프로젝트 계획서

김예원

<데이터 전처리>

- 사진 테두리 자르는 작업
  - 필요없다고 생각, 같은 사진을 이용한 데이터 셋이 이미 성공적으로 만들어졌기 때문
- 사진 이름 바꾸는 작업
  - 필요없다고 생각, 이미 감정별로 폴더가 구분되어 있기 때문

<데이터셋>

[참고한 데이터셋]

https://www.kaggle.com/code/soumyadeepp/face-emotion-recognition/data

[필요한 절차]

감정별로 이미지 체크

train과 test데이터로 나누고 valid

모델 build

모델 시각화

모델 fitting(epoch 이용)

accuracy & loss 확인

```
picture_size
```

45 ~ 48 정도가 적당하다고 생각

```
expression = 'disgust'
```

'disgust'를 예로 들어 설명

```
plt.figure(figsize= (12,12))
for i in range(1, 10, 1):
    plt.subplot(3,3,i)
    img = load_img(folder_path+"train/"+expression+"/"+
                   os.listdir(folder_path + "train/" + expression)[i],
target_size=(picture_size, picture_size))
```

3*3 으로 image 보여줌

```
batch_size  = 128

datagen_train  = ImageDataGenerator()
datagen_val = ImageDataGenerator()

train_set = datagen_train.flow_from_directory(folder_path+"train",
                                              target_size =
(picture_size,picture_size),
                                              color_mode = "grayscale",
                                              batch_size=batch_size,
                                              class_mode='categorical',
                                              shuffle=True)
```

∧ train data

class mode = categorical

: 2D one-hot 부호화된 라벨 반환

```
test_set = datagen_val.flow_from_directory(folder_path+"validation",
                                              target_size =
(picture_size,picture_size),
                                              color_mode = "grayscale",
                                              batch_size=batch_size,
                                              class_mode='categorical',
                                              shuffle=False)
```

∧ test data

```
def get_sequential_model(input_shape):
    model = keras.Sequential(
        [
           layers.Input(input_shape),

           layers.Conv2D(64, 3, strides=1, activation='relu', padding='same'),
            layers.Conv2D(64, 3, strides=1, activation='relu', padding='same'),
            layers.MaxPool2D(),
            layers.BatchNormalization(),
            layers.Dropout(0.5),

            # 2nd
            layers.Conv2D(128, 3, strides=1, activation='relu',padding='same'),
            layers.Conv2D(128, 3, strides=1, activation='relu',padding='same'),
            layers.MaxPool2D(),
            layers.BatchNormalization(),
            layers.Dropout(0.3),

            # Classifier
            layers.GlobalMaxPool2D(),
            layers.Dense(128, activation='relu'),
            layers.Dense(1, activation='sigmoid')
        ]
    )
    return model
```

∧ CNN Layer를 사용해서 이미지 분석

activation = relu 사용

max pooling 사용

```
class SimpleCNN(keras.Model):
    def __init__(self):
        super(SimpleCNN, self).__init__()

        self.conv_block_1 = keras.Sequential(
            [
                layers.Conv2D(64, 3, strides=1, activation='relu',
padding='same'),
                layers.Conv2D(64, 3, strides=1, activation='relu',
padding='same'),
                layers.MaxPool2D(),
                layers.BatchNormalization(),
                layers.Dropout(0.5)
            ], name='conv_block_1'
        )

        self.conv_block_2 = keras.Sequential(
            [
                layers.Conv2D(128, 3, strides=1, activation='relu',
padding='same'),
                layers.Conv2D(128, 3, strides=1, activation='relu',
padding='same'),
                layers.MaxPool2D(),
                layers.BatchNormalization(),
                layers.Dropout(0.3)
            ], name='conv_block_2'
        )

        self.classifier = keras.Sequential(
            [
                layers.GlobalMaxPool2D(),
                layers.Dense(128, activation='relu'),
                layers.Dense(1, activation='sigmoid')
            ], name='classifier'
        )

    def call(self, input_tensor, training=False):
        x = self.conv_block_1(input_tensor)
        x = self.conv_block_2(x)
        x = self.classifier(x)
        return x
history = model.fit(
    train_generator,
    validation_data=valid_generator,
    epochs=10,
    verbose=1
)
```

∧ 위 두 데이터는 학원에서 배운 내용 그대로 복붙함