Model Optimization and Tuning Phase

Date	20 June 2025
Name	Sakshi Santosh Patil
Project Title	Human-nail-image-processing-using-deep-learning
Maximum Marks	10 Marks

Model Optimization and Tuning Phase

The Model Optimization and Tuning Phase involves improving our neural network models to get the best results. This means adjusting the model's settings, comparing how well different settings work, and explaining why we chose our final model.

The neural network models were trained to classify Nail Disease Images into the following Seventeen classes: Darier_s disease, Muehrck-e_s lines, aloperia areata, beau_s lines, bluish nail, clubbing, eczema, half and half nailes (Lindsay_s nails), koilonychia, leukonychia, onycholycis, pale nail, red lunula, splinter hemmorrage, terry_s nail, white nail, yellow nails
. The training dataset consisted of around 650 labeled Nail disease images across the Seventeen target classes. A separate dataset of 200 images was used for validation and final evaluation of the models.

Hyperparameter Tuning Documentation (8 Marks):

Model	Tuned Hyperparameters

Model 1:
InceptionV3
(Baseline)
	Learning Rate: We adjusted the learning rate, which controls how much the model learns from its mistakes. We tried different learning rates to find one that helps the model learn effectively without becoming unstable.
	`model.compile(`
	`  loss='categorical_crossentropy',`
	`  optimizer='adam',`
	`  metrics=['accuracy']`
	`)`

Model 2:
VGG16
(Optimized)
	Epoch: We made finer adjustments to the epoch, building on what we learned from Model 1, to see if we could improve performance further.
	
	`r = model.fit(`
	`  train_set,`
	`  validation_data=test_set,`
	`  epochs=10,`
	`  steps_per_epoch=len(train_set)//3,`
	`  validation_steps=len(test_set)//3`
	`)`

Batch Size: We used the best batch size from Model 1.

```
train_set = train_datagen.flow_from_directory(
    train_path,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical'
)

test_set = test_datagen.flow_from_directory(
    test_path,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical'
)
```

```
r = model.fit(
    train_set,
    validation_data=test_set,
    epochs=100,
    steps_per_epoch=len(train_set)//3,
    validation_steps=len(test_set)//3
)
```

Accuracy:

```
7/7 ━━━━━━━━━━━━━━━━━━━━ 66s 10s/step - accuracy: 0.9084 - loss: 0.4217 - val_accuracy: 0.9375 - val_loss: 0.2952
Epoch 90/100
7/7 ━━━━━━━━━━━━━━━━━━━━ 58s 8s/step - accuracy: 0.9139 - loss: 0.3492 - val_accuracy: 0.9375 - val_loss: 0.3425
Epoch 91/100
7/7 ━━━━━━━━━━━━━━━━━━━━ 62s 9s/step - accuracy: 0.9077 - loss: 0.4100 - val_accuracy: 0.9688 - val_loss: 0.2367
Epoch 92/100
7/7 ━━━━━━━━━━━━━━━━━━━━ 56s 9s/step - accuracy: 0.8895 - loss: 0.4613 - val_accuracy: 0.9688 - val_loss: 0.2196
Epoch 93/100
7/7 ━━━━━━━━━━━━━━━━━━━━ 59s 9s/step - accuracy: 0.9042 - loss: 0.4053 - val_accuracy: 0.8906 - val_loss: 0.4190
Epoch 94/100
7/7 ━━━━━━━━━━━━━━━━━━━━ 60s 9s/step - accuracy: 0.9336 - loss: 0.3512 - val_accuracy: 0.9062 - val_loss: 0.3295
Epoch 95/100
7/7 ━━━━━━━━━━━━━━━━━━━━ 56s 8s/step - accuracy: 0.9253 - loss: 0.3706 - val_accuracy: 0.9531 - val_loss: 0.2696
Epoch 96/100
7/7 ━━━━━━━━━━━━━━━━━━━━ 59s 9s/step - accuracy: 0.8947 - loss: 0.4200 - val_accuracy: 0.9531 - val_loss: 0.2574
Epoch 97/100
7/7 ━━━━━━━━━━━━━━━━━━━━ 63s 9s/step - accuracy: 0.9417 - loss: 0.3394 - val_accuracy: 0.9375 - val_loss: 0.2911
Epoch 98/100
7/7 ━━━━━━━━━━━━━━━━━━━━ 60s 9s/step - accuracy: 0.9246 - loss: 0.3941 - val_accuracy: 1.0000 - val_loss: 0.1817
Epoch 99/100
7/7 ━━━━━━━━━━━━━━━━━━━━ 56s 8s/step - accuracy: 0.9192 - loss: 0.4135 - val_accuracy: 0.9531 - val_loss: 0.2623
Epoch 100/100
7/7 ━━━━━━━━━━━━━━━━━━━━ 58s 9s/step - accuracy: 0.9421 - loss: 0.3444 - val_accuracy: 0.9844 - val_loss: 0.2170
```

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
Model 1: VGG16 (Optimized)	We selected Model 1 as our final model because it demonstrated a significant improvement in validation accuracy compared to Model 2, achieving 94.21% compared to Model 1's best of 74.59% The image provided shows the training output. We felt the higher accuracy was worth the extra training time. Model 1 also seemed to generalize better to new images.