ViT Transformer Model X EfficientNetB4 Hybrid Model Observation and Analysis

- *First Training Test*

Parameter	Value
Learning Rate	0.00001
Epochs	10
Rotation Range	40
Width Shift Range	0.4
Height Shift Range	0.4
Shear Range	0.3
Zoom Range	0.4
Vertical Flip	True

- *Result and Observation*

Epoch	Step	Time	Loss	Accuracy	Val Loss	Val Accuracy
1	179/179	216s	1.2683	0.4304	1.3655	0.3362
2	179/179	145s	0.9244	0.6577	1.2519	0.3386
3	179/179	145s	0.6511	0.7668	1.0321	0.5483
4	179/179	145s	0.5333	0.7971	1.3233	0.3252

Maximum Validation-Accuracy Obtained: 54.83%

Validation-Loss: 1.0321

- *Second Training Test*

Parameter	Value
Learning Rate	0.0001
Epochs	30
Rotation Range	20
Width Shift Range	0.2
Height Shift Range	0.2
Shear Range	0.2
Zoom Range	0.2
Vertical Flip	False

- *Result and Observation*

Epoch	Step	Time	Loss	Accuracy	Val Loss	Val Accuracy
1	179/179	216s	0.4330	0.8381	1.5094	0.4030
2	179/179	144s	0.1641	0.9425	1.8209	0.3197
3	179/179	146s	0.1064	0.9637	2.3185	0.3370
4	179/179	144s	0.0690	0.9772	3.5033	0.3181
5	179/179	145s	0.0593	0.9814	0.6580	0.7439
6	179/179	149s	0.0528	0.9820	1.2787	0.5734

Epoch	Step	Time	Loss	Accuracy	Val Loss	Val Accuracy
7	179/179	145s	0.0346	0.9895	3.1254	0.3409
8	179/179	145s	0.0346	0.9884	0.3528	0.8775
9	179/179	144s	0.0316	0.9904	2.1506	0.4580
10	179/179	145s	0.0295	0.9895	1.6275	0.5020
11	179/179	145s	0.0159	0.9946	2.1292	0.4996
12	179/179	145s	0.0257	0.9916	6.0443	0.3559
13	179/179	145s	0.0198	0.9940	0.4916	0.8523

Maximum Validation-Accuracy Obtained: 87.75%

Validation-Loss: 0.3528

- *Third Training Test*

Parameter	Value
Learning Rate	0.00001
Epochs	30
Rotation Range	20
Width Shift Range	0.2
Height Shift Range	0.2
Shear Range	0.2
Zoom Range	0.2
Vertical Flip	False
Checkpoint Included	**Yes**

- ***Result and Observation***

Epoch	Step	Time	Loss	Accuracy	Val Loss	Val Accuracy
1	179/179	218s	1.1431	0.5464	1.3923	0.3134
2	179/179	150s	0.6690	0.7873	1.3529	0.3260
3	179/179	152s	0.4305	0.8562	0.8985	0.5797
4	179/179	150s	0.3240	0.8924	0.7187	0.7164
5	179/179	153s	0.2700	0.9047	0.5782	0.7958
6	179/179	144s	0.2347	0.9161	1.5516	0.3708
7	179/179	151s	0.1970	0.9306	0.5100	0.8154
8	179/179	145s	0.1872	0.9357	0.7193	0.7258
9	179/179	145s	0.1633	0.9425	0.8410	0.6096
10	179/179	156s	0.1512	0.9455	0.3125	0.8940
11	179/179	156s	0.1336	0.9559	0.2679	0.9144
12	179/179	145s	0.1291	0.9541	0.4792	0.8044
13	**179/179**	**156s**	**0.1099**	**0.9609**	**0.1809**	**0.9403**
14	179/179	145s	0.1160	0.9571	0.4956	0.7973
15	179/179	144s	0.0977	0.9655	0.2211	0.9199
16	179/179	145s	0.0986	0.9657	0.2903	0.8892
17	179/179	145s	0.0852	0.9711	0.4203	0.8358
18	179/179	145s	0.0896	0.9702	0.3545	0.8578
Test	***40/40***	***10s***	***0.1809***	***0.9403***		

**Note: The Early Stopping Mechanism stopped the Model at 18/30 Epoche for the model*

Maximum Validation-Accuracy Obtained: 94.03%

Validation-Loss: 0.1809

- *Fourth Training Test*

Parameter	Value
Epochs	30
Learning Rate	0.0001
Reduce LR Included	***Yes***
Reduce LR Details	***monitor='val_loss', factor=0.2, patience=3, min_lr=1e-6***

- *Result and Observation*

Epoch	Train Loss	Train Accuracy	Val Loss	Val Accuracy	Learning Rate
1	1.6980	0.8367	2.6975	0.3181	1.0000e-04
2	1.3018	0.9445	2.4951	0.3181	1.0000e-04
3	1.1099	0.9650	1.5460	0.7502	1.0000e-04
4	0.9592	0.9769	2.6085	0.4619	1.0000e-04
5	0.8246	0.9807	4.5175	0.3221	1.0000e-04
6	0.7133	0.9811	0.8404	0.9293	1.0000e-04
7	0.5947	0.9870	4.1597	0.3551	1.0000e-04
8	0.5039	0.9900	2.6479	0.4297	1.0000e-04
9	0.4251	0.9912	0.7380	0.8845	1.0000e-04
10	0.3613	0.9888	3.6102	0.3480	1.0000e-04

Epoch	Train Loss	Train Accuracy	Val Loss	Val Accuracy	Learning Rate
11	0.2908	0.9947	3.1961	0.4234	1.0000e-04
12	0.2390	0.9958	1.1449	0.7298	1.0000e-04
13	0.2105	0.9954	0.4991	0.9002	2.0000e-05
14	0.1983	0.9965	0.2667	0.9709	2.0000e-05
15	0.1903	0.9965	0.3599	0.9419	2.0000e-05
16	0.1806	0.9970	0.3404	0.9411	2.0000e-05
17	0.1697	0.9974	0.1882	0.9890	2.0000e-05
18	0.1592	0.9977	0.2454	0.9670	2.0000e-05
19	0.1500	0.9974	0.4338	0.9010	2.0000e-05
20	0.1422	0.9975	1.1522	0.6826	2.0000e-05
21	0.1357	0.9979	0.1469	0.9945	4.0000e-06
22	0.1342	0.9984	0.1464	0.9937	4.0000e-06
23	0.1304	0.9982	0.1362	0.9969	4.0000e-06
24	0.1277	0.9989	0.1352	0.9961	4.0000e-06
25	0.1283	0.9984	0.1822	0.9796	4.0000e-06
26	**0.1261**	**0.9981**	**0.1313**	**0.9976**	**4.0000e-06**
27	0.1235	0.9981	0.1306	0.9961	4.0000e-06
28	0.1198	0.9988	0.1346	0.9945	4.0000e-06
29	0.1183	0.9984	0.1306	0.9953	4.0000e-06
30	0.1162	0.9981	0.1225	0.9969	4.0000e-06
Test	***0.1313***	***0.9976***	-	-	-

Maximum Validation-Accuracy Obtained: 99.76%

Validation-Loss: 0.1313

- *Intuition or Insights*

  Learning Rate: The change in learning rate from 0.0001 to 0.00001 can have a profound impact on the training dynamics. The lower learning rate allows for more gradual updates to the model's weights, potentially leading to better convergence.

  ReduceLROnPlateau Callback: This callback helps in adjusting the learning rate dynamically based on the validation loss. If the validation loss plateaus, the learning rate is reduced by a factor (in this case, 0.2). This can help the model converge better by reducing the learning rate when necessary.

  Model Initialization and Randomness: Neural networks are initialized with random weights, and the training process involves stochastic processes (e.g., shuffling of data, dropout). These random elements can lead to different training outcomes even with the same model and data.

  Validation Performance: The new training results show better validation performance consistently across epochs. This indicates that the model is generalizing better to the validation data, which likely translates to better test performance as well.

- *Screenshot of Output:*

```
[ ]
179/179 [==============================] - 153s 853ms/step - loss: 0.1697 - accuracy: 0.9974 - val_loss: 0.1882 - val_accuracy: 0.9890 - lr: 2.0000e-05
Epoch 18/30
179/179 [==============================] - 145s 810ms/step - loss: 0.1592 - accuracy: 0.9977 - val_loss: 0.2454 - val_accuracy: 0.9670 - lr: 2.0000e-05
Epoch 19/30
179/179 [==============================] - 145s 809ms/step - loss: 0.1500 - accuracy: 0.9974 - val_loss: 0.4338 - val_accuracy: 0.9010 - lr: 2.0000e-05
Epoch 20/30
179/179 [==============================] - 145s 810ms/step - loss: 0.1422 - accuracy: 0.9975 - val_loss: 1.1522 - val_accuracy: 0.6826 - lr: 2.0000e-05
Epoch 21/30
179/179 [==============================] - 152s 850ms/step - loss: 0.1357 - accuracy: 0.9979 - val_loss: 0.1469 - val_accuracy: 0.9945 - lr: 4.0000e-06
Epoch 22/30
179/179 [==============================] - 145s 810ms/step - loss: 0.1342 - accuracy: 0.9984 - val_loss: 0.1464 - val_accuracy: 0.9937 - lr: 4.0000e-06
Epoch 23/30
179/179 [==============================] - 152s 846ms/step - loss: 0.1304 - accuracy: 0.9982 - val_loss: 0.1362 - val_accuracy: 0.9969 - lr: 4.0000e-06
Epoch 24/30
179/179 [==============================] - 145s 811ms/step - loss: 0.1277 - accuracy: 0.9989 - val_loss: 0.1352 - val_accuracy: 0.9961 - lr: 4.0000e-06
Epoch 25/30
179/179 [==============================] - 145s 810ms/step - loss: 0.1283 - accuracy: 0.9984 - val_loss: 0.1822 - val_accuracy: 0.9796 - lr: 4.0000e-06
Epoch 26/30
179/179 [==============================] - 152s 845ms/step - loss: 0.1261 - accuracy: 0.9981 - val_loss: 0.1313 - val_accuracy: 0.9976 - lr: 4.0000e-06
Epoch 27/30
179/179 [==============================] - 144s 802ms/step - loss: 0.1235 - accuracy: 0.9981 - val_loss: 0.1306 - val_accuracy: 0.9961 - lr: 4.0000e-06
Epoch 28/30
179/179 [==============================] - 144s 804ms/step - loss: 0.1198 - accuracy: 0.9988 - val_loss: 0.1346 - val_accuracy: 0.9945 - lr: 4.0000e-06
Epoch 29/30
179/179 [==============================] - 144s 803ms/step - loss: 0.1183 - accuracy: 0.9984 - val_loss: 0.1306 - val_accuracy: 0.9953 - lr: 4.0000e-06
Epoch 30/30
179/179 [==============================] - 144s 805ms/step - loss: 0.1162 - accuracy: 0.9981 - val_loss: 0.1225 - val_accuracy: 0.9969 - lr: 4.0000e-06
40/40 [==============================] - 9s 167ms/step - loss: 0.1313 - accuracy: 0.9976
Test Loss: 0.1313374936580658, Test Accuracy: 0.9976433515548706
```

```python
# Define data augmentation parameters
datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,  # Reduced range
    width_shift_range=0.2,  # Reduced range
    height_shift_range=0.2,  # Reduced range
    shear_range=0.2,  # Reduced range
    zoom_range=0.2,  # Reduced range
    horizontal_flip=True,
    fill_mode='nearest'
)

# Load training data with augmentation
train_generator = datagen.flow_from_directory(
    '/content/Dataset Brain Tumor/Dataset Brain Tumor/Training Dataset',
    target_size=(240, 240),
    batch_size=32,
    class_mode='categorical'
)

# Load testing data without augmentation
test_datagen = ImageDataGenerator(rescale=1./255)
test_generator = test_datagen.flow_from_directory(
    '/content/Dataset Brain Tumor/Dataset Brain Tumor/Testing Dataset',
    target_size=(240, 240),
    batch_size=32,
    class_mode='categorical'
)

# Load the base model
base_model = EfficientNetB4(weights='imagenet', include_top=False, input_shape=(240, 240, 3))

# Add custom layers on top of the base model
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(1024, activation='relu', kernel_regularizer=l2(0.001))(x)  # Added L2 regularization
x = Dropout(0.5)(x)  # Apply dropout
x = Dense(4, activation='softmax')(x)  # Output layer for 4 classes

# Create the final model
model = Model(inputs=base_model.input, outputs=x)

# Compile the model
model.compile(optimizer=Adam(learning_rate=0.0001), loss='categorical_crossentropy', metrics=['accuracy'])

# Define callbacks
early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)
checkpoint = ModelCheckpoint('best_model.h5', monitor='val_accuracy', save_best_only=True)
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=3, min_lr=1e-6)

# Train the model
history = model.fit(
    train_generator,
    epochs=30,
    validation_data=test_generator,
    callbacks=[early_stopping, checkpoint, reduce_lr]
)
```

- *Fifth Training Test:*

Performed a Cross-Validation Test for Fourth Training Test (Accuracy of 99.76%) and also included other metrics like Fscore, Recall, Precision, AUC to get a better Picture

(Rest Parameters remains same)

- *Result and Observation*

Epoch	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy	Learning Rate
1	1.7054	0.8320	3.4837	0.3181	1.0000e-04
2	1.2887	0.9441	4.0600	0.3181	1.0000e-04
3	1.1112	0.9620	2.2104	0.4941	1.0000e-04
4	0.9447	0.9762	2.3074	0.5609	1.0000e-04
5	0.8225	0.9783	3.3032	0.3896	1.0000e-04
6	0.6981	0.9832	3.3179	0.4375	1.0000e-04
7	0.6169	0.9911	0.9634	0.8445	2.0000e-05
8	0.5903	0.9933	0.9736	0.8460	2.0000e-05
9	0.5658	0.9932	1.8827	0.6316	2.0000e-05
10	0.5388	0.9947	0.5707	0.9811	2.0000e-05
11	0.5188	0.9937	0.6859	0.9332	2.0000e-05
12	0.4969	0.9942	1.7978	0.6402	2.0000e-05
13	0.4740	0.9944	1.9290	0.6339	2.0000e-05
14	0.4568	0.9953	0.6024	0.9466	4.0000e-06
15	0.4489	0.9960	0.4563	0.9921	4.0000e-06
16	0.4456	0.9960	0.4532	0.9929	4.0000e-06
17	0.4383	0.9960	0.4448	0.9953	4.0000e-06

Epoch	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy	Learning Rate
18	0.4344	0.9953	0.4384	0.9953	4.0000e-06
19	0.4275	0.9960	0.4509	0.9882	4.0000e-06
20	0.4221	0.9961	0.5025	0.9654	4.0000e-06
21	0.4145	0.9967	0.4322	0.9914	4.0000e-06
22	0.4123	0.9958	0.4232	0.9906	4.0000e-06
23	0.4044	0.9963	0.4977	0.9576	4.0000e-06
24	0.3961	0.9965	0.4400	0.9835	4.0000e-06
25	0.3895	0.9970	0.5116	0.9513	4.0000e-06
26	**0.3872**	**0.9968**	**0.3910**	**0.9969**	**1.0000e-06**
27	0.3904	0.9944	0.3915	0.9961	1.0000e-06
28	**0.3811**	**0.9972**	**0.3898**	**0.9969**	**1.0000e-06**
29	0.3823	0.9963	0.3871	0.9961	1.0000e-06
30	0.3809	0.9963	0.3856	0.9953	1.0000e-06

- *Classification Report:*

Class	Precision	Recall	F1-Score	Support
Glioma	1.00	1.00	1.00	262
Meningioma	0.99	0.99	0.99	306
No Tumor	1.00	1.00	1.00	405
Pituitary	0.99	1.00	1.00	300
Accuracy	**1.00**	**1.00**	**1.00**	**1273**
Macro Avg	1.00	1.00	1.00	1273
Weighted Avg	1.00	1.00	1.00	1273

- ***Test Results:***

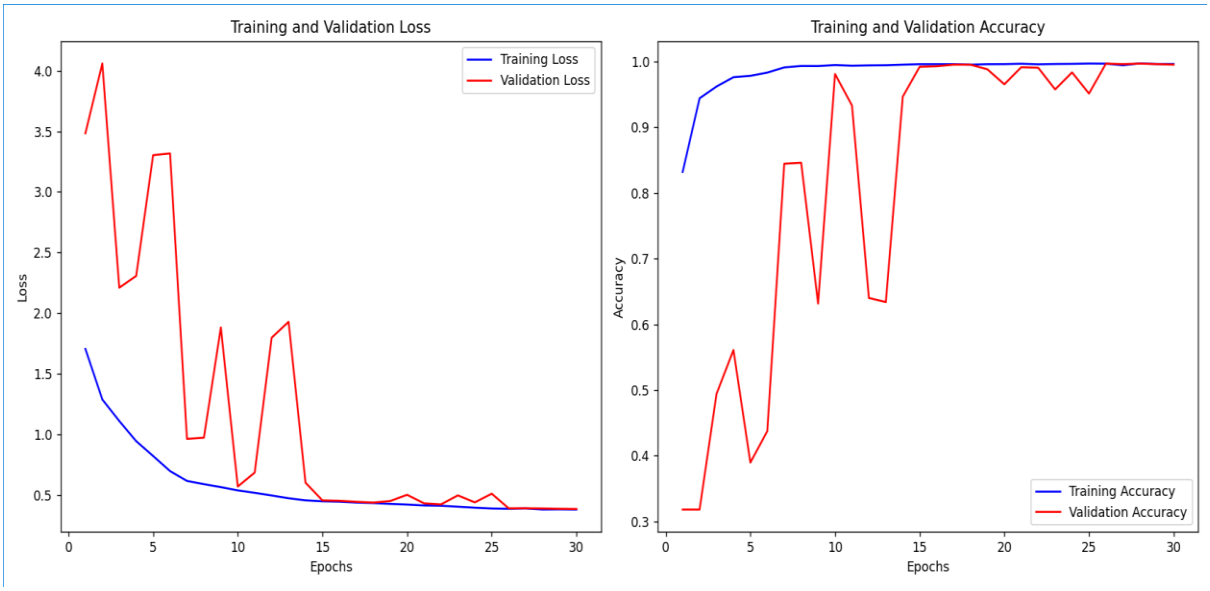
Metric	Value
Test Loss	0.3910
Test Accuracy	0.9969

- ***AUC Score per Class Results:***

Class	AUC Score
Glioma	0.9999886742020975
Meningioma	0.9995809423390176
No Tumor	1.0
Pituitary	0.9996848235697157

```
179/179 [==============================] - 149s 833ms/step - loss: 0.4489 - accuracy: 0.9960 - val_loss: 0.4563 - val_accuracy: 0.9921 - lr: 4.0000e-06
Epoch 16/30
179/179 [==============================] - 151s 844ms/step - loss: 0.4456 - accuracy: 0.9960 - val_loss: 0.4532 - val_accuracy: 0.9929 - lr: 4.0000e-06
Epoch 17/30
179/179 [==============================] - 152s 845ms/step - loss: 0.4383 - accuracy: 0.9960 - val_loss: 0.4448 - val_accuracy: 0.9953 - lr: 4.0000e-06
Epoch 18/30
179/179 [==============================] - 144s 802ms/step - loss: 0.4344 - accuracy: 0.9953 - val_loss: 0.4384 - val_accuracy: 0.9953 - lr: 4.0000e-06
Epoch 19/30
179/179 [==============================] - 144s 802ms/step - loss: 0.4275 - accuracy: 0.9960 - val_loss: 0.4509 - val_accuracy: 0.9882 - lr: 4.0000e-06
Epoch 20/30
179/179 [==============================] - 144s 802ms/step - loss: 0.4221 - accuracy: 0.9961 - val_loss: 0.5025 - val_accuracy: 0.9654 - lr: 4.0000e-06
Epoch 21/30
179/179 [==============================] - 144s 805ms/step - loss: 0.4145 - accuracy: 0.9967 - val_loss: 0.4322 - val_accuracy: 0.9914 - lr: 4.0000e-06
Epoch 22/30
179/179 [==============================] - 148s 823ms/step - loss: 0.4123 - accuracy: 0.9958 - val_loss: 0.4232 - val_accuracy: 0.9906 - lr: 4.0000e-06
Epoch 23/30
179/179 [==============================] - 144s 801ms/step - loss: 0.4044 - accuracy: 0.9963 - val_loss: 0.4977 - val_accuracy: 0.9576 - lr: 4.0000e-06
Epoch 24/30
179/179 [==============================] - 144s 801ms/step - loss: 0.3961 - accuracy: 0.9965 - val_loss: 0.4400 - val_accuracy: 0.9835 - lr: 4.0000e-06
Epoch 25/30
179/179 [==============================] - 144s 802ms/step - loss: 0.3895 - accuracy: 0.9970 - val_loss: 0.5116 - val_accuracy: 0.9513 - lr: 4.0000e-06
Epoch 26/30
179/179 [==============================] - 151s 843ms/step - loss: 0.3872 - accuracy: 0.9968 - val_loss: 0.3910 - val_accuracy: 0.9969 - lr: 1.0000e-06
Epoch 27/30
179/179 [==============================] - 144s 802ms/step - loss: 0.3904 - accuracy: 0.9944 - val_loss: 0.3915 - val_accuracy: 0.9961 - lr: 1.0000e-06
Epoch 28/30
179/179 [==============================] - 144s 803ms/step - loss: 0.3811 - accuracy: 0.9972 - val_loss: 0.3898 - val_accuracy: 0.9969 - lr: 1.0000e-06
Epoch 29/30
179/179 [==============================] - 144s 804ms/step - loss: 0.3823 - accuracy: 0.9963 - val_loss: 0.3871 - val_accuracy: 0.9961 - lr: 1.0000e-06
Epoch 30/30
179/179 [==============================] - 144s 802ms/step - loss: 0.3809 - accuracy: 0.9963 - val_loss: 0.3856 - val_accuracy: 0.9953 - lr: 1.0000e-06
40/40 [==============================] - 9s 167ms/step - loss: 0.3910 - accuracy: 0.9969
Test Loss: 0.3910380005836487, Test Accuracy: 0.9968578219413757
40/40 [==============================] - 9s 163ms/step
Classification Report:
              precision    recall  f1-score   support

      glioma       1.00      1.00      1.00       262
  meningioma       0.99      0.99      0.99       306
     notumor       1.00      1.00      1.00       405
   pituitary       0.99      1.00      1.00       300

    accuracy                           1.00      1273
   macro avg       1.00      1.00      1.00      1273
weighted avg       1.00      1.00      1.00      1273

AUC Scores per class: [0.9999886742020975, 0.9995809423390176, 1.0, 0.9996848235697157]
```

```
# Load the base model
base_model = EfficientNetB4(weights='imagenet', include_top=False, input_shape=(240, 240, 3))

# Add custom layers on top of the base model
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(1024, activation='relu', kernel_regularizer=l2(0.001))(x)
x = Dropout(0.5)(x)
x = Dense(4, activation='softmax')(x)

# Create the final model
model = Model(inputs=base_model.input, outputs=x)

# Compile the model
model.compile(optimizer=Adam(learning_rate=0.0001), loss='categorical_crossentropy', metrics=['accuracy'])

# Define callbacks
early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)
checkpoint = ModelCheckpoint('best_model.h5', monitor='val_accuracy', save_best_only=True)
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=3, min_lr=1e-6)

# Train the model
history = model.fit(
    train_generator,
    epochs=30,
    validation_data=test_generator,
    callbacks=[early_stopping, checkpoint, reduce_lr]
)

# Step 3: Evaluate the best model
best_model = load_model('best_model.h5')

# Evaluate the performance of the best model on the test dataset
evaluation = best_model.evaluate(test_generator)
print(f"Test Loss: {evaluation[0]}, Test Accuracy: {evaluation[1]}")

# Make predictions
test_generator.reset()
predictions = best_model.predict(test_generator)
predicted_classes = np.argmax(predictions, axis=1)
true_classes = test_generator.classes
class_labels = list(test_generator.class_indices.keys())

# Classification report
report = classification_report(true_classes, predicted_classes, target_names=class_labels)
print("Classification Report:\n", report)

# AUC calculation
# Calculate AUC for each class
y_true = tf.keras.utils.to_categorical(true_classes, num_classes=4)
auc_scores = []
for i in range(4):
    auc = roc_auc_score(y_true[:, i], predictions[:, i])
    auc_scores.append(auc)

print("AUC Scores per class:", auc_scores)
```


Training and Validation Loss
Training Loss
Validation Loss
Loss
Epochs
Training and Validation Accuracy
Training Accuracy
Validation Accuracy
Accuracy
Epochs