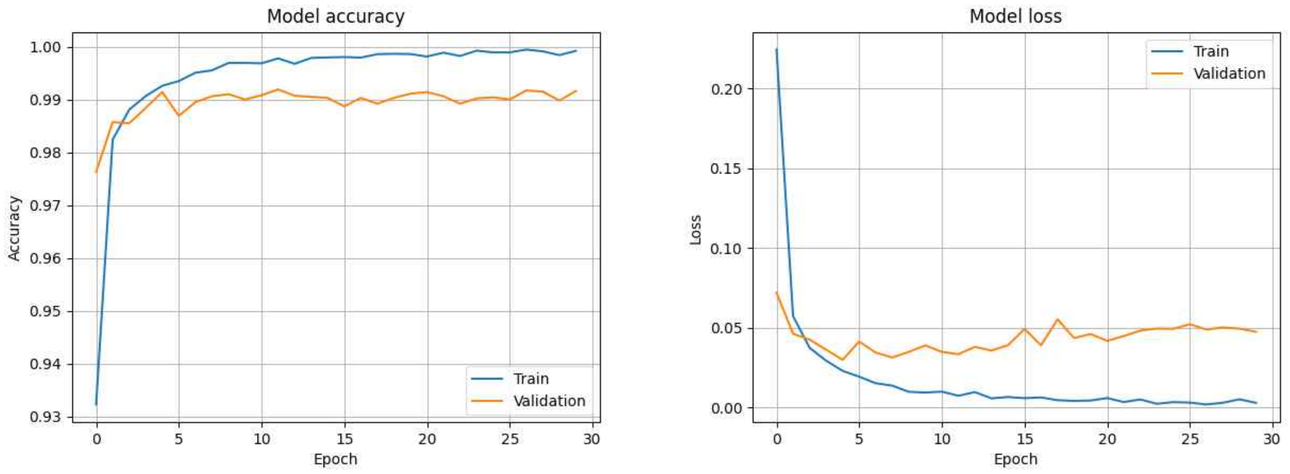
2021254010 이지호 딥러닝실제 13주차 과제

(과제) 프로그램 6-1과 6-2를 수행하여 결과를 정리하고, 프로그램의 동작을 설명하시오.

- 결과 정리

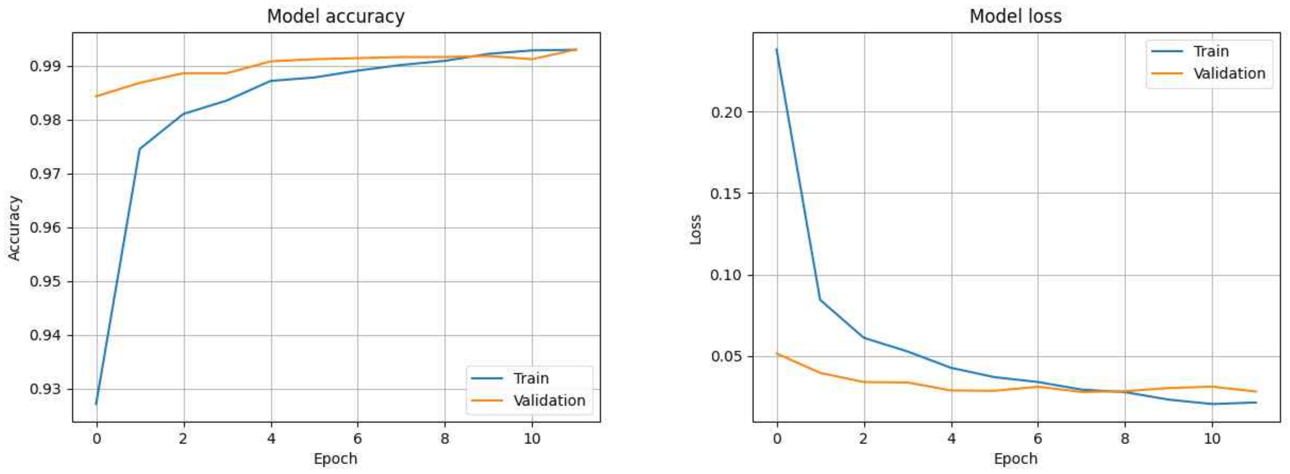
6-1.py

```
Epoch 29/30
469/469 - 14s - loss: 0.0052 - accuracy: 0.9984 - val_loss: 0.0495 - val_accuracy: 0.9898
Epoch 30/30
469/469 - 13s - loss: 0.0029 - accuracy: 0.9992 - val_loss: 0.0475 - val_accuracy: 0.9916
정확률은 99.1599977016449
```



6-2.py

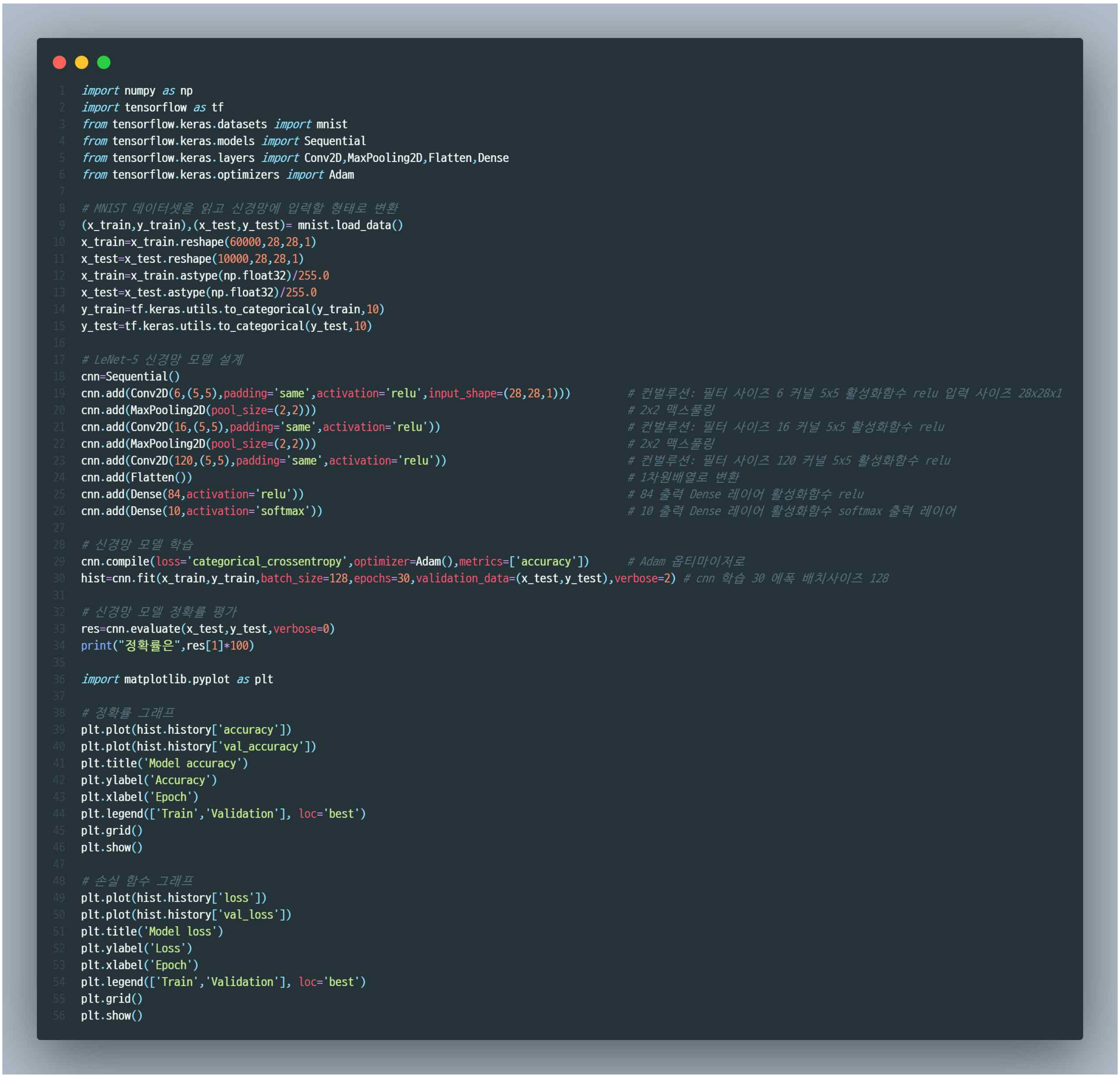
```
Epoch 11/12
469/469 - 36s - loss: 0.0205 - accuracy: 0.9928 - val_loss: 0.0312 - val_accuracy: 0.9912
Epoch 12/12
469/469 - 35s - loss: 0.0215 - accuracy: 0.9930 - val_loss: 0.0282 - val_accuracy: 0.9930
정확률은 99.29999709129333
```



- 동작 설명 (주석으로 레이어부분 설명)

6-1.py

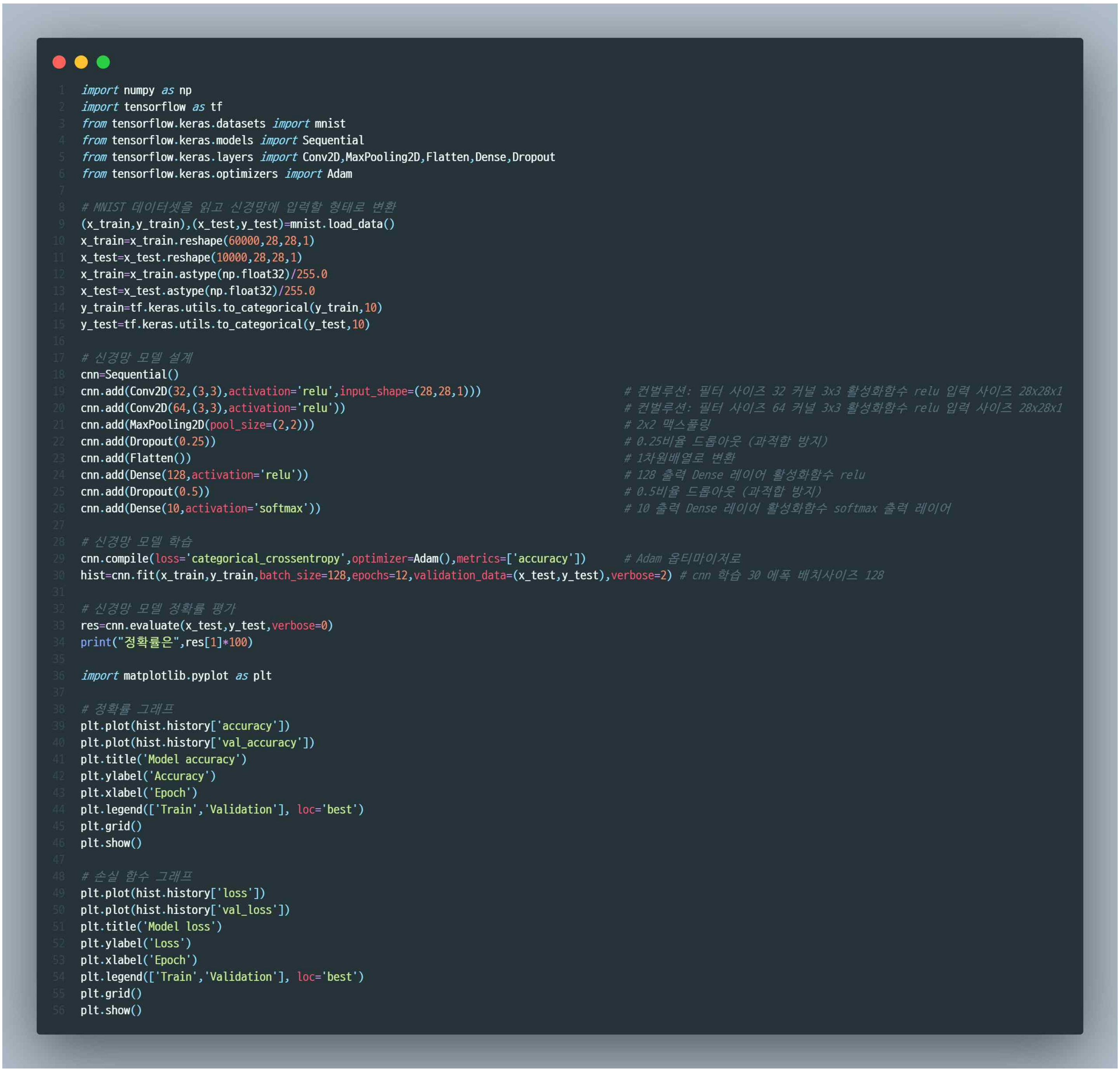
LeNet-5 구조로 C-P-C-P-C 형식으로 레이어를 구성 하였는데 결과를 보았더니 과적합이 발생 하였습니다.



```python
import numpy as np
import tensorflow as tf
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D,MaxPooling2D,Flatten,Dense
from tensorflow.keras.optimizers import Adam

# MNIST 데이터셋을 읽고 신경망에 입력할 형태로 변환
(x_train,y_train),(x_test,y_test)= mnist.load_data()
x_train=x_train.reshape(60000,28,28,1)
x_test=x_test.reshape(10000,28,28,1)
x_train=x_train.astype(np.float32)/255.0
x_test=x_test.astype(np.float32)/255.0
y_train=tf.keras.utils.to_categorical(y_train,10)
y_test=tf.keras.utils.to_categorical(y_test,10)

# LeNet-5 신경망 모델 설계
cnn=Sequential()
cnn.add(Conv2D(6,(5,5),padding='same',activation='relu',input_shape=(28,28,1)))        # 컨벌루션: 필터 사이즈 6 커널 5x5 활성화함수 relu 입력 사이즈 28x28x1
cnn.add(MaxPooling2D(pool_size=(2,2)))                                                  # 2x2 맥스풀링
cnn.add(Conv2D(16,(5,5),padding='same',activation='relu'))                              # 컨벌루션: 필터 사이즈 16 커널 5x5 활성화함수 relu
cnn.add(MaxPooling2D(pool_size=(2,2)))                                                  # 2x2 맥스풀링
cnn.add(Conv2D(120,(5,5),padding='same',activation='relu'))                             # 컨벌루션: 필터 사이즈 120 커널 5x5 활성화함수 relu
cnn.add(Flatten())                                                                      # 1차원배열로 변환
cnn.add(Dense(84,activation='relu'))                                                    # 84 출력 Dense 레이어 활성화함수 relu
cnn.add(Dense(10,activation='softmax'))                                                 # 10 출력 Dense 레이어 활성화함수 softmax 출력 레이어

# 신경망 모델 학습
cnn.compile(loss='categorical_crossentropy',optimizer=Adam(),metrics=['accuracy'])    # Adam 옵티마이저로
hist=cnn.fit(x_train,y_train,batch_size=128,epochs=30,validation_data=(x_test,y_test),verbose=2) # cnn 학습 30 에폭 배치사이즈 128

# 신경망 모델 정확률 평가
res=cnn.evaluate(x_test,y_test,verbose=0)
print("정확률은",res[1]*100)

import matplotlib.pyplot as plt

# 정확률 그래프
plt.plot(hist.history['accuracy'])
plt.plot(hist.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train','Validation'], loc='best')
plt.grid()
plt.show()

# 손실 함수 그래프
plt.plot(hist.history['loss'])
plt.plot(hist.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train','Validation'], loc='best')
plt.grid()
plt.show()
```

6-2.py

C-C-P 구조로 레이어를 구성하고 드롭아웃을 사용하여 과적합을 줄였습니다.



```python
import numpy as np
import tensorflow as tf
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D,MaxPooling2D,Flatten,Dense,Dropout
from tensorflow.keras.optimizers import Adam

# MNIST 데이터셋을 읽고 신경망에 입력할 형태로 변환
(x_train,y_train),(x_test,y_test)=mnist.load_data()
x_train=x_train.reshape(60000,28,28,1)
x_test=x_test.reshape(10000,28,28,1)
x_train=x_train.astype(np.float32)/255.0
x_test=x_test.astype(np.float32)/255.0
y_train=tf.keras.utils.to_categorical(y_train,10)
y_test=tf.keras.utils.to_categorical(y_test,10)

# 신경망 모델 설계
cnn=Sequential()
cnn.add(Conv2D(32,(3,3),activation='relu',input_shape=(28,28,1)))                       # 컨벌루션: 필터 사이즈 32 커널 3x3 활성화함수 relu 입력 사이즈 28x28x1
cnn.add(Conv2D(64,(3,3),activation='relu'))                                             # 컨벌루션: 필터 사이즈 64 커널 3x3 활성화함수 relu 입력 사이즈 28x28x1
cnn.add(MaxPooling2D(pool_size=(2,2)))                                                  # 2x2 맥스풀링
cnn.add(Dropout(0.25))                                                                  # 0.25비율 드롭아웃 (과적합 방지)
cnn.add(Flatten())                                                                      # 1차원배열로 변환
cnn.add(Dense(128,activation='relu'))                                                   # 128 출력 Dense 레이어 활성화함수 relu
cnn.add(Dropout(0.5))                                                                   # 0.5비율 드롭아웃 (과적합 방지)
cnn.add(Dense(10,activation='softmax'))                                                 # 10 출력 Dense 레이어 활성화함수 softmax 출력 레이어

# 신경망 모델 학습
cnn.compile(loss='categorical_crossentropy',optimizer=Adam(),metrics=['accuracy'])      # Adam 옵티마이저로
hist=cnn.fit(x_train,y_train,batch_size=128,epochs=12,validation_data=(x_test,y_test),verbose=2) # cnn 학습 30 에폭 배치사이즈 128

# 신경망 모델 정확률 평가
res=cnn.evaluate(x_test,y_test,verbose=0)
print("정확률은",res[1]*100)

import matplotlib.pyplot as plt

# 정확률 그래프
plt.plot(hist.history['accuracy'])
plt.plot(hist.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train','Validation'], loc='best')
plt.grid()
plt.show()

# 손실 함수 그래프
plt.plot(hist.history['loss'])
plt.plot(hist.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train','Validation'], loc='best')
plt.grid()
plt.show()
```