● 교통표지판 다중분류 인공지능 (코드 전문 : https://url.kr/5k6zbi)

1. 개발동기

AI와 머신러닝 과목에서 배운 내용을 토대로 기말 프로젝트를 진행하게 되었습니다. 자율주행 자동차에서 실제로 사용하고 있는 인공지능이고 배운 내용을 적절히 응용하여 만들 수 있을 것이라 생각하여 교통표지판을 분류하는 인공지능을 개발하기로 결정하였습니다.

2. 프로그램 설명

python과 keras를 이용하여 교통 표지판이 이미지 형태로 입력되면 어떤 표지판인지 분류해내는 CNN모델을 만들었습니다. 모델을 학습시키는데 필요한 데이터셋은 kaggle에서 구하였고 CNN모델의 하이퍼파라미터 중 커널의 개수와 Dropout 값을 조금씩 변경한 여러 개의 모델을 모두 학습시킨 뒤 가장 정확도가 높은 모델을 찾았습니다.

3. 중요 포인트

합성곱 신경망1 kernel(16-32-64), Dropout 0.3

```
from tensorflow import keras
model=keras.Sequential()

model.add(keras.layers.Conv2D(16, kernel_size=3, activatio
model.add(keras.layers.MaxPooling2D(2))
model.add(keras.layers.Conv2D(32, kernel_size=3, activatio
model.add(keras.layers.MaxPooling2D(2))
model.add(keras.layers.Conv2D(64, kernel_size=3, activatio
model.add(keras.layers.MaxPooling2D(2))
```

```
model.add(keras.layers.Flatten())
model.add(keras.layers.Dense(100, activation='relu'))
model.add(keras.layers.Dropout(0.3))
model.add(keras.layers.Dense(43, activation='softmax'))
```

합성곱 신경망1 평가

```
model.evaluate(test_scaled, test_target)
```

```
395/395 [==============================] - 1s
[0.23808132112026215, 0.9542359709739685]
```

합성곱신경망2 kernel(16-32-64), Dropout 0.4

```
model2=keras.Sequential()

model2.add(keras.layers.Conv2D(16, kernel_size=3, activati
model2.add(keras.layers.MaxPooling2D(2))
model2.add(keras.layers.Conv2D(32, kernel_size=3, activati
model2.add(keras.layers.MaxPooling2D(2))
model2.add(keras.layers.Conv2D(64, kernel_size=3, activati
model2.add(keras.layers.MaxPooling2D(2))
```

```
model2.add(keras.layers.Flatten())
model2.add(keras.layers.Dense(100, activation='relu'))
model2.add(keras.layers.Dropout(0.4))
model2.add(keras.layers.Dense(43, activation='softmax'))
model2.summary()
```

합성곱신경망2 평가

```
model2.evaluate(test_scaled, test_target)
```

```
395/395 [==============================] - 1s 3m
[0.38210660219192505, 0.9208234548568726]
```

합성곱신경망3 kernel(32-64), Dropout 0.3

```
model3=keras.Sequential()

model3.add(keras.layers.Conv2D(32, kernel_size=3, activat
model3.add(keras.layers.MaxPooling2D(2))
model3.add(keras.layers.Conv2D(64, kernel_size=3, activat
model3.add(keras.layers.MaxPooling2D(2))
```

```
model3.add(keras.layers.Flatten())
model3.add(keras.layers.Dense(100, activation='relu'))
model3.add(keras.layers.Dropout(0.3))
model3.add(keras.layers.Dense(43, activation='softmax'))
model3.summary()
```

합성곱신경망3 평가

```
model3.evaluate(test_scaled, test_target)
```

```
395/395 [==============================] - 2s
[0.21652837097644806, 0.9516231417655945]
```

합성곱신경망4 kernel(32-64), Dropout 0.4

```
model4=keras.Sequential()

model4.add(keras.layers.Conv2D(32, kernel_size=3, activat
model4.add(keras.layers.MaxPooling2D(2))
model4.add(keras.layers.Conv2D(64, kernel_size=3, activat
model4.add(keras.layers.MaxPooling2D(2))
```

```
model4.add(keras.layers.Flatten())
model4.add(keras.layers.Dense(100, activation='relu'))
model4.add(keras.layers.Dropout(0.4))
model4.add(keras.layers.Dense(43, activation='softmax'))
model4.summary()
```

합성곱신경망4 평가

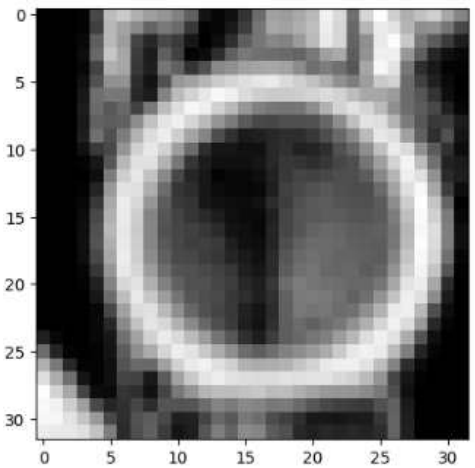
```
model4.evaluate(test_scaled, test_target)
```

```
395/395 [==============================] - 1s
[0.17391511797904968, 0.9617577195167542]
```

커널의 개수를 2개 혹은 3개, Dropout 값을 0.3 혹은 0.4로 설정한 모델을 4개 만들어 학습시킨 뒤 정확도를 평가하자 커널 2개, Dropout 0.4 모델의 정확도가 96.17%로 가장 높은 것을 확인하였습니다.

4. 프로그램 구현 모습

```
plt.imshow(val_scaled[4].reshape(32, 32), cmap='gray_r')
plt.show()
```




```
preds4=model.predict(val_scaled[4:5])
print(preds4)
```

```
1/1 [==============================] - 0s 19ms/step
[[6.2311398e-30 9.9079983e-23 1.9334176e-17 2.4420483e-12 4.0972613e-18
  5.3751987e-18 0.0000000e+00 1.4026244e-29 9.1405190e-26 2.5363018e-13
  1.4470102e-22 1.0397437e-31 8.7779599e-17 1.9035075e-14 1.9572150e-18
  1.0000000e+00 0.0000000e+00 4.7224965e-30 1.2319633e-18 2.4477893e-34
  1.2616204e-34 4.5927928e-34 1.2207761e-28 9.4758216e-33 6.7201767e-35
  2.0117057e-21 1.0553760e-24 5.7453669e-38 1.3545232e-34 9.4481194e-29
  0.0000000e+00 7.7554024e-22 2.0223727e-26 2.9520211e-28 3.1981500e-27
  9.2540411e-21 1.9565092e-22 2.7932526e-37 1.4317059e-17 1.2876339e-25
  1.0521305e-27 2.5546781e-32 0.0000000e+00]]
```

```
print(label_names[np.argmax(preds4)])
```

```
[15 'No vehicles']
```

위 표지판을 입력시켰더니 No vehicles이라고 정확하게 분류해 냈습니다.