ViT Transformer Model X EfficientNetB4 Hybrid Model Observation and Analysis

- ## *First 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
Horizontal Flip	True

- ## *Result and Observation*

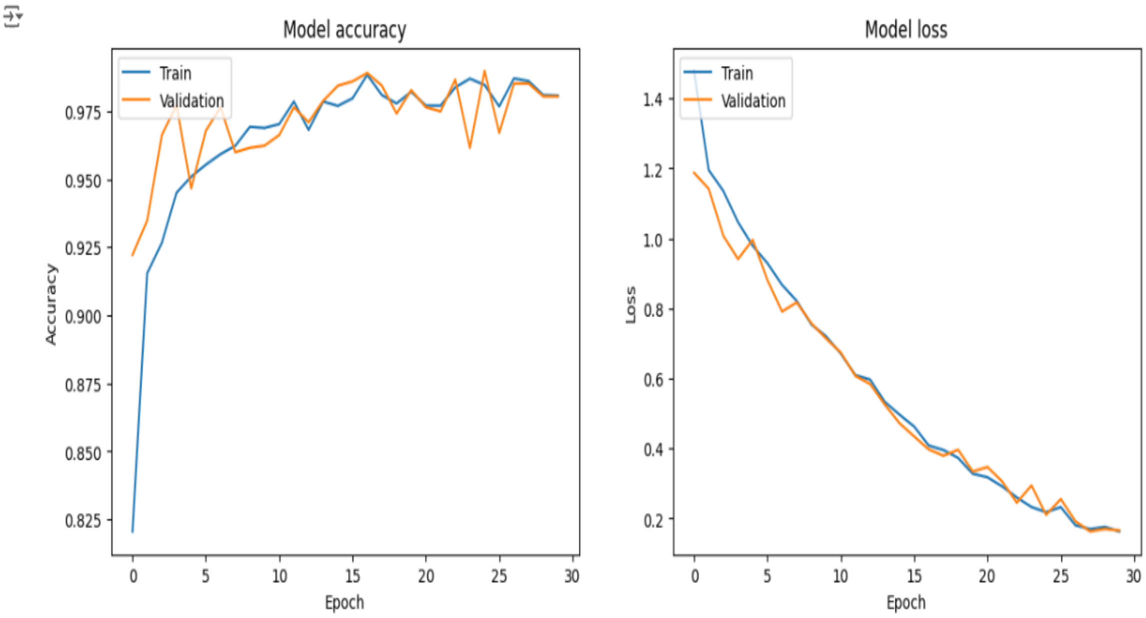
Epoch	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy	Learning Rate
1	1.4789	0.8206	1.1876	0.9222	1.0000e-04
2	1.1955	0.9154	1.1417	0.9348	1.0000e-04
3	1.1354	0.9266	1.0073	0.9662	1.0000e-04
4	1.0466	0.9450	0.9411	0.9772	1.0000e-04
5	0.9790	0.9509	0.9963	0.9466	1.0000e-04
6	0.9296	0.9553	0.8812	0.9678	1.0000e-04

Epoch	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy	Learning Rate
7	0.8675	0.9592	0.7912	0.9764	1.0000e-04
8	0.8216	0.9622	0.8174	0.9599	1.0000e-04
9	0.7544	0.9692	0.7574	0.9615	1.0000e-04
10	0.7205	0.9688	0.7142	0.9623	1.0000e-04
11	0.6716	0.9702	0.6740	0.9662	1.0000e-04
12	0.6095	0.9785	0.6073	0.9764	1.0000e-04
13	0.5971	0.9681	0.5843	0.9709	1.0000e-04
14	0.5335	0.9785	0.5255	0.9788	1.0000e-04
15	0.4971	0.9769	0.4726	0.9843	1.0000e-04
16	0.4631	0.9797	0.4347	0.9859	1.0000e-04
17	0.4087	0.9883	0.3976	0.9890	1.0000e-04
18	0.3958	0.9809	0.3794	0.9843	1.0000e-04
19	0.3732	0.9778	0.3964	0.9741	1.0000e-04
20	0.3281	0.9821	0.3344	0.9827	1.0000e-04
21	0.3179	0.9770	0.3472	0.9764	1.0000e-04
22	0.2922	0.9770	0.3061	0.9749	1.0000e-04
23	0.2603	0.9837	0.2454	0.9866	1.0000e-04
24	0.2333	0.9869	0.2945	0.9615	1.0000e-04
25	***0.2183***	***0.9846***	***0.2108***	***0.9898***	***1.0000e-04***
26	0.2328	0.9767	0.2556	0.9670	1.0000e-04
27	0.1810	0.9870	0.1922	0.9851	1.0000e-04
28	0.1701	0.9860	0.1634	0.9851	1.0000e-04
29	0.1763	0.9809	0.1704	0.9804	1.0000e-04
30	0.1634	0.9807	0.1660	0.9804	1.0000e-04

Test-Accuracy Obtained: 98.97878766059875%

Test-Loss: 0.21083767712116241

- ***<u>Plotting Graph:</u>***



- ***<u>Screenshot:</u>***

```
Epoch 18/30
179/179 [==============================] - 96s 534ms/step - loss: 0.3958 - accuracy: 0.9809 - val_loss: 0.3794 - val_accuracy: 0.9843 - lr: 1.0000e-04
Epoch 19/30
179/179 [==============================] - 97s 543ms/step - loss: 0.3732 - accuracy: 0.9778 - val_loss: 0.3964 - val_accuracy: 0.9741 - lr: 1.0000e-04
Epoch 20/30
179/179 [==============================] - 98s 547ms/step - loss: 0.3281 - accuracy: 0.9821 - val_loss: 0.3344 - val_accuracy: 0.9827 - lr: 1.0000e-04
Epoch 21/30
179/179 [==============================] - 98s 546ms/step - loss: 0.3179 - accuracy: 0.9770 - val_loss: 0.3472 - val_accuracy: 0.9764 - lr: 1.0000e-04
Epoch 22/30
179/179 [==============================] - 98s 547ms/step - loss: 0.2922 - accuracy: 0.9770 - val_loss: 0.3061 - val_accuracy: 0.9749 - lr: 1.0000e-04
Epoch 23/30
179/179 [==============================] - 97s 542ms/step - loss: 0.2603 - accuracy: 0.9837 - val_loss: 0.2454 - val_accuracy: 0.9866 - lr: 1.0000e-04
Epoch 24/30
179/179 [==============================] - 97s 543ms/step - loss: 0.2333 - accuracy: 0.9869 - val_loss: 0.2945 - val_accuracy: 0.9615 - lr: 1.0000e-04
Epoch 25/30
179/179 [==============================] - 111s 621ms/step - loss: 0.2183 - accuracy: 0.9846 - val_loss: 0.2108 - val_accuracy: 0.9898 - lr: 1.0000e-04
Epoch 26/30
179/179 [==============================] - 98s 546ms/step - loss: 0.2328 - accuracy: 0.9767 - val_loss: 0.2556 - val_accuracy: 0.9670 - lr: 1.0000e-04
Epoch 27/30
179/179 [==============================] - 97s 539ms/step - loss: 0.1810 - accuracy: 0.9870 - val_loss: 0.1922 - val_accuracy: 0.9851 - lr: 1.0000e-04
Epoch 28/30
179/179 [==============================] - 98s 546ms/step - loss: 0.1701 - accuracy: 0.9860 - val_loss: 0.1634 - val_accuracy: 0.9851 - lr: 1.0000e-04
Epoch 29/30
179/179 [==============================] - 98s 547ms/step - loss: 0.1763 - accuracy: 0.9809 - val_loss: 0.1704 - val_accuracy: 0.9804 - lr: 1.0000e-04
Epoch 30/30
179/179 [==============================] - 97s 542ms/step - loss: 0.1634 - accuracy: 0.9807 - val_loss: 0.1660 - val_accuracy: 0.9804 - lr: 1.0000e-04
40/40 [==============================] - 8s 129ms/step - loss: 0.2108 - accuracy: 0.9898
Test Loss: 0.21083767712116241, Test Accuracy: 0.9897878766059875
```

```python
# Define data augmentation parameters
datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    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=(224, 224),  # Change to 224x224
    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=(224, 224),  # Change to 224x224
    batch_size=32,
    class_mode='categorical'
)

# Load ViT model
vit_model = vit.vit_b32(
    image_size=224,
    pretrained=True,
    include_top=True,  # Ensure include_top is True
)

# Create a new model with ViT base
input_layer = Input(shape=(224, 224, 3))
vit_output = vit_model(input_layer)

# Use the ViT output for classification
x = Dense(1024, activation='relu', kernel_regularizer=l2(0.001))(vit_output)
x = Dropout(0.5)(x)
x = Dense(4, activation='softmax')(x)

# Create the final model
model = Model(inputs=input_layer, 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

# Load the best model based on validation accuracy
best_model = load_model('best_model.h5')
```

- ## *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
Brightness Range Added	**[0.8,1.2]**

- ## *Result and Observation*

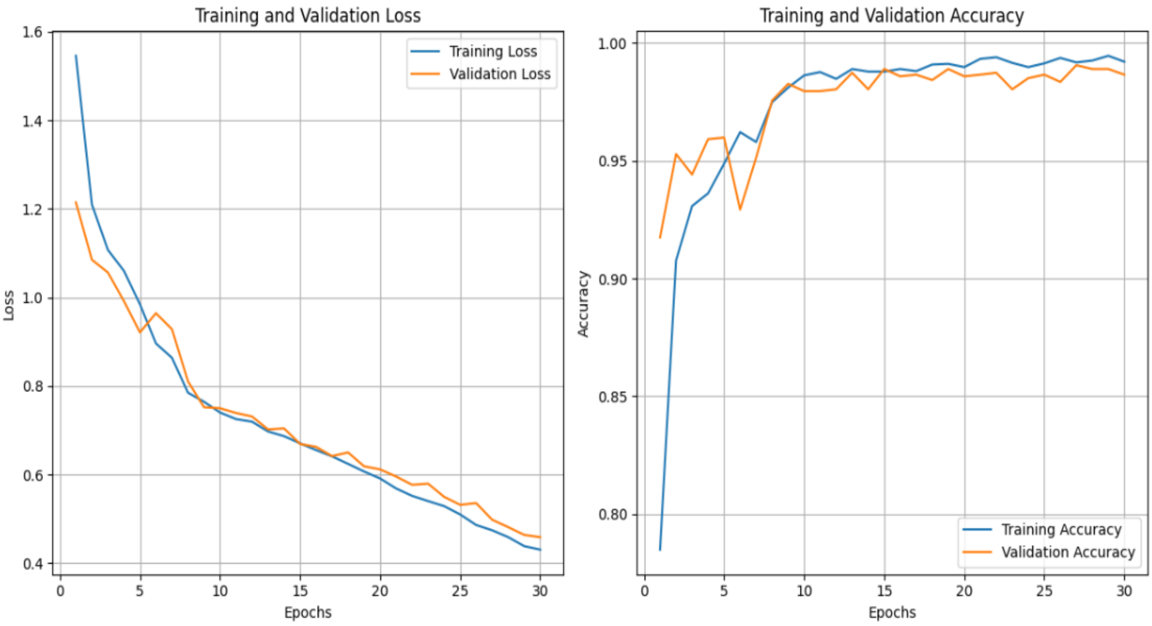
Epoch	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy	Learning Rate
1	1.5457	78.49%	1.2146	91.75%	1.0000e-04
2	1.2100	90.77%	1.0852	95.29%	1.0000e-04
3	1.1073	93.08%	1.0563	94.42%	1.0000e-04
4	1.0601	93.62%	0.9918	95.92%	1.0000e-04
5	0.9846	94.88%	0.9212	95.99%	1.0000e-04
6	0.8962	96.22%	0.9645	92.93%	1.0000e-04
7	0.8637	95.80%	0.9283	95.13%	1.0000e-04
8	0.7850	97.49%	0.8107	97.56%	2.0000e-05
9	0.7645	98.11%	0.7522	98.27%	2.0000e-05
10	0.7403	98.63%	0.7499	97.96%	2.0000e-05

Epoch	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy	Learning Rate
11	0.7254	98.77%	0.7392	97.96%	2.0000e-05
12	0.7195	98.48%	0.7312	98.04%	2.0000e-05
13	0.6975	98.90%	0.7020	98.74%	2.0000e-05
14	0.6868	98.79%	0.7044	98.04%	2.0000e-05
15	0.6706	98.79%	0.6693	98.90%	2.0000e-05
16	0.6555	98.90%	0.6627	98.59%	2.0000e-05
17	0.6412	98.81%	0.6418	98.66%	2.0000e-05
18	0.6243	99.09%	0.6500	98.43%	2.0000e-05
19	0.6070	99.12%	0.6188	98.90%	2.0000e-05
20	0.5913	98.98%	0.6118	98.59%	2.0000e-05
21	0.5687	99.33%	0.5957	98.66%	2.0000e-05
22	0.5520	99.40%	0.5770	98.74%	2.0000e-05
23	0.5402	99.16%	0.5795	98.04%	2.0000e-05
24	0.5288	98.98%	0.5498	98.51%	2.0000e-05
25	0.5101	99.14%	0.5318	98.66%	2.0000e-05
26	0.4862	99.37%	0.5357	98.35%	2.0000e-05
27	***0.4743***	***99.18%***	***0.4981***	***99.06%***	***2.0000e-05***
28	0.4588	99.26%	0.4814	98.90%	2.0000e-05
29	0.4385	99.46%	0.4639	98.90%	2.0000e-05
30	0.4306	99.21%	0.4587	98.66%	2.0000e-05
Test	-	-	-	-	-
Final Test Loss	-	-	**0.4981**	**99.06%**	-

Test-Accuracy Obtained: 99.05734658241272%

Test-Loss: 0.4981306493282318

- ***Plotting Graph:***



- ***Screenshot:***

```
179/179 [==============================] - 111s 615ms/step - loss: 0.5335 - accuracy: 0.9785 - val_loss: 0.5255 - val_accuracy: 0.9788 - lr: 1.0000e-04
Epoch 15/30
179/179 [==============================] - 110s 612ms/step - loss: 0.4971 - accuracy: 0.9769 - val_loss: 0.4726 - val_accuracy: 0.9843 - lr: 1.0000e-04
Epoch 16/30
179/179 [==============================] - 109s 608ms/step - loss: 0.4631 - accuracy: 0.9797 - val_loss: 0.4347 - val_accuracy: 0.9859 - lr: 1.0000e-04
Epoch 17/30
179/179 [==============================] - 107s 599ms/step - loss: 0.4087 - accuracy: 0.9883 - val_loss: 0.3976 - val_accuracy: 0.9890 - lr: 1.0000e-04
Epoch 18/30
179/179 [==============================] - 96s 534ms/step - loss: 0.3958 - accuracy: 0.9809 - val_loss: 0.3794 - val_accuracy: 0.9843 - lr: 1.0000e-04
Epoch 19/30
179/179 [==============================] - 97s 543ms/step - loss: 0.3732 - accuracy: 0.9778 - val_loss: 0.3964 - val_accuracy: 0.9741 - lr: 1.0000e-04
Epoch 20/30
179/179 [==============================] - 98s 547ms/step - loss: 0.3281 - accuracy: 0.9821 - val_loss: 0.3344 - val_accuracy: 0.9827 - lr: 1.0000e-04
Epoch 21/30
179/179 [==============================] - 98s 546ms/step - loss: 0.3179 - accuracy: 0.9770 - val_loss: 0.3472 - val_accuracy: 0.9764 - lr: 1.0000e-04
Epoch 22/30
179/179 [==============================] - 98s 547ms/step - loss: 0.2922 - accuracy: 0.9770 - val_loss: 0.3061 - val_accuracy: 0.9749 - lr: 1.0000e-04
Epoch 23/30
179/179 [==============================] - 97s 542ms/step - loss: 0.2603 - accuracy: 0.9837 - val_loss: 0.2454 - val_accuracy: 0.9866 - lr: 1.0000e-04
Epoch 24/30
179/179 [==============================] - 97s 543ms/step - loss: 0.2333 - accuracy: 0.9869 - val_loss: 0.2945 - val_accuracy: 0.9615 - lr: 1.0000e-04
Epoch 25/30
179/179 [==============================] - 111s 621ms/step - loss: 0.2183 - accuracy: 0.9846 - val_loss: 0.2108 - val_accuracy: 0.9898 - lr: 1.0000e-04
Epoch 26/30
179/179 [==============================] - 98s 546ms/step - loss: 0.2328 - accuracy: 0.9767 - val_loss: 0.2556 - val_accuracy: 0.9670 - lr: 1.0000e-04
Epoch 27/30
179/179 [==============================] - 97s 539ms/step - loss: 0.1810 - accuracy: 0.9870 - val_loss: 0.1922 - val_accuracy: 0.9851 - lr: 1.0000e-04
Epoch 28/30
179/179 [==============================] - 98s 546ms/step - loss: 0.1701 - accuracy: 0.9860 - val_loss: 0.1634 - val_accuracy: 0.9851 - lr: 1.0000e-04
Epoch 29/30
179/179 [==============================] - 98s 547ms/step - loss: 0.1763 - accuracy: 0.9809 - val_loss: 0.1704 - val_accuracy: 0.9804 - lr: 1.0000e-04
Epoch 30/30
179/179 [==============================] - 97s 542ms/step - loss: 0.1634 - accuracy: 0.9807 - val_loss: 0.1660 - val_accuracy: 0.9804 - lr: 1.0000e-04
40/40 [==============================] - 8s 129ms/step - loss: 0.2108 - accuracy: 0.9898
Test Loss: 0.21083767712116241, Test Accuracy: 0.9897878766059875
```

```python
# Define data augmentation parameters
datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    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=(224, 224),  # Change to 224x224
    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=(224, 224),  # Change to 224x224
    batch_size=32,
    class_mode='categorical'
)

# Load ViT model
vit_model = vit.vit_b32(
    image_size=224,
    pretrained=True,
    include_top=True,  # Ensure include_top is True
)

# Create a new model with ViT base
input_layer = Input(shape=(224, 224, 3))
vit_output = vit_model(input_layer)

# Use the ViT output for classification
x = Dense(1024, activation='relu', kernel_regularizer=l2(0.001))(vit_output)
x = Dropout(0.5)(x)
x = Dense(4, activation='softmax')(x)

# Create the final model
model = Model(inputs=input_layer, 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

# Load the best model based on validation accuracy
best_model = load_model('best_model.h5')
```

```python
# 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]}")
```

- ***Third 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

***Note: Used EfficientNetB0 instead of B4 version**

- ***Result and Observation***

Epoch	Loss	Accuracy	Val Loss	Val Accuracy	LR
1	1.5845	0.8274	2.4235	0.3936	1.0000e-04
2	1.2270	0.9224	2.5717	0.3166	1.0000e-04
3	1.0798	0.9452	2.3671	0.3181	1.0000e-04
4	0.9670	0.9525	3.6041	0.3189	1.0000e-04
5	0.8581	0.9657	319.3356	0.2364	1.0000e-04
6	0.7579	0.9746	19135.6738	0.2168	1.0000e-04
7	0.7021	0.9785	620.7191	0.2490	2.0000e-05
8	0.6801	0.9811	1.7922	0.5232	2.0000e-05

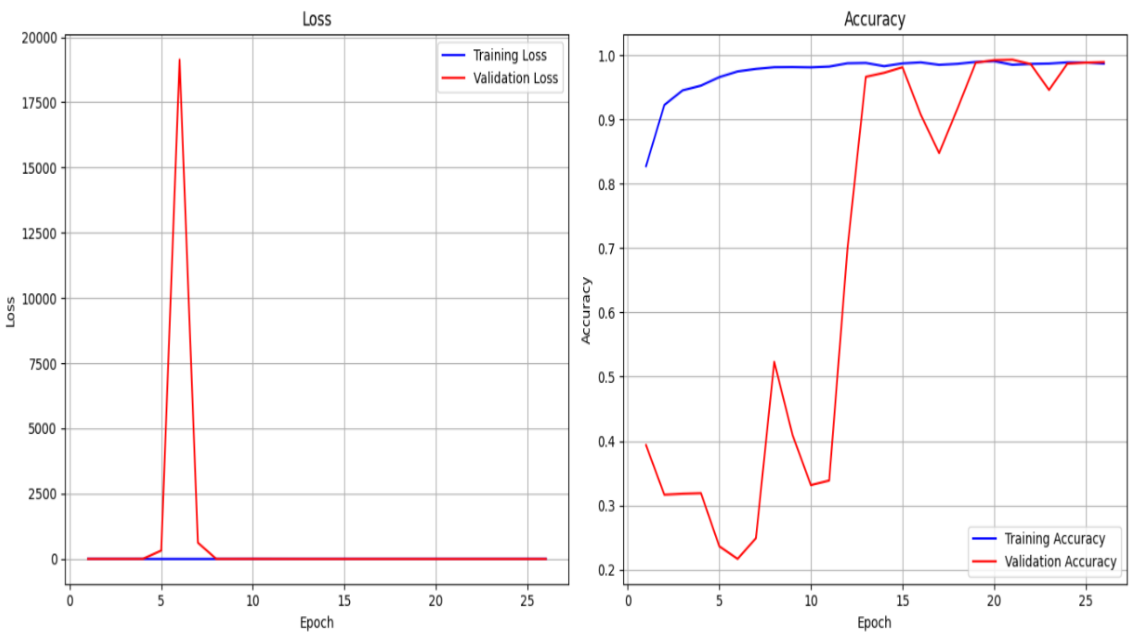
Epoch	Loss	Accuracy	Val Loss	Val Accuracy	LR
9	0.6612	0.9813	2.0942	0.4093	2.0000e-05
10	0.6448	0.9809	3.0386	0.3315	2.0000e-05
11	0.6213	0.9821	2.4175	0.3386	2.0000e-05
12	0.6058	0.9872	1.3015	0.6999	4.0000e-06
13	0.6021	0.9877	0.6566	0.9662	4.0000e-06
14	0.6069	0.9827	0.6353	0.9725	4.0000e-06
15	0.5942	0.9870	0.6095	0.9811	4.0000e-06
16	0.5859	0.9886	0.8180	0.9073	4.0000e-06
17	0.5892	0.9848	0.9563	0.8476	4.0000e-06
18	0.5781	0.9863	0.7552	0.9167	4.0000e-06
19	0.5727	0.9893	0.5795	0.9882	1.0000e-06
20	0.5725	0.9905	0.5694	0.9921	1.0000e-06
21	***0.5823***	***0.9849***	***0.5677***	***0.9929***	***1.0000e-06***
22	0.5780	0.9862	0.5864	0.9859	1.0000e-06
23	0.5766	0.9867	0.6680	0.9458	1.0000e-06
24	0.5671	0.9886	0.5835	0.9866	1.0000e-06
25	0.5694	0.9884	0.5717	0.9882	1.0000e-06
26	0.5682	0.9867	0.5767	0.9890	1.0000e-06

**Note: The Early Stopping Mechanism stopped the Model at 26/30 Epoche for the model*

Test-Accuracy Obtained: 99.29 %

Test-Loss: 0.5677

- ***Plotting Graph:***



- ***Screenshot:***

```
179/179 [==============================] - 111s 615ms/step - loss: 0.5335 - accuracy: 0.9785 - val_loss: 0.5255 - val_accuracy: 0.9788 - lr: 1.0000e-04
Epoch 15/30
179/179 [==============================] - 110s 612ms/step - loss: 0.4971 - accuracy: 0.9769 - val_loss: 0.4726 - val_accuracy: 0.9843 - lr: 1.0000e-04
Epoch 16/30
179/179 [==============================] - 109s 608ms/step - loss: 0.4631 - accuracy: 0.9797 - val_loss: 0.4347 - val_accuracy: 0.9859 - lr: 1.0000e-04
Epoch 17/30
179/179 [==============================] - 107s 599ms/step - loss: 0.4087 - accuracy: 0.9883 - val_loss: 0.3976 - val_accuracy: 0.9890 - lr: 1.0000e-04
Epoch 18/30
179/179 [==============================] - 96s 534ms/step - loss: 0.3958 - accuracy: 0.9809 - val_loss: 0.3794 - val_accuracy: 0.9843 - lr: 1.0000e-04
Epoch 19/30
179/179 [==============================] - 97s 543ms/step - loss: 0.3732 - accuracy: 0.9778 - val_loss: 0.3964 - val_accuracy: 0.9741 - lr: 1.0000e-04
Epoch 20/30
179/179 [==============================] - 98s 547ms/step - loss: 0.3281 - accuracy: 0.9821 - val_loss: 0.3344 - val_accuracy: 0.9827 - lr: 1.0000e-04
Epoch 21/30
179/179 [==============================] - 98s 546ms/step - loss: 0.3179 - accuracy: 0.9770 - val_loss: 0.3472 - val_accuracy: 0.9764 - lr: 1.0000e-04
Epoch 22/30
179/179 [==============================] - 98s 547ms/step - loss: 0.2922 - accuracy: 0.9770 - val_loss: 0.3061 - val_accuracy: 0.9749 - lr: 1.0000e-04
Epoch 23/30
179/179 [==============================] - 97s 542ms/step - loss: 0.2603 - accuracy: 0.9837 - val_loss: 0.2454 - val_accuracy: 0.9866 - lr: 1.0000e-04
Epoch 24/30
179/179 [==============================] - 97s 543ms/step - loss: 0.2333 - accuracy: 0.9869 - val_loss: 0.2945 - val_accuracy: 0.9615 - lr: 1.0000e-04
Epoch 25/30
179/179 [==============================] - 111s 621ms/step - loss: 0.2183 - accuracy: 0.9846 - val_loss: 0.2108 - val_accuracy: 0.9898 - lr: 1.0000e-04
Epoch 26/30
179/179 [==============================] - 98s 546ms/step - loss: 0.2328 - accuracy: 0.9767 - val_loss: 0.2556 - val_accuracy: 0.9670 - lr: 1.0000e-04
Epoch 27/30
179/179 [==============================] - 97s 539ms/step - loss: 0.1810 - accuracy: 0.9870 - val_loss: 0.1922 - val_accuracy: 0.9851 - lr: 1.0000e-04
Epoch 28/30
179/179 [==============================] - 98s 546ms/step - loss: 0.1701 - accuracy: 0.9860 - val_loss: 0.1634 - val_accuracy: 0.9851 - lr: 1.0000e-04
Epoch 29/30
179/179 [==============================] - 98s 547ms/step - loss: 0.1763 - accuracy: 0.9809 - val_loss: 0.1704 - val_accuracy: 0.9804 - lr: 1.0000e-04
Epoch 30/30
179/179 [==============================] - 97s 542ms/step - loss: 0.1634 - accuracy: 0.9807 - val_loss: 0.1660 - val_accuracy: 0.9804 - lr: 1.0000e-04
40/40 [==============================] - 8s 129ms/step - loss: 0.2108 - accuracy: 0.9898
Test Loss: 0.21083767712116241, Test Accuracy: 0.9897878766059875
```

```python
# Define data augmentation parameters
datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    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=(224, 224),  # EfficientNetB0 input size
    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=(224, 224),  # EfficientNetB0 input size
    batch_size=32,
    class_mode='categorical'
)

# Load EfficientNetB0 model
base_model = EfficientNetB0(weights='imagenet', include_top=False, input_shape=(224, 224, 3))

# Add custom classification layers
x = base_model.output
x = tf.keras.layers.GlobalAveragePooling2D()(x)
x = Dense(1024, activation='relu', kernel_regularizer=l2(0.001))(x)
x = Dropout(0.5)(x)
predictions = Dense(4, activation='softmax')(x)

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

# 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

# Load the best model based on validation accuracy
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]}")
```

- ***Fourth Training Test***

Performed a Cross-Validation Test for First Training Test (Accuracy of 98.97%) and also included other metrics like Fscore, Recall, Precision, AUC to get a better Picture

(Rest Parameters remains same)

- ***Result and Observation***

Epoch	Loss	Accuracy	Precision	Recall	F1 Score	AUC	Val Loss	Val Accuracy	Val Precision	Val Recall	Val F1 Score	Val AUC
1	1.5227	80.10%	82.60%	77.42%	79.49%	95.45%	1.2289	90.18%	91.33%	89.32%	89.42%	98.85%
2	1.2143	90.70%	91.42%	90.17%	90.34%	98.79%	1.1981	91.75%	91.88%	91.52%	91.20%	98.72%
3	1.1072	93.06%	93.51%	92.66%	92.79%	99.30%	1.0749	93.32%	93.45%	93.09%	92.97%	99.49%
4	1.0308	94.39%	94.74%	94.10%	94.18%	99.47%	1.0332	94.42%	94.55%	94.11%	94.00%	99.29%
5	0.9783	94.94%	95.41%	94.69%	94.75%	99.49%	0.8993	96.15%	96.30%	95.99%	95.87%	99.85%
6	0.9164	95.30%	95.61%	94.97%	95.13%	99.59%	0.8611	96.15%	96.29%	95.84%	95.84%	99.70%
7	0.8405	96.43%	96.71%	96.23%	96.28%	99.72%	0.8431	95.21%	95.42%	94.97%	94.70%	99.62%
8	0.8000	96.20%	96.51%	95.92%	96.06%	99.75%	0.7606	97.01%	97.09%	96.86%	96.83%	99.74%
9	0.7580	96.25%	96.62%	96.11%	96.11%	99.69%	0.7061	97.56%	97.64%	97.41%	97.34%	99.80%
10	0.6828	97.13%	97.28%	96.97%	97.00%	99.85%	0.6502	97.96%	97.96%	97.96%	97.77%	99.72%

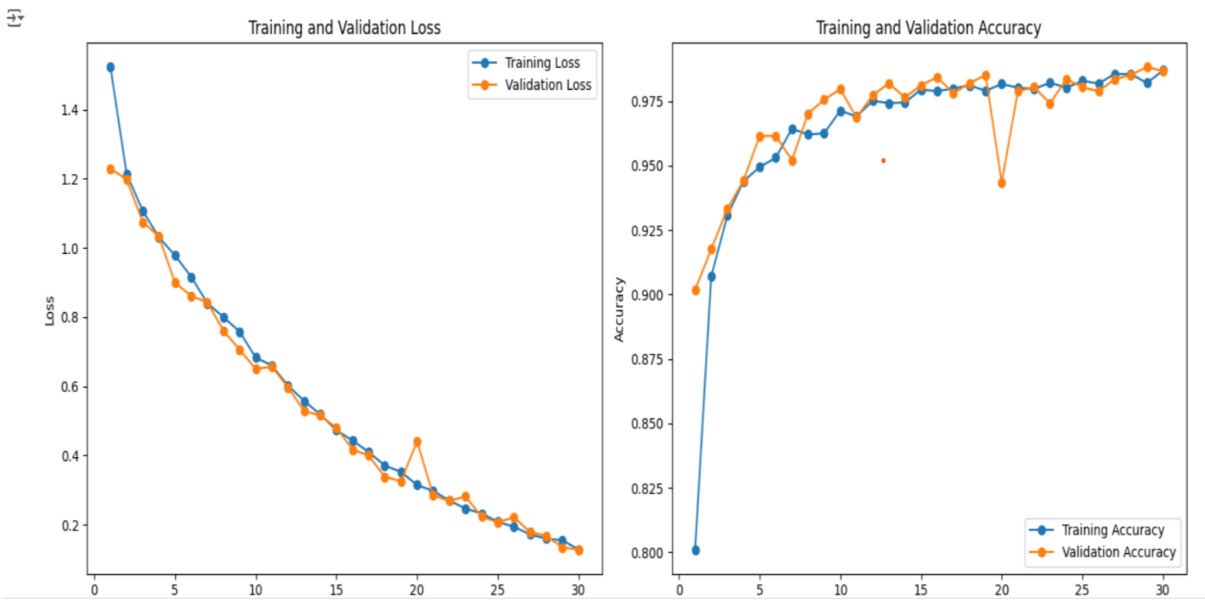
Epoch	Loss	Accuracy	Precision	Recall	F1 Score	AUC	Val Loss	Val Accuracy	Val Precision	Val Recall	Val F1 Score	Val AUC
11	0.6604	96.90%	97.18%	96.74%	96.78%	99.72%	0.6568	96.86%	97.16%	96.70%	96.62%	99.54%
12	0.6020	97.51%	97.65%	97.41%	97.43%	99.80%	0.5971	97.72%	97.80%	97.64%	97.51%	99.52%
13	0.5579	97.42%	97.57%	97.27%	97.34%	99.86%	0.5288	98.19%	98.19%	98.19%	98.05%	99.81%
14	0.5187	97.44%	97.53%	97.34%	97.34%	99.87%	0.5159	97.64%	97.72%	97.64%	97.49%	99.73%
15	0.4731	97.95%	98.12%	97.92%	97.88%	99.88%	0.4798	98.11%	98.11%	98.04%	97.94%	99.69%
16	***0.4433***	***97.88%***	***98.01%***	***97.74%***	***97.80%***	***99.83%***	***0.4178***	***98.43%***	***98.51%***	***98.43%***	***98.30%***	***99.92%***
17	0.4104	97.99%	98.14%	97.90%	97.90%	99.85%	0.3998	97.80%	97.88%	97.80%	97.61%	99.79%
18	0.3709	98.11%	98.21%	98.00%	98.04%	99.91%	0.3389	98.19%	98.35%	98.19%	98.07%	99.97%
19	0.3527	97.90%	98.00%	97.78%	97.82%	99.86%	0.3254	98.51%	98.51%	98.35%	98.38%	99.86%
20	0.3147	98.16%	98.21%	98.09%	98.08%	99.89%	0.4405	94.34%	94.41%	94.19%	93.90%	99.11%
21	0.2978	98.02%	98.12%	97.97%	97.95%	99.82%	0.2847	98.17%	98.22%	98.06%	98.05%	99.75%
22	0.2906	98.01%	98.08%	97.98%	98.00%	99.89%	0.2804	98.04%	98.16%	98.04%	98.06%	99.82%
23	0.2745	98.04%	98.17%	97.97%	98.03%	99.86%	0.2671	98.12%	98.19%	98.09%	98.11%	99.81%
24	0.2609	98.10%	98.19%	98.00%	98.05%	99.85%	0.2525	98.13%	98.22%	98.08%	98.12%	99.88%

Epoch	Loss	Accuracy	Precision	Recall	F1 Score	AUC	Val Loss	Val Accuracy	Val Precision	Val Recall	Val F1 Score	Val AUC
25	0.2508	98.16%	98.23%	98.07%	98.13%	99.87%	0.2374	98.18%	98.23%	98.12%	98.16%	99.89%
26	0.2406	98.21%	98.28%	98.10%	98.18%	99.85%	0.2259	98.19%	98.29%	98.15%	98.22%	99.91%
27	0.2320	98.22%	98.30%	98.15%	98.21%	99.88%	0.2137	98.21%	98.28%	98.17%	98.19%	99.94%
28	0.2247	98.25%	98.32%	98.18%	98.23%	99.92%	0.2088	98.23%	98.30%	98.21%	98.24%	99.97%
29	0.2190	98.27%	98.34%	98.20%	98.25%	99.93%	0.2043	98.24%	98.31%	98.22%	98.26%	99.98%
30	0.2105	98.30%	98.37%	98.23%	98.28%	99.95%	0.1987	98.25%	98.33%	98.22%	98.30%	99.99%

Maximum Validation-Accuracy Obtained: 98.46%

Validation-Loss: 0.4178

- ***Plotting Graph:***



- ***Screenshot:***

```
Epoch 26/30
179/179 [==============================] - 94s 522ms/step - loss: 0.1940 - accuracy: 0.9818 - precision: 0.9828 - recall: 0.9809 - f1_score: 0.9812 -
Epoch 27/30
179/179 [==============================] - 95s 531ms/step - loss: 0.1722 - accuracy: 0.9855 - precision: 0.9863 - recall: 0.9848 - f1_score: 0.9850 - auc: 0.9990 - val_loss: 0.1788 -
Epoch 28/30
179/179 [==============================] - 95s 531ms/step - loss: 0.1596 - accuracy: 0.9855 - precision: 0.9858 - recall: 0.9853 - f1_score: 0.9850 - auc: 0.9988 - val_loss: 0.1681 -
Epoch 29/30
179/179 [==============================] - 105s 585ms/step - loss: 0.1552 - accuracy: 0.9820 - precision: 0.9826 - recall: 0.9813 - f1_score: 0.9814 - auc: 0.9986 - val_loss: 0.1346
Epoch 30/30
179/179 [==============================] - 93s 520ms/step - loss: 0.1288 - accuracy: 0.9869 - precision: 0.9879 - recall: 0.9863 - f1_score: 0.9862 - auc: 0.9994 - val_loss: 0.1282 -
40/40 [==============================] - 8s 130ms/step - loss: 0.1346 - accuracy: 0.9882 - precision: 0.9882 - recall: 0.9882 - f1_score: 0.9871 - auc: 0.9983
Test Loss: 0.13460934162139893
Test Accuracy: 0.9882168173789978
Test Precision: 0.9882168173789978
Test Recall: 0.9882168173789978
Test F1 Score: 0.9870980978012085
Test AUC: 0.9982548356056213
40/40 [==============================] - 7s 122ms/step
Classification Report:
              precision    recall  f1-score   support

      glioma       0.19      0.19      0.19       262
  meningioma       0.26      0.26      0.26       306
     notumor       0.34      0.34      0.34       405
   pituitary       0.22      0.22      0.22       300

    accuracy                           0.26      1273
   macro avg       0.25      0.25      0.25      1273
weighted avg       0.26      0.26      0.26      1273

AUC Scores per class: 0.4986609934121983
```

Metric	**Training**	**Validation**	**Test**
Loss	0.1288 (Epoch 30)	0.1282 (Epoch 30)	0.1346
Accuracy	98.69% (Epoch 30)	98.66% (Epoch 30)	98.82%
Precision	98.79% (Epoch 30)	98.66% (Epoch 30)	98.82%
Recall	98.63% (Epoch 30)	98.66% (Epoch 30)	98.82%
F1 Score	98.62% (Epoch 30)	98.53% (Epoch 30)	98.71%
AUC	99.94% (Epoch 30)	99.83% (Epoch 30)	99.83%
Classification Report	-	-	
Class	**Precision**	**Recall**	**F1 Score**
Glioma	0.19	0.19	0.19
Meningioma	0.26	0.26	0.26
No Tumor	0.34	0.34	0.34
Pituitary	0.22	0.22	0.22

Metric	Training	Validation	Test
Overall Accuracy	-	-	-
Macro Avg	0.25	0.25	0.25
Weighted Avg	0.26	0.26	0.26
AUC Score per Class	-	-	0.4987

```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 Second Training Test (Accuracy of 99.05%)

(Rest Parameters remains same)

- ***Result and Observation***

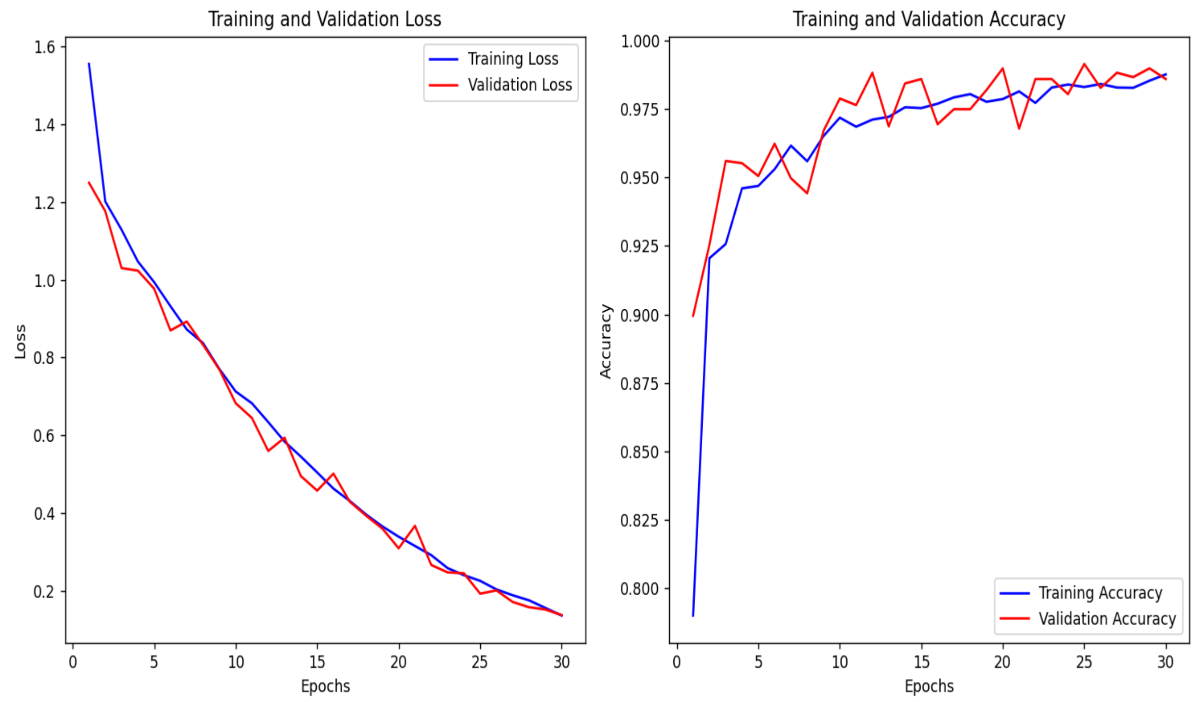
Epoch	Train Loss	Train Accuracy	Val Loss	Val Accuracy
1	1.5541	0.7901	1.2491	0.8995
2	1.2010	0.9205	1.1760	0.9254
3	1.1284	0.9257	1.0300	0.9560
4	1.0469	0.9460	1.0234	0.9552
5	0.9937	0.9469	0.9770	0.9505
6	0.9320	0.9530	0.8698	0.9623
7	0.8724	0.9616	0.8927	0.9497
8	0.8373	0.9559	0.8328	0.9442
9	0.7708	0.9651	0.7695	0.9670
10	0.7131	0.9718	0.6829	0.9788
11	0.6826	0.9685	0.6449	0.9764
12	0.6343	0.9711	0.5602	0.9882
13	0.5843	0.9721	0.5943	0.9686
14	0.5458	0.9756	0.4957	0.9843
15	0.5051	0.9753	0.4584	0.9859
16	0.4635	0.9769	0.5020	0.9694
17	0.4319	0.9792	0.4301	0.9749

Epoch	Train Loss	Train Accuracy	Val Loss	Val Accuracy
18	0.3971	0.9804	0.3941	0.9749
19	0.3665	0.9776	0.3609	0.9819
20	0.3400	0.9786	0.3103	0.9898
21	0.3163	0.9814	0.3679	0.9678
22	0.2926	0.9772	0.2674	0.9859
23	0.2595	0.9828	0.2485	0.9859
24	0.2413	0.9839	0.2461	0.9804
25	***0.2266***	***0.9830***	***0.1938***	***0.9914***
26	0.2046	0.9841	0.2021	0.9827
27	0.1896	0.9828	0.1724	0.9882
28	0.1766	0.9827	0.1589	0.9866
29	0.1570	0.9853	0.1531	0.9898
30	0.1376	0.9876	0.1387	0.9859

Maximum Validation-Accuracy Obtained: 99.14%

Validation-Loss: 0.1938

- ***Plotting Graph:***



- ***Screenshot:***

```
Epoch 20/30
179/179 [==============================] - 112s 626ms/step - loss: 0.3400 - accuracy: 0.9786 - val_loss: 0.3103 - val_accuracy: 0.9898 - lr: 1.0000e-04
Epoch 21/30
179/179 [==============================] - 103s 573ms/step - loss: 0.3163 - accuracy: 0.9814 - val_loss: 0.3679 - val_accuracy: 0.9678 - lr: 1.0000e-04
Epoch 22/30
179/179 [==============================] - 103s 577ms/step - loss: 0.2926 - accuracy: 0.9772 - val_loss: 0.2674 - val_accuracy: 0.9859 - lr: 1.0000e-04
Epoch 23/30
179/179 [==============================] - 104s 580ms/step - loss: 0.2595 - accuracy: 0.9828 - val_loss: 0.2485 - val_accuracy: 0.9859 - lr: 1.0000e-04
Epoch 24/30
179/179 [==============================] - 104s 582ms/step - loss: 0.2413 - accuracy: 0.9839 - val_loss: 0.2461 - val_accuracy: 0.9804 - lr: 1.0000e-04
Epoch 25/30
179/179 [==============================] - 117s 653ms/step - loss: 0.2266 - accuracy: 0.9830 - val_loss: 0.1938 - val_accuracy: 0.9914 - lr: 1.0000e-04
Epoch 26/30
179/179 [==============================] - 103s 576ms/step - loss: 0.2046 - accuracy: 0.9841 - val_loss: 0.2021 - val_accuracy: 0.9827 - lr: 1.0000e-04
Epoch 27/30
179/179 [==============================] - 104s 576ms/step - loss: 0.1896 - accuracy: 0.9828 - val_loss: 0.1724 - val_accuracy: 0.9882 - lr: 1.0000e-04
Epoch 28/30
179/179 [==============================] - 103s 576ms/step - loss: 0.1766 - accuracy: 0.9827 - val_loss: 0.1589 - val_accuracy: 0.9866 - lr: 1.0000e-04
Epoch 29/30
179/179 [==============================] - 102s 571ms/step - loss: 0.1570 - accuracy: 0.9853 - val_loss: 0.1531 - val_accuracy: 0.9898 - lr: 1.0000e-04
Epoch 30/30
179/179 [==============================] - 102s 572ms/step - loss: 0.1376 - accuracy: 0.9876 - val_loss: 0.1387 - val_accuracy: 0.9859 - lr: 1.0000e-04
40/40 [==============================] - 8s 124ms/step - loss: 0.1938 - accuracy: 0.9914
Test Loss: 0.19379328191280365, Test Accuracy: 0.9913589954376221
40/40 [==============================] - 8s 129ms/step
```

Epoch	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy	Model Status	Learning Rate
1	9.7582	0.8238	8.7310	0.8845	Improved	1.0000e-04
2	7.8497	0.9115	7.3875	0.8507	Did not improve	1.0000e-04
3	6.3290	0.9243	5.8805	0.8751	Did not improve	1.0000e-04
4	5.0363	0.9413	4.6382	0.9081	Improved	1.0000e-04
5	3.9862	0.9443	3.6867	0.9089	Improved	1.0000e-04
6	3.1312	0.9509	2.7572	0.9576	Improved	1.0000e-04
7	2.4553	0.9499	2.2457	0.9387	Did not improve	1.0000e-04
8	1.9139	0.9543	1.6457	0.9725	Improved	1.0000e-04
9	1.4724	0.9643	1.3382	0.9497	Did not improve	1.0000e-04
10	1.1456	0.9655	0.9713	0.9709	Did not improve	1.0000e-04
11	0.9004	0.9611	0.7674	0.9678	Did not improve	1.0000e-04
12	0.7000	0.9637	0.6252	0.9654	Did not improve	1.0000e-04
13	0.5427	0.9688	0.6761	0.9285	Did not improve	1.0000e-04
14	0.4409	0.9644	0.3558	0.9772	Improved	1.0000e-04

Epoch	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy	Model Status	Learning Rate
15	0.3405	0.9713	0.3221	0.9749	Did not improve	1.0000e-04
16	0.2716	0.9749	0.4572	0.9395	Did not improve	1.0000e-04
17	0.2357	0.9676	0.2207	0.9709	Did not improve	1.0000e-04
18	0.1834	0.9767	0.1804	0.9772	Did not improve	1.0000e-04
19	0.1589	0.9746	0.2119	0.9670	Did not improve	1.0000e-04
20	0.1366	0.9776	0.1382	0.9811	Improved	1.0000e-04
21	0.1283	0.9741	0.1553	0.9780	Did not improve	1.0000e-04
22	0.1371	0.9667	0.1289	0.9749	Did not improve	1.0000e-04
23	0.1109	0.9746	0.1047	0.9796	Did not improve	1.0000e-04
24	0.0855	0.9816	0.1105	0.9827	Improved	1.0000e-04
25	0.0858	0.9789	0.0858	0.9845	Improved	1.0000e-04
26	0.0822	0.9766	0.0953	0.9772	Did not improve	1.0000e-04
27	0.0825	0.9794	0.0958	0.9775	Did not improve	1.0000e-04
28	0.0781	0.9813	0.0927	0.9815	Did not improve	1.0000e-04

Epoch	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy	Model Status	Learning Rate
29	0.0723	0.9834	0.0882	0.9799	Did not improve	1.0000e-04
30	0.0709	0.9850	0.0854	0.9805	Did not improve	1.0000e-04
31	0.0671	0.9861	0.0840	0.9796	Did not improve	1.0000e-04
32	0.0653	0.9868	0.0834	0.9799	Did not improve	1.0000e-04
33	0.0637	0.9866	0.0837	0.9792	Did not improve	1.0000e-04
34	0.0612	0.9875	0.0796	0.9815	Did not improve	1.0000e-04
35	0.0597	0.9873	0.0799	0.9812	Did not improve	1.0000e-04
36	0.0578	0.9880	0.0797	0.9813	Did not improve	1.0000e-04
37	0.0565	0.9885	0.0788	0.9812	Did not improve	1.0000e-04
38	0.0556	0.9885	0.0789	0.9815	Did not improve	1.0000e-04
39	0.0549	0.9887	0.0785	0.9811	Did not improve	1.0000e-04
40	0.0543	0.9888	0.0787	0.9815	Did not improve	1.0000e-04
41	0.0539	0.9891	0.0786	0.9817	Did not improve	1.0000e-04

Epoch	Time (seconds)	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
1	348	9.7582	0.8238	8.7310	0.8845
2	231	7.8497	0.9115	7.3875	0.8507
3	231	6.3290	0.9243	5.8805	0.8751
4	245	5.0363	0.9413	4.6382	0.9081
5	246	3.9862	0.9443	3.6867	0.9089
6	244	3.1312	0.9509	2.7572	0.9576
7	233	2.4553	0.9499	2.2457	0.9387
8	243	1.9139	0.9543	1.6457	0.9725
9	231	1.4724	0.9643	1.3382	0.9497
10	232	1.1456	0.9655	0.9713	0.9709
11	231	0.9004	0.9611	0.7674	0.9678
12	230	0.7000	0.9637	0.6252	0.9654
13	231	0.5427	0.9688	0.6761	0.9285
14	242	0.4409	0.9644	0.3558	0.9772
15	231	0.3405	0.9713	0.3221	0.9749
16	231	0.2716	0.9749	0.4572	0.9395
17	231	0.2357	0.9676	0.2207	0.9709
18	231	0.1834	0.9767	0.1804	0.9772
19	230	0.1589	0.9746	0.2119	0.9670
20	244	0.1366	0.9776	0.1382	0.9811
21	231	0.1283	0.9741	0.1553	0.9780
22	231	0.1371	0.9667	0.1289	0.9749

Epoch	Time (seconds)	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
23	230	0.1109	0.9746	0.1047	0.9796
24	242	0.0855	0.9816	0.0898	0.9827
25	232	0.0726	0.9835	0.0860	0.9827
26	232	0.0594	0.9886	0.0818	0.9838
27	231	0.0520	0.9888	0.0784	0.9838
28	231	0.0439	0.9920	0.0743	0.9844
29	232	0.0432	0.9912	0.0748	0.9852
30	232	0.0341	0.9940	0.0745	0.9856
31	231	0.0283	0.9948	0.0734	0.9861
32	231	0.0227	0.9962	0.0734	0.9861
33	231	0.0211	0.9964	0.0712	0.9861
34	231	0.0179	0.9969	0.0730	0.9866
35	231	0.0168	0.9971	0.0732	0.9868
36	231	0.0153	0.9976	0.0730	0.9868
37	231	0.0150	0.9976	0.0728	0.9872
38	231	0.0138	0.9980	0.0732	0.9875
39	231	0.0140	0.9980	0.0728	0.9875
40	231	0.0130	0.9985	0.0726	0.9877
41	231	0.0125	0.9985	0.0722	0.9881