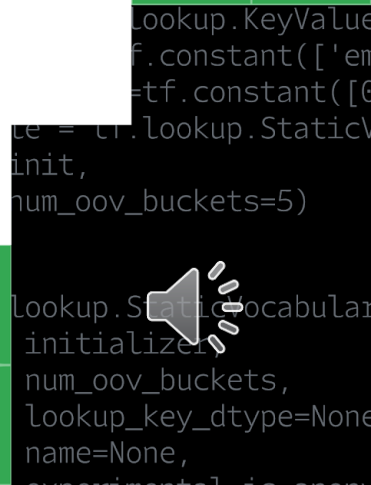
Google Developer Student Clubs

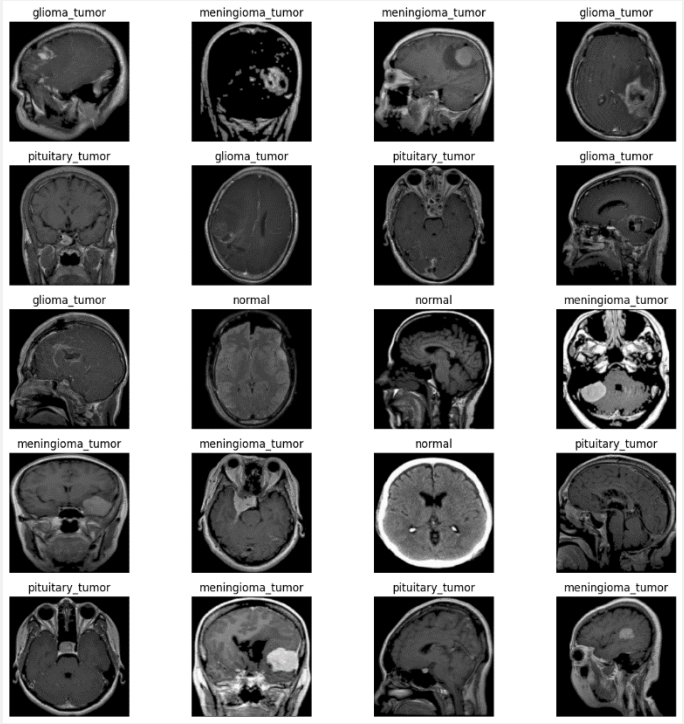
Mini Project Week03

Basic 1조: 이재승, 서지현



프로젝트 소개 및 목표

Goal



1. Glioma Tumor
2. Meningioma Tumor
3. Normal
4. Pituitary Tumor

평가 지표

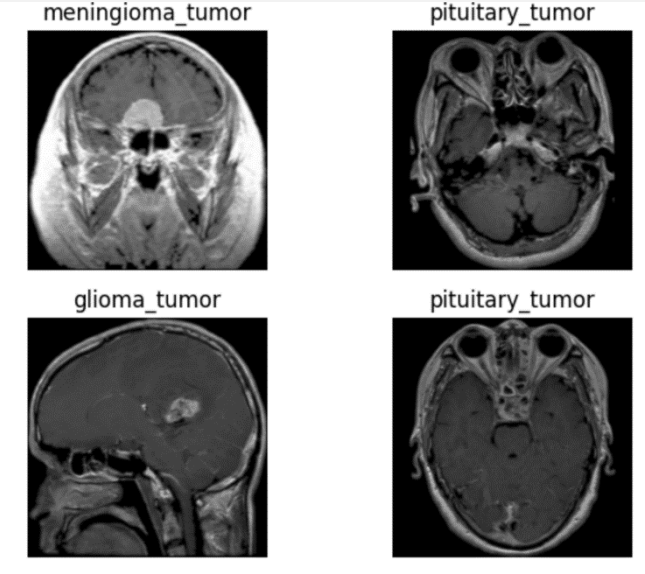
	precision	recall	f1-score	support
0	0.96	0.86	0.91	184
1	0.83	0.92	0.87	179
2	0.96	0.98	0.97	81
3	0.94	0.94	0.94	175
accuracy			0.91	619
macro avg	0.92	0.92	0.92	619
weighted avg	0.92	0.91	0.91	619

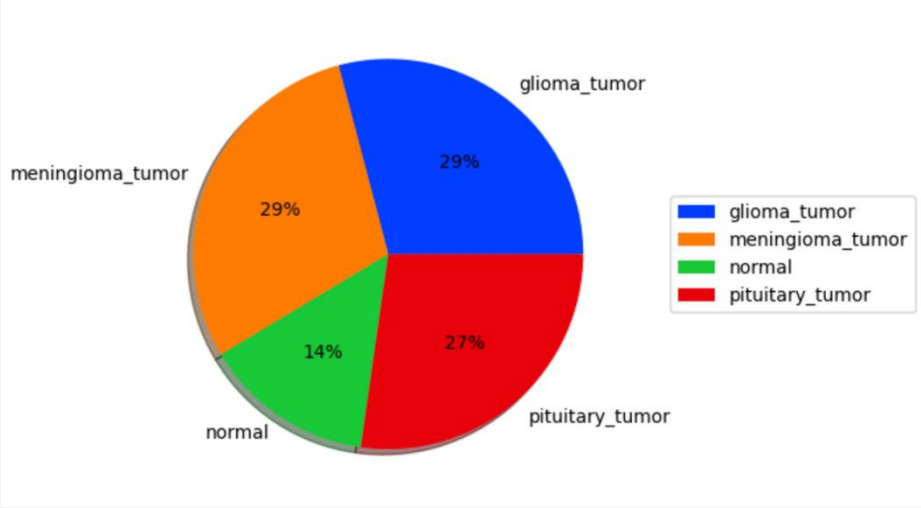
$$(Recall) = \frac{TP}{TP + FN}$$

		실제 정답	
		True	False
분류 결과	True	True Positive	False Positive
	False	False Negative	True Negative

데이터셋과 전처리

Dataset


meningioma_tumor
pituitary_tumor
glioma_tumor
pituitary_tumor


glioma_tumor
meningioma_tumor
normal
pituitary_tumor
29%
29%
14%
27%
glioma_tumor
meningioma_tumor
normal
pituitary_tumor

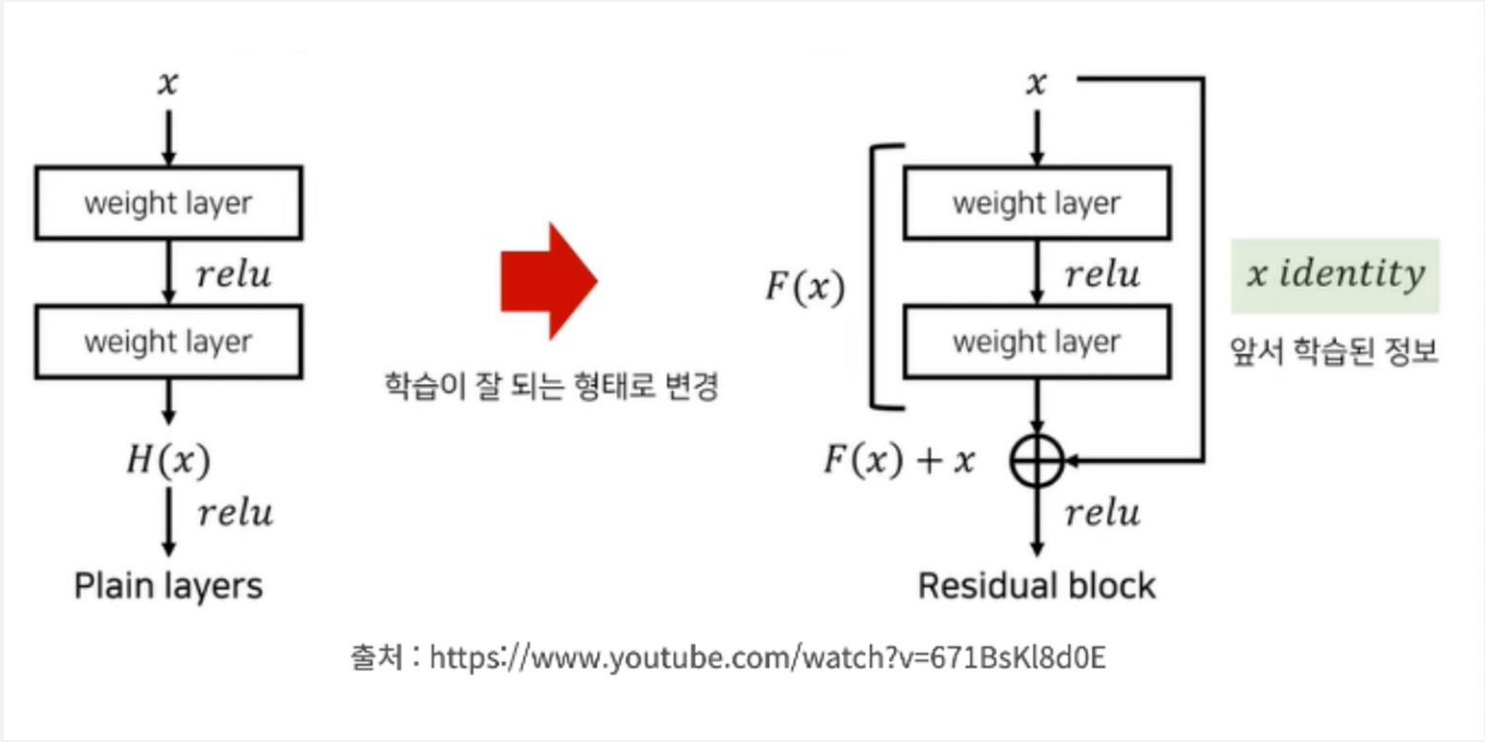
Data Augmentation

```
data_augmentation = tf.keras.Sequential(
    [
        tf.keras.layers.experimental.preprocessing.RandomFlip("horizontal"),
        tf.keras.layers.experimental.preprocessing.RandomRotation(0.1),
        tf.keras.layers.experimental.preprocessing.RandomContrast(0.1),
    ]
)
```

- Keras API를 사용하여 Flip, Rotation, Contrast를 진행.
- Sequential에 정의되어 있어, 미리 데이터를 증식하는게 아니라, 데이터 input이 들어올 때 증식 과정을 거침

모델 설계 및 개발 과정

ResNet50



출처 : https://www.youtube.com/watch?v=671BsKl8d0E

사용한 기법

- Data Augmentation
- Batch Normalization
- Bayesian Optimization
- Relu
- Dropout
- Callback

Batch Normalization & Relu

```
x = tf.keras.layers.Flatten()(x)
x = tf.keras.layers.Dense(512, activation=None)(x)
x = tf.keras.layers.BatchNormalization()(x)  # BatchNormalization 추가
x = tf.keras.layers.Activation('relu')(x)   # 활성화 함수 적용
x = tf.keras.layers.Dropout(0.5)(x)
```

Bayesian Optimization

```
space = {
    'lr': hp.choice('lr', [0.1, 0.01, 0.001, 0.0001]),
    'units': hp.choice('units', [256, 512, 1024]),
    'dropout_rate': hp.uniform('dropout_rate', 0.3, 0.7)
}
```

- Bayesian Optimization: 사전정보를 반영하여 최적의 하이퍼파라미터를 찾는 방법
- lr = 0.1,units = 512, dropout_rate = 0.47로 결정
  - hp.choice: 후보들 중 하나의 값 선택
  - hp.uniform: 범위 내의 실수값 선택

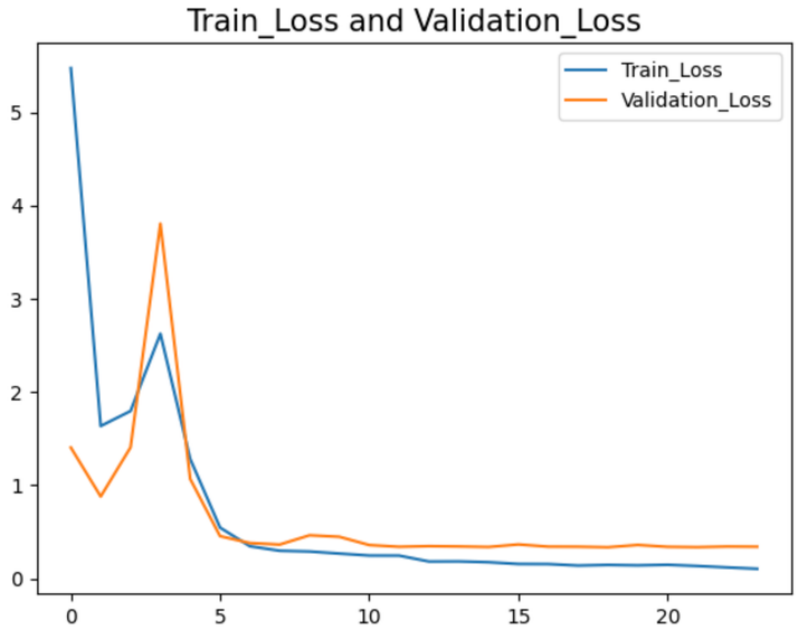
Callback

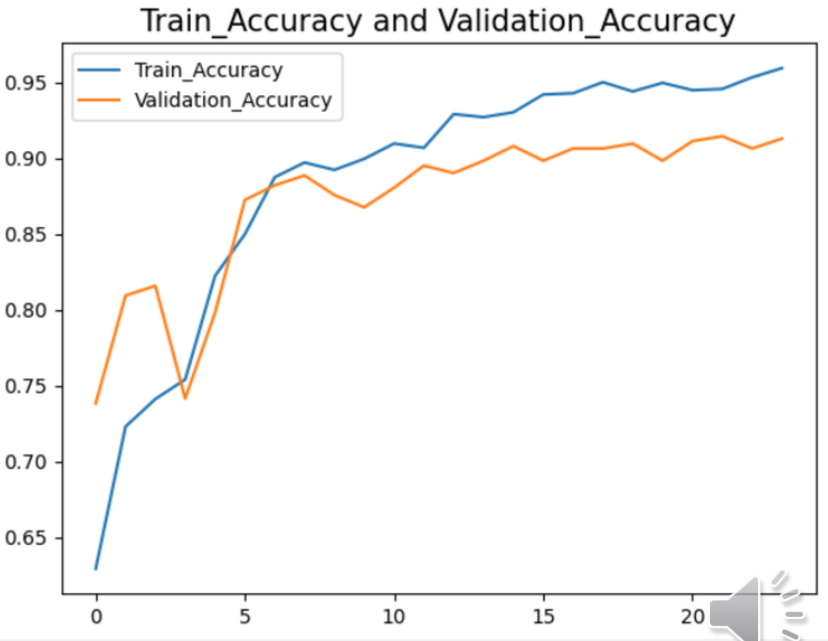
```
learning_rate_reduction = tf.keras.callbacks.ReduceLROnPlateau(
    monitor="val_loss", patience=2, factor=0.5, min_lr=0.00001, verbose=1)
Early_Stopping = tf.keras.callbacks.EarlyStopping(
    monitor="val_loss", patience=5, restore_best_weights=True)
```

- Early Stopping: val_loss가 개선되지 않으면 학습을 조기 중단
- Reduce Learning rate: val_loss가 개선되지 않으면 학습률 감소

결과 및 평가

Loss & Accuracy



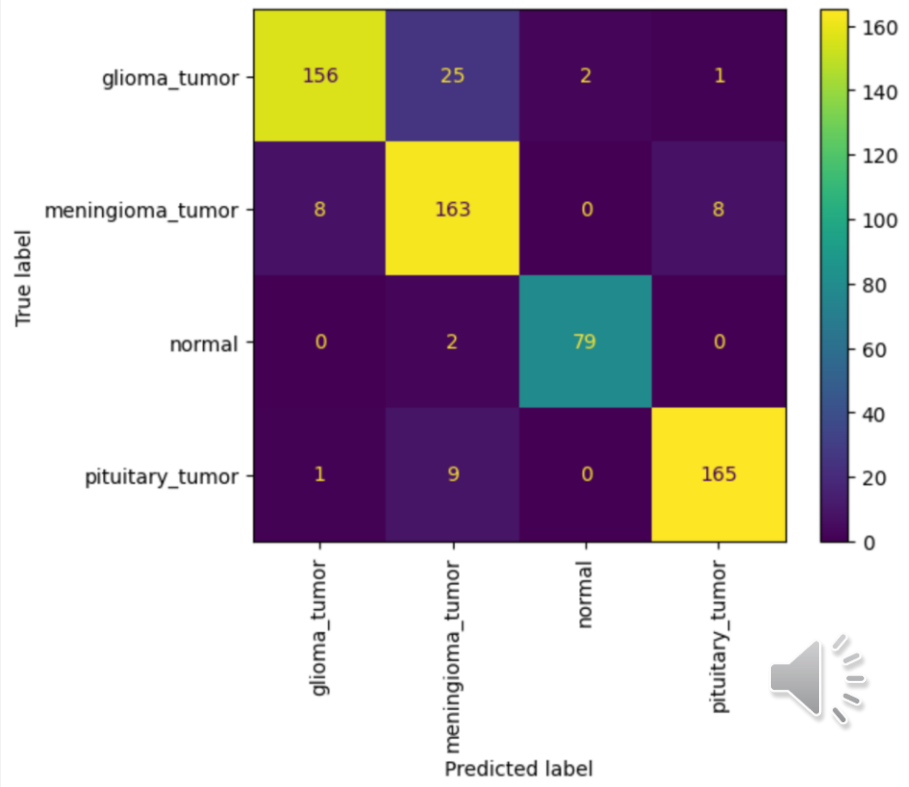


Recall: 0.9220

```
F1-Score:[0.9193210129217769]
Precision:[0.9191515992321595]
Recall:[0.9220336317214386]
Classification_Report
----------------------
              precision    recall  f1-score   support

           0       0.95      0.85      0.89       184
           1       0.82      0.91      0.86       179
           2       0.98      0.98      0.98        81
           3       0.95      0.94      0.95       175

    accuracy                           0.91       619
   macro avg       0.92      0.92      0.92       619
weighted avg       0.91      0.91      0.91       619
```



결론 및 느낀점

결론

- Recall: 0.9220 (base code보다 0.02향상)
- 성능 개선에 영향을 미친 방법
  - Batch Normalization
  - Bayesian optimization

느낀점

- 이미 점수가 높은 베이스코드의 성능을 더 좋게 개선하는 것이 매우 어려웠다.
- 데이터셋의 특성을 이해하고 그 특성에 맞는 모델과 기법을 사용하는 것이 중요함을 깨달았다.
- 문제 특성에 따라 accuracy가 아닌 다른 평가 지표도 주의깊게 살펴볼 필요가 있음을 알 수 있었다.