# Model Optimization and Tuning Phase Template

| | |
|---|---|
| Date | 10 July 2024 |
| Team ID | SWTID1720043892 |
| Project Title | WCE Curated Colon Disease Using Deep Learning |
| Maximum Marks | 10 Marks |

## Model Optimization and Tuning Phase

The Model Optimization and Tuning Phase involves refining neural network models for peak performance. It includes optimized model code, fine-tuning hyperparameters, comparing performance metrics, and justifying the final model selection for enhanced predictive accuracy and efficiency.

## Hyperparameter Tuning Documentation (8 Marks):

| Model | Tuned Hyperparameters |
|---|---|
| Resnet 50 | Training with 5 Epochs - The model starts to learn basic patterns in the data with just 5 epochs, it's highly likely that the model will underfit, meaning it hasn't had enough time to learn the underlying patterns in the training data. The accuracy and loss metrics might show improvement, but the model's performance will likely be suboptimal.<br><br>Training with 50 Epochs - The model has more time to learn and adjust its weights, leading to better performance. With more epochs, there's a risk of overfitting, where the model learns the noise in the training data, reducing its performance on unseen data. Using techniques like early stopping helps mitigate overfitting by stopping training once the model performance on the validation set stops improving. More epochs usually result in higher training accuracy and potentially higher validation accuracy if overfitting is controlled.<br><br>Early Stopping: raining when the validation accuracy stops improving for a specified patience |

## RESNET 50

```
[23] from tensorflow.keras.applications.resnet50 import ResNet50
     from tensorflow.keras.layers import Dense,Flatten
     from tensorflow.keras.models import Model

     resnet50 = ResNet50(include_top=False,input_shape=(224,224,3))
```

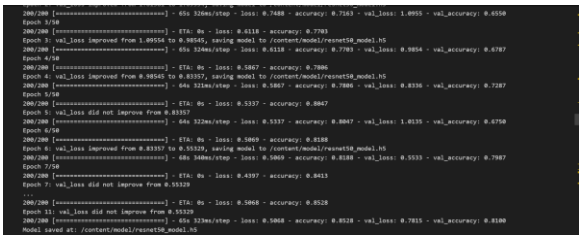
```
[24] for layer in resnet50.layers:
       layer.trainable=False


     x = Flatten()(resnet50.output)
     output = Dense(4,activation='softmax')(x)
     resnet50 = Model(resnet50.input,output)
```

```
     import os

     from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint
     base_path = '/content'
     model_save_path = os.path.join(base_path, "model/resnet50_model.h5")
     ##model.fit(train_generator, validation_data=validation_generator, epochs=2)
     # Define callbacks
     early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)
     checkpoint = ModelCheckpoint(model_save_path, monitor='val_loss', save_best_only=True, verbose=1)

     # Compile the model
     resnet50.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

     # Train the model with callbacks
     history = resnet50.fit(
         train,
         validation_data=test,
         epochs=50,
         callbacks=[early_stopping, checkpoint]
     )

     # Print the model save path
     print(f"Model saved at: {model_save_path}")

     # Save the model
     os.makedirs(os.path.dirname(model_save_path), exist_ok=True)
     resnet50.save(model_save_path)
```

Accuracy-
Epochs-5:

```
Epoch 1/5
200/200 [==============================] - ETA: 0s - loss: 1.0378 - accuracy: 0.7075
Epoch 1: val_loss improved from inf to 0.80137, saving model to /content/model/resnet50_model.h5
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWarning: You are saving your model as an HDF5 file via `model.save()`. This file format is considered
  saving_api.save_model(
200/200 [==============================] - 74s 337ms/step - loss: 1.0378 - accuracy: 0.7075 - val_loss: 0.8014 - val_accuracy: 0.7150
Epoch 2/5
200/200 [==============================] - ETA: 0s - loss: 0.8893 - accuracy: 0.7172
Epoch 2: val_loss did not improve from 0.80137
200/200 [==============================] - 66s 328ms/step - loss: 0.8893 - accuracy: 0.7172 - val_loss: 0.8857 - val_accuracy: 0.7125
Epoch 3/5
200/200 [==============================] - ETA: 0s - loss: 0.4844 - accuracy: 0.8128
Epoch 3: val_loss improved from 0.80137 to 0.73082, saving model to /content/model/resnet50_model.h5
200/200 [==============================] - 66s 330ms/step - loss: 0.4844 - accuracy: 0.8128 - val_loss: 0.7308 - val_accuracy: 0.7400
Epoch 4/5
200/200 [==============================] - ETA: 0s - loss: 0.7683 - accuracy: 0.7681
Epoch 4: val_loss did not improve from 0.73082
200/200 [==============================] - 65s 325ms/step - loss: 0.7683 - accuracy: 0.7681 - val_loss: 1.9453 - val_accuracy: 0.5437
Epoch 5/5
200/200 [==============================] - ETA: 0s - loss: 0.6698 - accuracy: 0.7969
Epoch 5: val_loss did not improve from 0.73082
200/200 [==============================] - 65s 326ms/step - loss: 0.6698 - accuracy: 0.7969 - val_loss: 1.0004 - val_accuracy: 0.7212
Model saved at: /content/model/resnet50_model.h5
```

Epochs-50:




Training with 5 Epochs - The model starts to learn basic patterns in the data With just 5 epochs, it's highly likely that the model will underfit, meaning it hasn't had enough time to learn the underlying patterns in the training data. The accuracy and loss metrics might show improvement, but the model's performance will likely be suboptimal.

Training with 50 Epochs - The model has more time to learn and adjust its weights, leading to better performance.With more epochs, there's a risk of overfitting, where the model learns the noise in the training data, reducing its performance on unseen data.Using techniques like early stopping helps mitigate overfitting by stopping training once the model performance on the validation set stops improving. More epochs usually result in higher training accuracy and potentially higher validation accuracy if overfitting is controlled.

ModelCheckpoint('best_inception_model.h5', monitor='val_accuracy', save_best_only=True): Saves the best model during training based on validation accuracy.
fit(train_gen, validation_data=val_gen, epochs=50, callbacks=[early_stopping, checkpoint]): Trains the model with the training data, validates it with validation data, and uses callbacks for early stopping and checkpointing.

Vgg16

```
from tensorflow.keras.applications.vgg16 import VGG16
from tensorflow.keras.layers import Dense,Flatten
from tensorflow.keras.models import Model

vgg = VGG16(include_top=False,input_shape=(224,224,3))
vgg.summary()
```

```
for layer in vgg.layers:
    print(layer)
```

```
[18] for layer in vgg.layers:
        layer.trainable = False
     x = Flatten()(vgg.output)
     output = Dense(4,activation='softmax')(x)
     vgg16 = Model(vgg.input,output)
     vgg16.summary()
```

```python
import os

from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint
base_path = '/content'
model_save_path = os.path.join(base_path, "model/vgg16_model.h5")
##model.fit(train_generator, validation_data=validation_generator, epochs=2)
# Define callbacks
early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)
checkpoint = ModelCheckpoint(model_save_path, monitor='val_loss', save_best_only=True, verbose=1)

# Compile the model
vgg16.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Train the model with callbacks
history = vgg16.fit(
    train,
    validation_data=test,
    epochs=50,
    callbacks=[early_stopping, checkpoint]
)

# Print the model save path
print(f"Model saved at: {model_save_path}")

# Save the model
os.makedirs(os.path.dirname(model_save_path), exist_ok=True)
vgg16.save(model_save_path)
```

Accuracy –

Epochs-5:

```
Epoch 1/5
200/200 [==============================] - ETA: 0s - loss: 0.8065 - accuracy: 0.8844
Epoch 1: val_loss improved from inf to 1.08180, saving model to /content/model/vgg16_model.h5
200/200 [==============================] - 101s 501ms/step - loss: 0.8065 - accuracy: 0.8844 - val_loss: 1.0818 - val_accuracy: 0.7960
Epoch 2/5
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWarning: You are saving your model as an HDF5 file via `model.sav
  saving_api.save_model(
200/200 [==============================] - ETA: 0s - loss: 0.2207 - accuracy: 0.9641
Epoch 2: val_loss did not improve from 1.08180
200/200 [==============================] - 98s 491ms/step - loss: 0.2207 - accuracy: 0.9641 - val_loss: 2.0029 - val_accuracy: 0.7770
Epoch 3/5
200/200 [==============================] - ETA: 0s - loss: 0.3329 - accuracy: 0.9569
Epoch 3: val_loss did not improve from 1.08180
200/200 [==============================] - 99s 494ms/step - loss: 0.3329 - accuracy: 0.9569 - val_loss: 1.7315 - val_accuracy: 0.8205
Epoch 4/5
200/200 [==============================] - ETA: 0s - loss: 0.2280 - accuracy: 0.9669
Epoch 4: val_loss did not improve from 1.08180
200/200 [==============================] - 98s 490ms/step - loss: 0.2280 - accuracy: 0.9669 - val_loss: 1.1852 - val_accuracy: 0.8895
Epoch 5/5
200/200 [==============================] - ETA: 0s - loss: 0.1681 - accuracy: 0.9812
Epoch 5: val_loss did not improve from 1.08180
200/200 [==============================] - 97s 484ms/step - loss: 0.1681 - accuracy: 0.9812 - val_loss: 3.5791 - val_accuracy: 0.7960
Model saved at: /content/model/vgg16_model.h5
```

Epochs-50:

```
200/200 [==============================] - ETA: 0s - loss: 0.0296 - accuracy: 0.9900
Epoch 5: val_loss did not improve from 0.20602
200/200 [==============================] - 65s 325ms/step - loss: 0.0296 - accuracy: 0.9900 - val_loss: 1.2455 - val_accuracy: 0.7700
Epoch 6/50
200/200 [==============================] - ETA: 0s - loss: 0.0275 - accuracy: 0.9897
Epoch 6: val_loss did not improve from 0.20602
200/200 [==============================] - 64s 320ms/step - loss: 0.0275 - accuracy: 0.9897 - val_loss: 0.3648 - val_accuracy: 0.8712
Epoch 7/50
200/200 [==============================] - ETA: 0s - loss: 0.0292 - accuracy: 0.9891
Epoch 7: val_loss did not improve from 0.20602
200/200 [==============================] - 64s 318ms/step - loss: 0.0292 - accuracy: 0.9891 - val_loss: 1.0586 - val_accuracy: 0.7788
Epoch 8/50
200/200 [==============================] - ETA: 0s - loss: 0.0220 - accuracy: 0.9925
Epoch 8: val_loss did not improve from 0.20602
200/200 [==============================] - 69s 345ms/step - loss: 0.0220 - accuracy: 0.9925 - val_loss: 1.0403 - val_accuracy: 0.7912
Model saved at: /content/model/vgg16_model.h5
```

| | |
|---|---|
| Inception V3 | rescale=1./255: Normalizes the image pixel values to the range [0,1]<br>flow_from_directory: Generates batches of data with real-time data augmentation.<br>target_size: Resizes the input images.<br>class_mode='categorical': Specifies that the target labels are one-hot encoded.<br>Flatten: Converts the 3D output of the base model to 1D.<br>Dense(4, activation='softmax'): Adds a fully connected layer with 4 output units and a softmax activation function for categorical classification.<br><br>Training with 5 Epochs - The model starts to learn basic patterns in the data With just 5 epochs, it's highly likely that the model will underfit, meaning it hasn't had enough time to learn the underlying patterns in the training data. The accuracy and loss metrics might show improvement, but the model's performance will likely be suboptimal.<br><br>Training with 50 Epochs - The model has more time to learn and adjust its weights, leading to better performance.With more epochs, there's a risk of overfitting, where the model learns the noise in the training data, reducing its performance on unseen data.Using techniques like early stopping helps mitigate overfitting by stopping training once the model performance on the validation set stops improving. More epochs usually result in higher training accuracy and potentially higher validation accuracy if overfitting is controlled.<br><br>`from tensorflow.keras.applications.inception_v3 import InceptionV3`<br>`from tensorflow.keras.layers import Dense,Flatten`<br>`from tensorflow.keras.models import Model`<br><br>`InceptionV3 = InceptionV3(include_top=False,input_shape=(224 ,224,3))`<br><br>`for layer in InceptionV3.layers:`<br>`  print(layer)`<br><br>`x = Flatten()(InceptionV3.output)`<br>`output = Dense(4,activation='softmax')(x)`<br>`InceptionV3 = Model(InceptionV3.input,output)`<br>`InceptionV3.summary()` |

```
InceptionV3.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
InceptionV3.fit(train,validation_data=test,epochs=5)

import os

from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint
base_path = '/content'
model_save_path = os.path.join(base_path, "model/InceptionV3_model.h5")
##model.fit(train_generator, validation_data=validation_generator, epochs=2)
# Define callbacks
early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)
checkpoint = ModelCheckpoint(model_save_path, monitor='val_loss', save_best_only=True, verbose=1)

# Compile the model
InceptionV3.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Train the model with callbacks
history = InceptionV3.fit(
    train,
    validation_data=test,
    epochs=50,
    callbacks=[early_stopping, checkpoint]
)

# Print the model save path
print(f"Model saved at: {model_save_path}")

# Save the model
os.makedirs(os.path.dirname(model_save_path), exist_ok=True)
InceptionV3.save(model_save_path)
```

Accuracy –

Epochs -5:

```
Epoch 1/5
200/200 [==============================] - 111s 345ms/step - loss: 0.9464 - accuracy: 0.8166 - val_loss: 808537.1250 - val_accuracy: 0.2500
Epoch 2/5
200/200 [==============================] - 69s 345ms/step - loss: 1.4173 - accuracy: 0.7303 - val_loss: 0.9928 - val_accuracy: 0.7538
Epoch 3/5
200/200 [==============================] - 67s 337ms/step - loss: 0.3797 - accuracy: 0.8672 - val_loss: 0.9083 - val_accuracy: 0.7387
Epoch 4/5
200/200 [==============================] - 68s 337ms/step - loss: 0.3335 - accuracy: 0.8975 - val_loss: 5.1727 - val_accuracy: 0.6237
Epoch 5/5
200/200 [==============================] - 68s 341ms/step - loss: 0.4610 - accuracy: 0.8813 - val_loss: 0.5140 - val_accuracy: 0.7412
```

Epochs -50:

```
Epoch 4/50
200/200 [==============================] - ETA: 0s - loss: 0.2561 - accuracy: 0.9181
Epoch 4: val_loss did not improve from 0.59138
200/200 [==============================] - 69s 342ms/step - loss: 0.2561 - accuracy: 0.9181 - val_loss: 0.6276 - val_accuracy: 0.7412
Epoch 5/50
200/200 [==============================] - ETA: 0s - loss: 0.5895 - accuracy: 0.8800
Epoch 5: val_loss did not improve from 0.59138
200/200 [==============================] - 72s 362ms/step - loss: 0.5895 - accuracy: 0.8800 - val_loss: 1.0470 - val_accuracy: 0.7500
Epoch 6/50
200/200 [==============================] - ETA: 0s - loss: 0.5622 - accuracy: 0.8800
Epoch 6: val_loss did not improve from 0.59138
200/200 [==============================] - 68s 341ms/step - loss: 0.5622 - accuracy: 0.8800 - val_loss: 1.4380 - val_accuracy: 0.7400
Model saved at: /content/model/InceptionV3_model.h5
```

**Final Model Selection Justification (2 Marks):**

| Final Model | Reasoning |
|---|---|
| VGG16 | As Vgg16 has train accuracy of 99.25% which is highest when compared to resnet50's train accuracy 85.28% and inception V3's train accuracy is 88.00% |