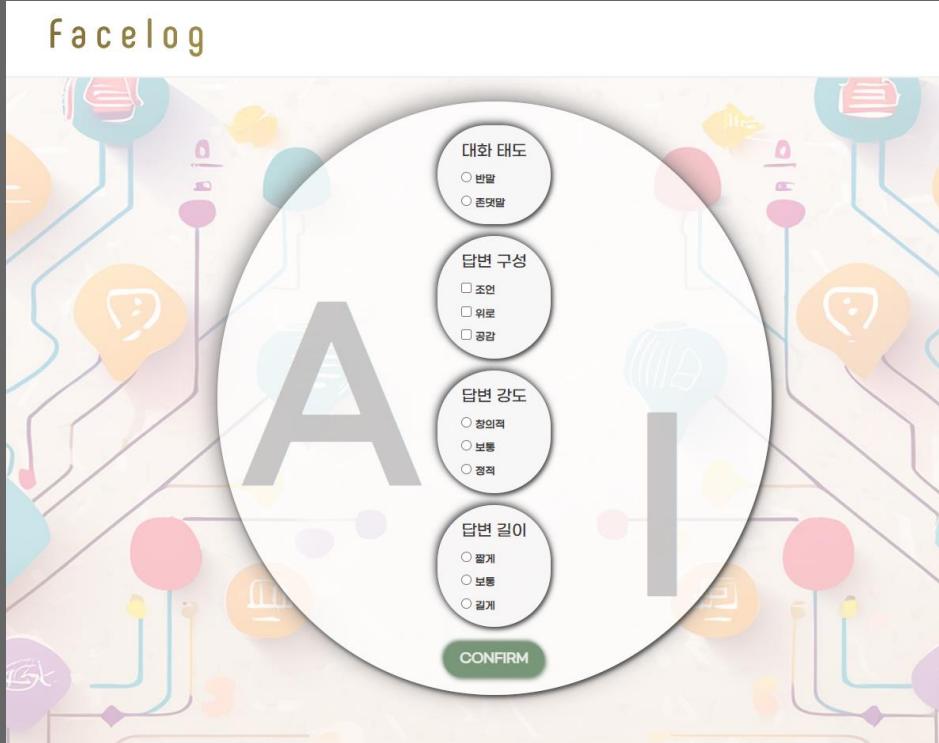
서비스 화면 구성



사용자 정보 입력 페이지

- 사용자의 이름을 필수로 입력받고 소개는 선택적으로 입력받으며 'GET PROFILE' 버튼을 클릭하면 사용자 프로필을 생성함.
- 사용자가 입력한 '이름'과 '소개'는 AI Prompt에 반영되어 게시물 작성 페이지에서 언급하지 않아도 AI Prompt에서 반영된 결과를 출력함.

서비스 화면 구성



프롬프트 커스텀 설정 페이지

- 사용자는 대화 태도, 답변 강도, 답변 길이를 필수적으로 선택해야 하며 답변 구성은 선택하지 않을 시 조언, 위로, 공감 모두 선택한 것으로 간주함.
- 답변 강도는 생성 과정의 Temperature Value를 의미하며 창의적(1.0), 보통(0.75), 정직(0.5)의 값을 가진다.
- 답변 길이는 짧게(150자 내외), 보통(300자 내외), 길게(450자 내외)의 값을 가진다.
- 사용자가 선택한 각각의 Value들을 서버로 전송해 저장하며 수정시 Update가 이루어진다.
- 또한, 사용자의 입력 값은 LLM API Prompt 입력에 모두 반영된다.

서비스 화면 구성

Facelog



박주성

평범한 하루를 살아가는 중



NEW POST

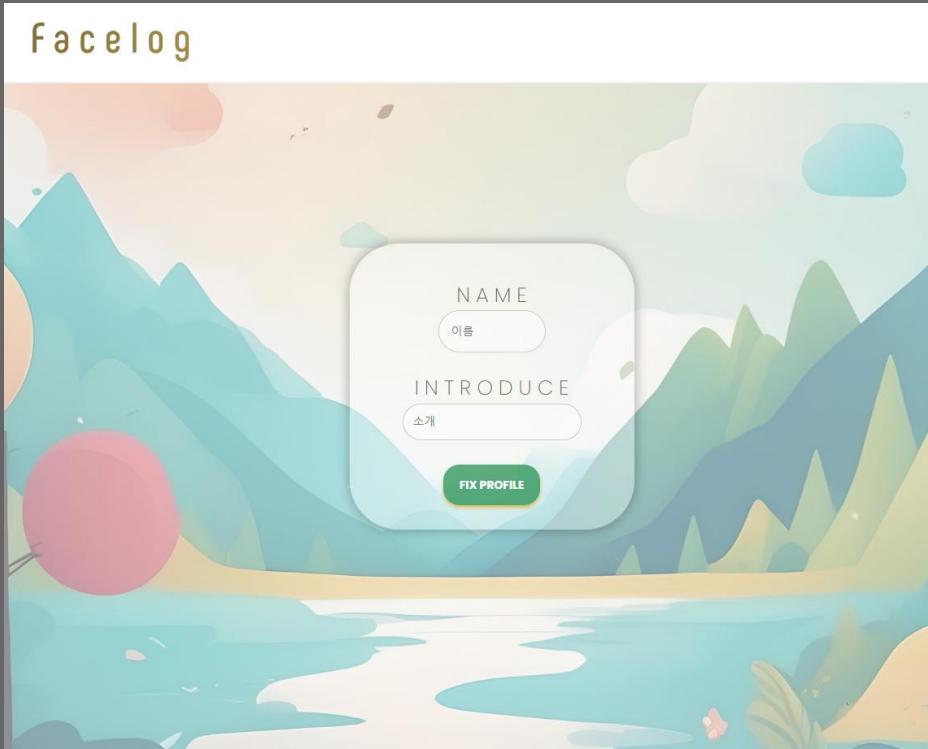
AI SETTING



메인 프로필 페이지

- 'NEW POST' 버튼을 클릭해 게시물 작성 페이지에 진입할 수 있다.
- 'AI SETTING' 버튼을 클릭해 프롬프트 커스텀 설정 페이지에 재진입할 수 있다.
- 프로필 이름 출력 우측의 '설정 이미지 버튼'을 클릭해 사용자 정보 수정 페이지에 진입할 수 있다.
- 'My Emotions' 누적 감정 카운트 출력 위의 'DM 이미지 버튼'을 클릭해 RealTime ChatApp 페이지에 진입할 수 있다.
- 'CALENDAR'의 원하는 Month를 클릭하면 생성 및 저장되어 있는 사용자의 정보를 탐색해 해당하는 Month의 게시물 버튼을 생성하며 기존에 작성되어 있는 게시물을 확인할 수 있고, 게시물을 클릭하면 게시물 상세 페이지로 진입할 수 있다.

서비스 화면 구성



사용자 정보 수정 페이지

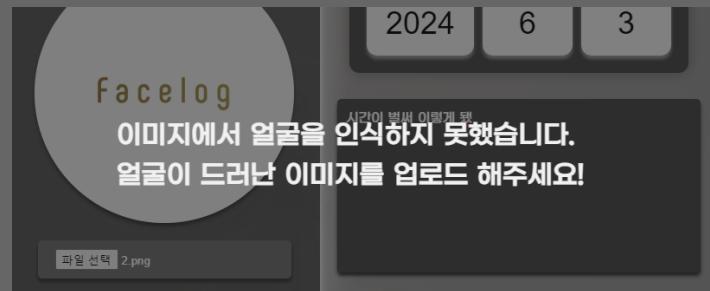
- 사용자의 이름을 필수로 입력받고 소개는 선택적으로 입력받는다.
- 기존에 생성된 클래스에 Update하며 기존에 작성 및 저장된 게시물 내역은 유지함.

서비스 화면 구성



5. 게시물 작성 페이지

- 이미지, 연도, 월, 날짜, 텍스트의 입력을 요구하는 페이지
- 하나의 필드라도 입력되지 않으면 진행되지 않는다.
- 얼굴이 드러나지 않은 이미지를 업로드하면 화면에 2.5초간 '이미지에서 얼굴을 인식하지 못했습니다. 얼굴이 드러난 이미지를 업로드해주세요!'를 출력 후 페이지가 reload된다.



서비스 화면 구성

Facelog



Emotion of the day

happy



오늘 날씨도 좋고, 바람도 선선해서 여름치고 체감온도도 나쁘지 않은 것 같아. 얼른 할 일을 마치고 우리집 강아지 말리랑 산책가야지ㅋㅋ

2024년 6월 3일

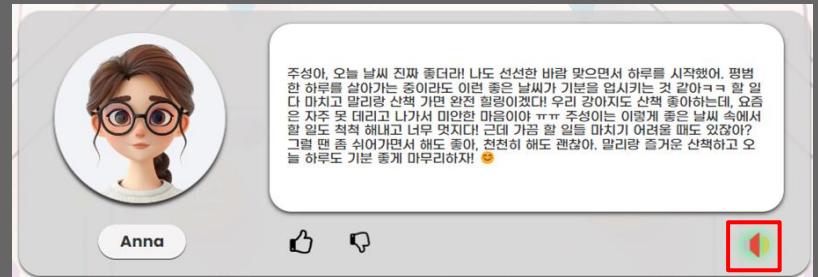


주성아, 오늘 날씨 진짜 좋더라! 나도 선선한 바람 맞으면서 하루를 시작했어. 평범한 하루를 살아가는 중이라도 이런 좋은 날씨가 기분을 업시키는 것 같아ㅋㅋ 할 일 다 마치고 말리랑 산책 가면 완전 힐링이겠대! 우리 강아지도 산책 좋아하는데, 요즘은 자주 못 데리고 나가서 미안한 마음이야ㅠㅠ 주성이는 이렇게 좋은 날씨 속에서 할 일도 척척 해내고 너무 멋지다 근데 가끔 할 일을 마치기 어려울 때도 있잖아? 그럼 면 좀 쉬어가면서 해도 좋아, 천천히 해도 괜찮아. 말리랑 즐거운 산책하고 오늘 하루도 기분 좋게 마무리하자! 😊



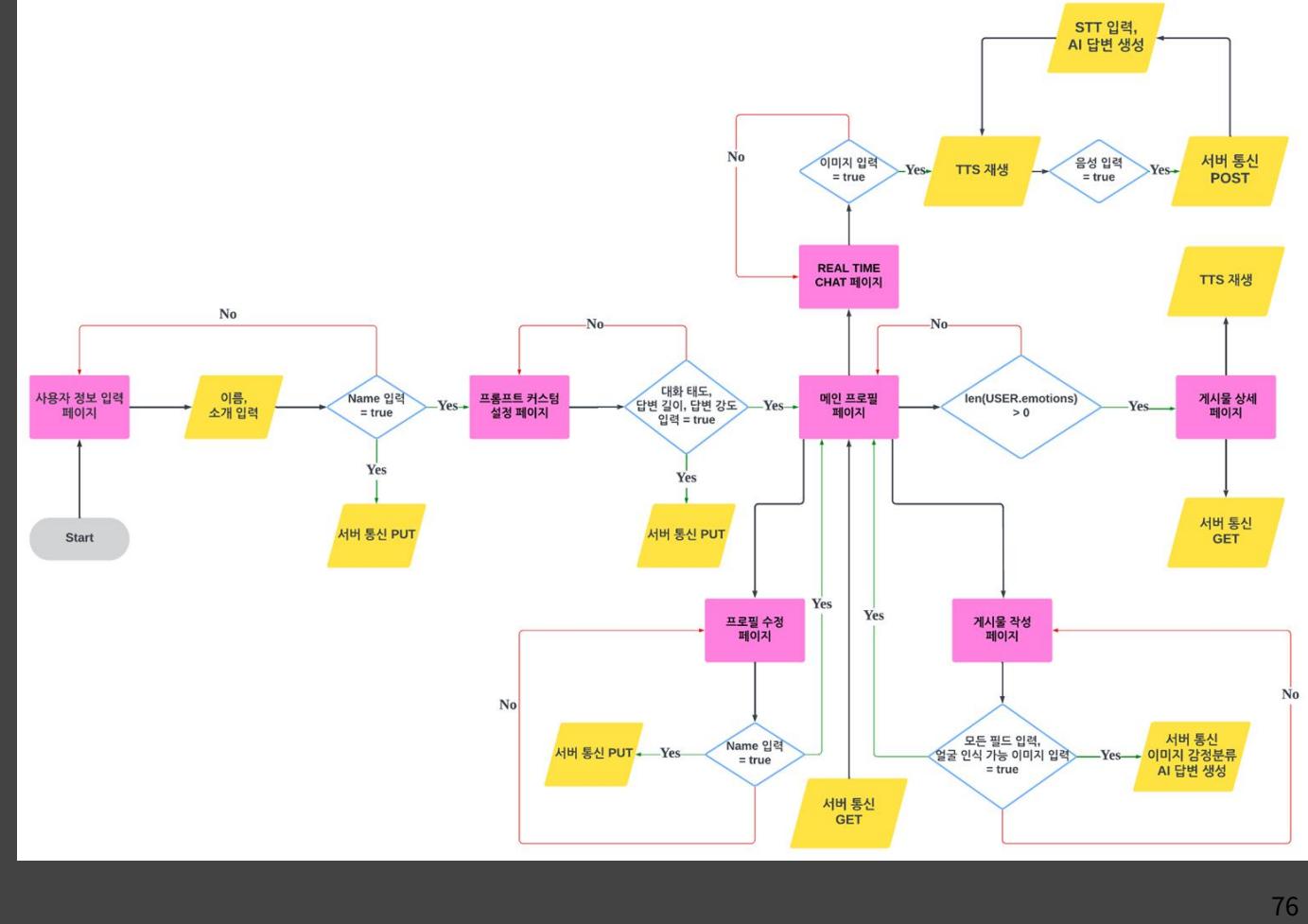
6. 게시물 상세 페이지

- 입력 이미지 데이터, 감정 분류 결과, 입력 텍스트 데이터, 입력 날짜 데이터를 사용자 필드에 출력하고 AI 필드에는 AI 캐릭터 'Anna'와 생성된 답변 텍스트를 출력한다.
- 오른쪽 하단의 '스피커 이미지 버튼'을 클릭하면 TTS 기능이 실행되며 생성된 텍스트를 음성으로 재생한다.

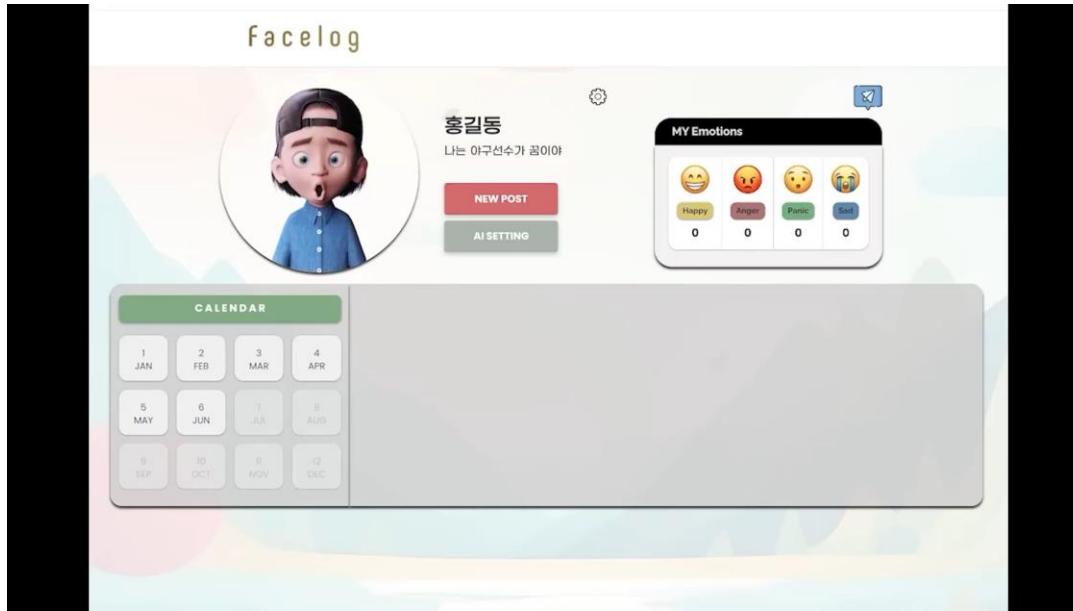


동작 로직

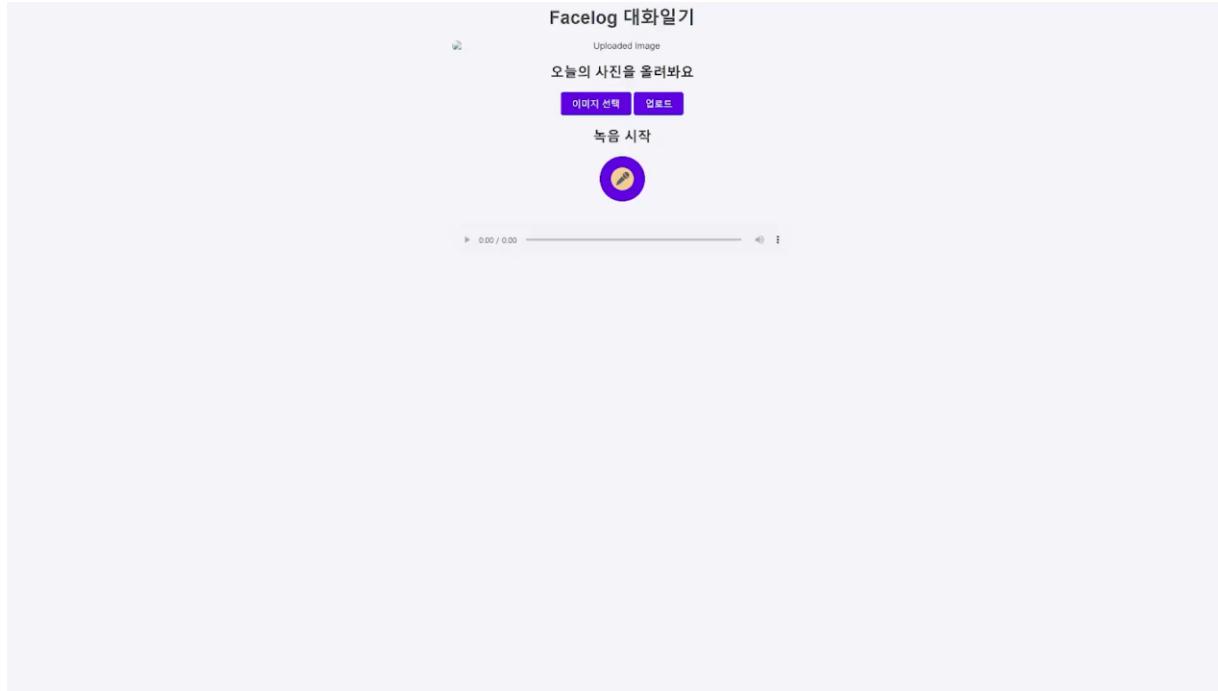
□ 시스템 흐름도



시연 동영상 1



시연 동영상 2



— 시연

Facelog

마무리

문제점 및 개선방안

PROBLEM

외부 API에 대한 높은 의존성

학습 내용 프로젝트 소요 시간상 사용이 불가피



SOLUTION

랭체인을 활용한 자체 sLLM 모델 구현

외부 API에 의존하지 않고 동작을 보장



API 입력 토큰 수 과다

한국어의 특성 상 필요 토큰 수가 방대함



프롬프트 최적화

프롬프트 최적화를 통해 필요 토큰 수 조절



문제점 및 개선방안

PROBLEM



LLM 모델의 한계

LLM 모델의 답변은 실제 전문가의 답변에 비해
상대적으로 전문성이 부족함



데이터 불균형

제한된 데이터 양으로 인하여 편향성을 보임

SOLUTION



전문가 답변 데이터를 통한 파인튜닝

실제 전문가의 상담 기록 등을 이용하여 파인튜닝 진행



추가 학습 데이터 확보

학습 데이터를 확충하여 균형있는 데이터 보장

문제점 및 개선방안

PROBLEM



기존 SNS 서비스 대체의 어려움

기존 상용 서비스들과 비교하였을 때
특징이 매우 다르기 때문에 대체를 고민하기 힘듦



비즈니스 모델의 한계

수익 창출에 필요한 사업성 확보에 한계점이 명확함

SOLUTION



확실한 차별점 확보

BeReal 등의 새로운 SNS가 그랬듯이 차별점 확보가 필수적



이용자 확보를 통한 장기서비스 도모

SNS로서 기능하기 위해선 많은 이용자가 필요하므로 최대한 확보

제안사항

MODEL

- 모델 성능, 전처리 과정 개선
- EfficienNet, ViT 기반
Hybrid Custom 모델 설계

SERVICE

- API 입력 토큰 수 조정
- 사용자수 증가에 따른 API 토큰 사용
- 데이터베이스 구현 및 연동
- 다양한 AI 캐릭터 생성
- 로컬 LLM 활용 방안 모색
- RAG sLLM 구현
- MOOD TRACKER 서비스 추가
(사용자 감정 변화 REPORT)
- 감정 상태 진단
- 감정 상태에 대한 컨텐츠 추천
- 실제 서비스 배포시 보안 정책 수립

참고자료

<https://github.com/co60ca/EmotionNet>

<https://arxiv.org/pdf/1905.11946.pdf> #EfficientNet

<https://openreview.net/forum?id=YicbFdNTTy> #ViT

<https://arxiv.org/abs/2012.12877> #DeiT

<https://aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&aihubDataSe=realm&dataSetSn=82> #한국인 얼굴 영상

Q & A

THANK YOU