안녕하세요, 2팀 발표를 맡은 000입니다. 발표 시작하겠습니다.

저희의 주제는 딥러닝을 이용한 감정인식과 그에 따른 음악 추천 프로그램입니다.

오늘 발표의 순서는 감정인식모델, 음악추천 방식, 웹사이트 구현, 앞으로의 일정 순으로 발표하도록 하겠습니다.

감정인식 모델구현에 설명하기에 앞서 저희는 지난주에 사용한 표정인식을 위한 딥러닝 모델인 VGG모델이 정확도는 높은편이지만, parameters,매개변수가 약 1억개가 있는만큼 학습속도가 느리고 용량이 커서 성능이 떨어지므로, 기존의 VGG 모델에서 mini-xception 모델로 변경하였습니다. mini-xception모델은 매개변수가 약6만개로 VGG보다도 학습속도가 빠르며, 저희가 사용하는 Fer2013 데이터셋에 대해서 가장 높은 정확도를 보여줍니다.

이러한 이유로 모델을 변경하였습니다.

다음으로는 감정인식 모델구현에 대해 설명 드리겠습니다. 지금 보이는 사진은 저희가 노트북 웹캠을 통해 실시간으로 감정과 그에 대한 감정의 인식률을 나타내고 있는 모습입니다. 이에 대해서는 앞으로 시간을 정해 그 시간 동안 인식된 최종감정상태를 출력할 수 있도록 코드를 수정할 계획입니다.

다음으로 음악 추천 방식에 대해 설명 드리겠습니다.

감정인식 모델을 통해 추출된 감정에 따라 사전에 지정해 놓은 키워드를 유튜브에 검색하여 해당 URL을 제시하는 방식으로 추천 할 예정입니다. 사실 이 부분은 웹사이트 안에서 재생을 하거나, 음악 추천리스트와 함께 스트리밍 사이트를 연결하려고 노력중이고, 가능하면 그렇게 구현할 계획입니다.

다음으로는 웹사이트 구현에 대해 설명 드리겠습니다.

저희는 sublime text3를 이용하여 flask를 통한 웹페이지 개발을 하였습니다.

아직은 웹사이트에서 실시간 웹캠을 통한 얼굴인식은 넣지 못하였지만, 이미지파일을 업로드하면, 해당 사진에서 얼굴을 harascade 방식으로 인식한 뒤, 저희가 학습시킨 모델로 감정인식하는 것에는 성공하였습니다. 앞으로 웹사이트에서 웹캠을 통한 실시간 감정인식을 실현시키고, 음악추천 및 재생을 넣을 계획입니다.

마지막으로 앞으로의 일정에 대해 말씀드리겠습니다. 차주에는 실시간 감정인식을 웹사이트에 적용하고, 웹사이트에서 감정인식을 통해서 음악추천 재생 프로그램을 최종설계할 계획입니다.

-딴지

1. mini-xception이 어떤구조인지

def mini\_XCEPTION(input\_shape, num\_classes, l2\_regularization=0.01):

regularization = l2(l2\_regularization)

# base

img\_input = Input(input\_shape)

x = Conv2D(8, (3, 3), strides=(1, 1), kernel\_regularizer=regularization,

use\_bias=False)(img\_input)

x = BatchNormalization()(x)

x = Activation('relu')(x)

x = Conv2D(8, (3, 3), strides=(1, 1), kernel\_regularizer=regularization,

use\_bias=False)(x)

x = BatchNormalization()(x)

x = Activation('relu')(x)

# module 1

residual = Conv2D(16, (1, 1), strides=(2, 2),

padding='same', use\_bias=False)(x)

residual = BatchNormalization()(residual)

x = SeparableConv2D(16, (3, 3), padding='same',

kernel\_regularizer=regularization,

use\_bias=False)(x)

x = BatchNormalization()(x)

x = Activation('relu')(x)

x = SeparableConv2D(16, (3, 3), padding='same',

kernel\_regularizer=regularization,

use\_bias=False)(x)

x = BatchNormalization()(x)

x = MaxPooling2D((3, 3), strides=(2, 2), padding='same')(x)

x = layers.add([x, residual])

# module 2

residual = Conv2D(32, (1, 1), strides=(2, 2),

padding='same', use\_bias=False)(x)

residual = BatchNormalization()(residual)

x = SeparableConv2D(32, (3, 3), padding='same',

kernel\_regularizer=regularization,

use\_bias=False)(x)

x = BatchNormalization()(x)

x = Activation('relu')(x)

x = SeparableConv2D(32, (3, 3), padding='same',

kernel\_regularizer=regularization,

use\_bias=False)(x)

x = BatchNormalization()(x)

x = MaxPooling2D((3, 3), strides=(2, 2), padding='same')(x)

x = layers.add([x, residual])

# module 3

residual = Conv2D(64, (1, 1), strides=(2, 2),

padding='same', use\_bias=False)(x)

residual = BatchNormalization()(residual)

x = SeparableConv2D(64, (3, 3), padding='same',

kernel\_regularizer=regularization,

use\_bias=False)(x)

x = BatchNormalization()(x)

x = Activation('relu')(x)

x = SeparableConv2D(64, (3, 3), padding='same',

kernel\_regularizer=regularization,

use\_bias=False)(x)

x = BatchNormalization()(x)

x = MaxPooling2D((3, 3), strides=(2, 2), padding='same')(x)

x = layers.add([x, residual])

# module 4

residual = Conv2D(128, (1, 1), strides=(2, 2),

padding='same', use\_bias=False)(x)

residual = BatchNormalization()(residual)

x = SeparableConv2D(128, (3, 3), padding='same',

kernel\_regularizer=regularization,

use\_bias=False)(x)

x = BatchNormalization()(x)

x = Activation('relu')(x)

x = SeparableConv2D(128, (3, 3), padding='same',

kernel\_regularizer=regularization,

use\_bias=False)(x)

x = BatchNormalization()(x)

x = MaxPooling2D((3, 3), strides=(2, 2), padding='same')(x)

x = layers.add([x, residual])

x = Conv2D(num\_classes, (3, 3),

#kernel\_regularizer=regularization,

padding='same')(x)

x = GlobalAveragePooling2D()(x)

output = Activation('softmax',name='predictions')(x)

model = Model(img\_input, output)

return model