**교육일지**

**교육 제목 : 딥러닝**

**교육 장소 : YGL C6 강의실**

**교육 일시 : 2021/10/28**

**MNlST\_CNN**

**import matplotlib.pyplot as plt**

**import os**

**import tensorflow as tf**

**import numpy as np**

**(x\_train, y\_train), (x\_test, y\_test) = tf.keras.datasets.mnist.load\_data()**

**# 데이터 (배치사이즈 x 28 x 28 x 1)로 이미지로 변환 --> 그레이스케일 채널은 1**

**x\_train = x\_train.reshape(x\_train.shape[0], 28, 28, 1).astype('float32')**

**x\_test = x\_test.reshape(x\_test.shape[0], 28, 28, 1).astype('float32')**

**y\_train = tf.keras.utils.to\_categorical(y\_train)**

**y\_test = tf.keras.utils.to\_categorical(y\_test)**

**# CNN 모델 설계**

**input\_layer = tf.keras.layers.Input(shape=(28, 28, 1))**

**x = tf.keras.layers.Conv2D(32, (3, 3), strides =1, activation='relu', padding='same')(input\_layer)**

**x = tf.keras.layers.MaxPool2D((2, 2))(x)**

**x = tf.keras.layers.Conv2D(64, (3, 3), strides=1, activation='relu')(x)**

**x = tf.keras.layers.MaxPool2D((2, 2))(x)**

**x = tf.keras.layers.Flatten()(x)**

**output\_layer = tf.keras.layers.Dense(10, activation='softmax')(x)**

**model = tf.keras.Model(inputs=[input\_layer], outputs=[output\_layer])**

**model.summary()**

**loss = tf.keras.losses.categorical\_crossentropy**

**optimizer = tf.keras.optimizers.RMSprop(learning\_rate=0.0001)**

**metrics = tf.keras.metrics.categorical\_accuracy**

**model.compile(loss=loss,**

**optimizer = optimizer,**

**metrics= [metrics])**

**model\_dir = './cnn\_mnist\_model/'**

**if not os.path.exists(model\_dir):**

**os.mkdir(model\_dir)**

**model\_path='./cnn\_mnist\_model/{epoch:02d}-{val\_loss:.4f}.hdf5'**

**callback\_list = [tf.keras.callbacks.ModelCheckpoint(filepath=model\_path, monitor='val\_loss', verbose=1, save\_best\_only=True),**

**tf.keras.callbacks.EarlyStopping(monitor='val\_loss', patience=10)]**

**history = model.fit(x\_train, y\_train, validation\_split=0.2, epochs=5, batch\_size=200, verbose=1, callbacks=callback\_list)**

**# 테스트 정확도 출력**

**print("\n Test Accuracy: %.4f" % (model.evaluate(x\_test, y\_test)[1]))**

**# 테스트 셋의 오차**

**y\_vloss = history.history['val\_loss']**

**# 학습셋의 오차**

**y\_loss = history.history['loss']**

**# 그래프로 표현**

**x\_len = np.arange(len(y\_loss))**

**plt.plot(x\_len, y\_vloss, marker='.', c="red", label='Testset\_loss')**

**plt.plot(x\_len, y\_loss, marker='.', c="blue", label='Trainset\_loss')**

**# 그래프에 그리드를 주고 레이블을 표시**

**plt.legend(loc='upper right')**

**plt.grid()**

**plt.xlabel('epoch')**

**plt.ylabel('loss')**

**plt.show()**