

## Model Development Phase

Date	20 June 2025
Team ID	SWTID1749826875
Project Title	Dog Breed Identification using Transfer Learning
Maximum Marks	10 Marks

### Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include a summary and training and validation performance metrics for multiple models, presented through respective screenshots.

### Initial Model Training Code (5 marks):

Paste the screenshot of the model training code

```

Model Building

[ ] import tensorflow as tf
    from tensorflow import keras
    from tensorflow.keras.preprocessing.image import ImageDataGenerator
    from tensorflow.keras.layers import Dense
    from tensorflow.keras.activations import softmax
    from keras import activations

[ ] from tensorflow.keras.applications.vgg19 import VGG19
    from tensorflow.keras.layers import Dense, Flatten
    from tensorflow.keras.models import Model
    from tensorflow.keras.optimizers import Adam

[ ] Image_size=[224,224]

[ ] base_model=VGG19(input_shape=Image_size + [3], weights='imagenet', include_top = False)

[ ] for i in base_model.layers:
    i.trainable = False

[ ] y=Flatten()(base_model.output)

[ ] output=Dense(20,activation='softmax')(y)

[ ] vgg19=Model(inputs=base_model.input,outputs=output)

```

## Model Validation and Evaluation Report (5 marks):

Model	Summary	Training and Validation Performance Metrics																																																																																								
VGG19	<div><div>Model: "functional_1"</div><table><thead><tr><th>Layer (type)</th><th>Output Shape</th><th>Param #</th></tr></thead><tbody><tr><td>input_layer_1 (InputLayer)</td><td>(None, 100, 100, 3)</td><td>0</td></tr><tr><td>block1_conv1 (Conv2D)</td><td>(None, 100, 100, 64)</td><td>1,952</td></tr><tr><td>block1_conv2 (Conv2D)</td><td>(None, 100, 100, 64)</td><td>36,800</td></tr><tr><td>block1_pool (MaxPooling2D)</td><td>(None, 50, 50, 64)</td><td>0</td></tr><tr><td>block2_conv1 (Conv2D)</td><td>(None, 50, 50, 128)</td><td>71,680</td></tr><tr><td>block2_conv2 (Conv2D)</td><td>(None, 50, 50, 128)</td><td>147,328</td></tr><tr><td>block2_pool (MaxPooling2D)</td><td>(None, 25, 25, 128)</td><td>0</td></tr><tr><td>block3_conv1 (Conv2D)</td><td>(None, 25, 25, 256)</td><td>298,496</td></tr><tr><td>block3_conv2 (Conv2D)</td><td>(None, 25, 25, 256)</td><td>590,976</td></tr><tr><td>block3_conv3 (Conv2D)</td><td>(None, 25, 25, 256)</td><td>590,976</td></tr><tr><td>block3_conv4 (Conv2D)</td><td>(None, 25, 25, 256)</td><td>590,976</td></tr><tr><td>block3_pool (MaxPooling2D)</td><td>(None, 12, 12, 256)</td><td>0</td></tr><tr><td>block4_conv1 (Conv2D)</td><td>(None, 12, 12, 512)</td><td>1,236,096</td></tr><tr><td>block4_conv2 (Conv2D)</td><td>(None, 12, 12, 512)</td><td>2,472,192</td></tr><tr><td>block4_conv3 (Conv2D)</td><td>(None, 12, 12, 512)</td><td>2,472,192</td></tr><tr><td>block4_conv4 (Conv2D)</td><td>(None, 12, 12, 512)</td><td>2,472,192</td></tr><tr><td>block4_pool (MaxPooling2D)</td><td>(None, 6, 6, 512)</td><td>0</td></tr><tr><td>block5_conv1 (Conv2D)</td><td>(None, 6, 6, 512)</td><td>2,472,192</td></tr><tr><td>block5_conv2 (Conv2D)</td><td>(None, 6, 6, 512)</td><td>2,472,192</td></tr><tr><td>block5_conv3 (Conv2D)</td><td>(None, 6, 6, 512)</td><td>2,472,192</td></tr><tr><td>block5_conv4 (Conv2D)</td><td>(None, 6, 6, 512)</td><td>2,472,192</td></tr><tr><td>block5_pool (MaxPooling2D)</td><td>(None, 3, 3, 512)</td><td>0</td></tr><tr><td>flatten_1 (Flatten)</td><td>(None, 15360)</td><td>0</td></tr><tr><td>dense_1 (Dense)</td><td>(None, 100)</td><td>1,537,000</td></tr></tbody></table><div>Total params: 15,370,000 (78.30 MB) Trainable params: 15,370,000 (78.30 MB) Non-trainable params: 0 (0.00 MB)</div></div>	Layer (type)	Output Shape	Param #	input_layer_1 (InputLayer)	(None, 100, 100, 3)	0	block1_conv1 (Conv2D)	(None, 100, 100, 64)	1,952	block1_conv2 (Conv2D)	(None, 100, 100, 64)	36,800	block1_pool (MaxPooling2D)	(None, 50, 50, 64)	0	block2_conv1 (Conv2D)	(None, 50, 50, 128)	71,680	block2_conv2 (Conv2D)	(None, 50, 50, 128)	147,328	block2_pool (MaxPooling2D)	(None, 25, 25, 128)	0	block3_conv1 (Conv2D)	(None, 25, 25, 256)	298,496	block3_conv2 (Conv2D)	(None, 25, 25, 256)	590,976	block3_conv3 (Conv2D)	(None, 25, 25, 256)	590,976	block3_conv4 (Conv2D)	(None, 25, 25, 256)	590,976	block3_pool (MaxPooling2D)	(None, 12, 12, 256)	0	block4_conv1 (Conv2D)	(None, 12, 12, 512)	1,236,096	block4_conv2 (Conv2D)	(None, 12, 12, 512)	2,472,192	block4_conv3 (Conv2D)	(None, 12, 12, 512)	2,472,192	block4_conv4 (Conv2D)	(None, 12, 12, 512)	2,472,192	block4_pool (MaxPooling2D)	(None, 6, 6, 512)	0	block5_conv1 (Conv2D)	(None, 6, 6, 512)	2,472,192	block5_conv2 (Conv2D)	(None, 6, 6, 512)	2,472,192	block5_conv3 (Conv2D)	(None, 6, 6, 512)	2,472,192	block5_conv4 (Conv2D)	(None, 6, 6, 512)	2,472,192	block5_pool (MaxPooling2D)	(None, 3, 3, 512)	0	flatten_1 (Flatten)	(None, 15360)	0	dense_1 (Dense)	(None, 100)	1,537,000	<div><pre>[ ] from keras.callbacks import EarlyStopping from keras.optimizers import Adam opt = Adam(learning_rate=0.0001)  # Define Early Stopping callback early_stopping = EarlyStopping(monitor='accuracy', patience=3, restore_best_weights=True)  # Compile the model (you may have already done this) vgg19.compile(optimizer=opt, loss='categorical_crossentropy', metrics=['accuracy'])  # Train the model with early stopping callback history = vgg19.fit(generator, epochs=50, callbacks=[early_stopping])</pre><div>Epoch 1/50: 27s 476ms/step - accuracy: 0.5952 - loss: 2.0641 Epoch 2/50: 24s 456ms/step - accuracy: 0.9821 - loss: 0.0747 Epoch 3/50: 24s 457ms/step - accuracy: 0.9716 - loss: 0.0978 Epoch 4/50: 24s 455ms/step - accuracy: 0.9732 - loss: 0.0928 Epoch 5/50: 41s 468ms/step - accuracy: 0.9820 - loss: 0.0710</div><div>x Saving the model</div></div>													
Layer (type)	Output Shape	Param #																																																																																								
input_layer_1 (InputLayer)	(None, 100, 100, 3)	0																																																																																								
block1_conv1 (Conv2D)	(None, 100, 100, 64)	1,952																																																																																								
block1_conv2 (Conv2D)	(None, 100, 100, 64)	36,800																																																																																								
block1_pool (MaxPooling2D)	(None, 50, 50, 64)	0																																																																																								
block2_conv1 (Conv2D)	(None, 50, 50, 128)	71,680																																																																																								
block2_conv2 (Conv2D)	(None, 50, 50, 128)	147,328																																																																																								
block2_pool (MaxPooling2D)	(None, 25, 25, 128)	0																																																																																								
block3_conv1 (Conv2D)	(None, 25, 25, 256)	298,496																																																																																								
block3_conv2 (Conv2D)	(None, 25, 25, 256)	590,976																																																																																								
block3_conv3 (Conv2D)	(None, 25, 25, 256)	590,976																																																																																								
block3_conv4 (Conv2D)	(None, 25, 25, 256)	590,976																																																																																								
block3_pool (MaxPooling2D)	(None, 12, 12, 256)	0																																																																																								
block4_conv1 (Conv2D)	(None, 12, 12, 512)	1,236,096																																																																																								
block4_conv2 (Conv2D)	(None, 12, 12, 512)	2,472,192																																																																																								
block4_conv3 (Conv2D)	(None, 12, 12, 512)	2,472,192																																																																																								
block4_conv4 (Conv2D)	(None, 12, 12, 512)	2,472,192																																																																																								
block4_pool (MaxPooling2D)	(None, 6, 6, 512)	0																																																																																								
block5_conv1 (Conv2D)	(None, 6, 6, 512)	2,472,192																																																																																								
block5_conv2 (Conv2D)	(None, 6, 6, 512)	2,472,192																																																																																								
block5_conv3 (Conv2D)	(None, 6, 6, 512)	2,472,192																																																																																								
block5_conv4 (Conv2D)	(None, 6, 6, 512)	2,472,192																																																																																								
block5_pool (MaxPooling2D)	(None, 3, 3, 512)	0																																																																																								
flatten_1 (Flatten)	(None, 15360)	0																																																																																								
dense_1 (Dense)	(None, 100)	1,537,000																																																																																								
MobileNet V2	<div><div>Model: "functional"</div><table><thead><tr><th>Layer (type)</th><th>Output Shape</th><th>Param #</th><th>Connected to</th></tr></thead><tbody><tr><td>input_layer (InputLayer)</td><td>(None, 100, 100, 3)</td><td>0</td><td>-</td></tr><tr><td>conv1 (Conv2D)</td><td>(None, 100, 100, 32)</td><td>960</td><td>input_layer [1]</td></tr><tr><td>bn_conv1 (BatchNormalization)</td><td>(None, 100, 100, 32)</td><td>0</td><td>conv1 [1]</td></tr><tr><td>conv1_relu (ReLU)</td><td>(None, 100, 100, 32)</td><td>0</td><td>bn_conv1 [1]</td></tr><tr><td>expanded_conv_depthwise (DepthwiseConv2D)</td><td>(None, 100, 100, 32)</td><td>320</td><td>conv1_relu [1]</td></tr><tr><td>expanded_conv_bottleneck (Bottleneck)</td><td>(None, 100, 100, 32)</td><td>0</td><td>expanded_conv_depthwise [1]</td></tr><tr><td>expanded_conv_project (Conv2D)</td><td>(None, 100, 100, 32)</td><td>320</td><td>expanded_conv_bottleneck [1]</td></tr><tr><td>expanded_conv_project_bn (BatchNormalization)</td><td>(None, 100, 100, 32)</td><td>0</td><td>expanded_conv_project [1]</td></tr><tr><td>block1_expand (Conv2D)</td><td>(None, 100, 100, 32)</td><td>1,280</td><td>expanded_conv_project_bn [1]</td></tr><tr><td>block1_expand_bn (BatchNormalization)</td><td>(None, 100, 100, 32)</td><td>0</td><td>block1_expand [1]</td></tr><tr><td>block1_expand_relu (ReLU)</td><td>(None, 100, 100, 32)</td><td>0</td><td>block1_expand_bn [1]</td></tr><tr><td>block1_pool (MaxPooling2D)</td><td>(None, 50, 50, 32)</td><td>0</td><td>block1_expand_relu [1]</td></tr><tr><td>block1_depthwise (DepthwiseConv2D)</td><td>(None, 50, 50, 32)</td><td>320</td><td>block1_pool [1]</td></tr><tr><td>block1_depthwise_bn (BatchNormalization)</td><td>(None, 50, 50, 32)</td><td>0</td><td>block1_depthwise [1]</td></tr><tr><td>block1_depthwise_relu (ReLU)</td><td>(None, 50, 50, 32)</td><td>0</td><td>block1_depthwise_bn [1]</td></tr><tr><td>block1_project (Conv2D)</td><td>(None, 50, 50, 32)</td><td>1,280</td><td>block1_depthwise_relu [1]</td></tr><tr><td>block1_project_bn (BatchNormalization)</td><td>(None, 50, 50, 32)</td><td>0</td><td>block1_project [1]</td></tr><tr><td>block2_expand (Conv2D)</td><td>(None, 50, 50, 32)</td><td>1,280</td><td>block1_project_bn [1]</td></tr><tr><td>block2_expand_bn (BatchNormalization)</td><td>(None, 50, 50, 32)</td><td>0</td><td>block2_expand [1]</td></tr><tr><td>block2_expand_relu (ReLU)</td><td>(None, 50, 50, 32)</td><td>0</td><td>block2_expand_bn [1]</td></tr><tr><td>block2_depthwise (DepthwiseConv2D)</td><td>(None, 50, 50, 32)</td><td>1,280</td><td>block2_expand_relu [1]</td></tr></tbody></table></div>	Layer (type)	Output Shape	Param #	Connected to	input_layer (InputLayer)	(None, 100, 100, 3)	0	-	conv1 (Conv2D)	(None, 100, 100, 32)	960	input_layer [1]	bn_conv1 (BatchNormalization)	(None, 100, 100, 32)	0	conv1 [1]	conv1_relu (ReLU)	(None, 100, 100, 32)	0	bn_conv1 [1]	expanded_conv_depthwise (DepthwiseConv2D)	(None, 100, 100, 32)	320	conv1_relu [1]	expanded_conv_bottleneck (Bottleneck)	(None, 100, 100, 32)	0	expanded_conv_depthwise [1]	expanded_conv_project (Conv2D)	(None, 100, 100, 32)	320	expanded_conv_bottleneck [1]	expanded_conv_project_bn (BatchNormalization)	(None, 100, 100, 32)	0	expanded_conv_project [1]	block1_expand (Conv2D)	(None, 100, 100, 32)	1,280	expanded_conv_project_bn [1]	block1_expand_bn (BatchNormalization)	(None, 100, 100, 32)	0	block1_expand [1]	block1_expand_relu (ReLU)	(None, 100, 100, 32)	0	block1_expand_bn [1]	block1_pool (MaxPooling2D)	(None, 50, 50, 32)	0	block1_expand_relu [1]	block1_depthwise (DepthwiseConv2D)	(None, 50, 50, 32)	320	block1_pool [1]	block1_depthwise_bn (BatchNormalization)	(None, 50, 50, 32)	0	block1_depthwise [1]	block1_depthwise_relu (ReLU)	(None, 50, 50, 32)	0	block1_depthwise_bn [1]	block1_project (Conv2D)	(None, 50, 50, 32)	1,280	block1_depthwise_relu [1]	block1_project_bn (BatchNormalization)	(None, 50, 50, 32)	0	block1_project [1]	block2_expand (Conv2D)	(None, 50, 50, 32)	1,280	block1_project_bn [1]	block2_expand_bn (BatchNormalization)	(None, 50, 50, 32)	0	block2_expand [1]	block2_expand_relu (ReLU)	(None, 50, 50, 32)	0	block2_expand_bn [1]	block2_depthwise (DepthwiseConv2D)	(None, 50, 50, 32)	1,280	block2_expand_relu [1]	<div><pre>from keras.callbacks import EarlyStopping from keras.optimizers import Adam opt = Adam(learning_rate=0.0001)  # Define Early Stopping callback early_stopping = EarlyStopping(monitor='accuracy', patience=3, restore_best_weights=True)  # Compile the model (you may have already done this) mn.compile(optimizer=opt, loss='categorical_crossentropy', metrics=['accuracy'])  # Train the model with early stopping callback history = mn.fit(train_generator, epochs=10, callbacks=[early_stopping])</pre><div>/usr/local/lib/python3.11/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: self._warn_if_super_not_called()  Epoch 1/10: 33s 417ms/step - accuracy: 0.4533 - loss: 2.1744 Epoch 2/10: 21s 394ms/step - accuracy: 0.8767 - loss: 0.4818 Epoch 3/10: 21s 395ms/step - accuracy: 0.8736 - loss: 0.3911 Epoch 4/10: 20s 369ms/step - accuracy: 0.9041 - loss: 0.2937 Epoch 5/10: 21s 397ms/step - accuracy: 0.9080 - loss: 0.2961 Epoch 6/10: 20s 373ms/step - accuracy: 0.9246 - loss: 0.2685 Epoch 7/10: 21s 387ms/step - accuracy: 0.9417 - loss: 0.1875 Epoch 8/10: 21s 387ms/step - accuracy: 0.9424 - loss: 0.1770 Epoch 9/10: 20s 368ms/step - accuracy: 0.9589 - loss: 0.1678 Epoch 10/10: 20s 368ms/step - accuracy: 0.9589 - loss: 0.1678</div></div>
Layer (type)	Output Shape	Param #	Connected to																																																																																							
input_layer (InputLayer)	(None, 100, 100, 3)	0	-																																																																																							
conv1 (Conv2D)	(None, 100, 100, 32)	960	input_layer [1]																																																																																							
bn_conv1 (BatchNormalization)	(None, 100, 100, 32)	0	conv1 [1]																																																																																							
conv1_relu (ReLU)	(None, 100, 100, 32)	0	bn_conv1 [1]																																																																																							
expanded_conv_depthwise (DepthwiseConv2D)	(None, 100, 100, 32)	320	conv1_relu [1]																																																																																							
expanded_conv_bottleneck (Bottleneck)	(None, 100, 100, 32)	0	expanded_conv_depthwise [1]																																																																																							
expanded_conv_project (Conv2D)	(None, 100, 100, 32)	320	expanded_conv_bottleneck [1]																																																																																							
expanded_conv_project_bn (BatchNormalization)	(None, 100, 100, 32)	0	expanded_conv_project [1]																																																																																							
block1_expand (Conv2D)	(None, 100, 100, 32)	1,280	expanded_conv_project_bn [1]																																																																																							
block1_expand_bn (BatchNormalization)	(None, 100, 100, 32)	0	block1_expand [1]																																																																																							
block1_expand_relu (ReLU)	(None, 100, 100, 32)	0	block1_expand_bn [1]																																																																																							
block1_pool (MaxPooling2D)	(None, 50, 50, 32)	0	block1_expand_relu [1]																																																																																							
block1_depthwise (DepthwiseConv2D)	(None, 50, 50, 32)	320	block1_pool [1]																																																																																							
block1_depthwise_bn (BatchNormalization)	(None, 50, 50, 32)	0	block1_depthwise [1]																																																																																							
block1_depthwise_relu (ReLU)	(None, 50, 50, 32)	0	block1_depthwise_bn [1]																																																																																							
block1_project (Conv2D)	(None, 50, 50, 32)	1,280	block1_depthwise_relu [1]																																																																																							
block1_project_bn (BatchNormalization)	(None, 50, 50, 32)	0	block1_project [1]																																																																																							
block2_expand (Conv2D)	(None, 50, 50, 32)	1,280	block1_project_bn [1]																																																																																							
block2_expand_bn (BatchNormalization)	(None, 50, 50, 32)	0	block2_expand [1]																																																																																							
block2_expand_relu (ReLU)	(None, 50, 50, 32)	0	block2_expand_bn [1]																																																																																							
block2_depthwise (DepthwiseConv2D)	(None, 50, 50, 32)	1,280	block2_expand_relu [1]																																																																																							

## EfficientNet B0

```

en.summary()

```

rescaling (rescaling)	(None, 224, 224, 3)	0	input_layer[ ][ ]
normalization (normalization)	(None, 224, 224, 3)	7	rescaling[ ][ ]
rescaling_1 (rescaling)	(None, 224, 224, 3)	0	normalization[ ][ ]
stem_conv_pad (conv2d)	(None, 224, 224, 3)	9	rescaling_1[ ][ ]
stem_conv (conv2d)	(None, 112, 112, 16)	864	stem_conv_pad[ ][ ]
stem_bn (batchnormalization)	(None, 112, 112, 16)	128	stem_conv[ ][ ]
stem_activation (activation)	(None, 112, 112, 16)	0	stem_bn[ ][ ]
block1a_deconv (batchnormalization)	(None, 112, 112, 16)	128	stem_activation[ ][ ]
block1a_bn (batchnormalization)	(None, 112, 112, 16)	128	block1a_deconv[ ][ ]
block1a_activation (activation)	(None, 112, 112, 16)	0	block1a_bn[ ][ ]
block1a_se_squeeze (squeeze_excite)	(None, 16)	0	block1a_activation[ ][ ]
block1a_se_reshape (reshape)	(None, 1, 1, 16)	0	block1a_se_squeeze[ ][ ]
block1a_se_reduce (conv2d)	(None, 1, 1, 16)	160	block1a_se_reshape[ ][ ]
block1a_se_expand (conv2d)	(None, 1, 1, 16)	160	block1a_se_reduce[ ][ ]
block1a_se_excite (excite)	(None, 112, 112, 16)	0	block1a_se_expand[ ][ ]
block1a_project_ch (conv2d)	(None, 112, 112, 112)	112	block1a_se_excite[ ][ ]
block1a_project_bn (batchnormalization)	(None, 112, 112, 112)	64	block1a_project_ch[ ][ ]
block1a_expand_conv (conv2d)	(None, 112, 112, 16)	1,536	block1a_project_bn[ ][ ]
block1a_expand_bn (batchnormalization)	(None, 112, 112, 16)	128	block1a_expand_conv[ ][ ]
block1a_expand_act. (activation)	(None, 112, 112, 16)	0	block1a_expand_bn[ ][ ]

```

[ ] from keras.callbacks import EarlyStopping
from keras.optimizers import Adam
opt = Adam(learning_rate=0.0001)

# Define Early Stopping callback
early_stopping = EarlyStopping(monitor='accuracy', patience=3, restore_best_weights=True)

# Compile the model (you may have already done this)
en.compile(optimizer=opt, loss='categorical_crossentropy', metrics=['accuracy'])

# Train the model with early stopping callback
history = en.fit(train_generator, epochs=10, callbacks=[early_stopping])

```

```

/usr/local/lib/python3.11/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:1
self._warn_if_super_not_called()
Epoch 1/10
53/53 — 53s 565ms/step - accuracy: 0.3755 - loss: 2.1788
Epoch 2/10
53/53 — 21s 392ms/step - accuracy: 0.8292 - loss: 0.5447
Epoch 3/10
53/53 — 21s 386ms/step - accuracy: 0.8884 - loss: 0.3277
Epoch 4/10
53/53 — 21s 403ms/step - accuracy: 0.8893 - loss: 0.3062
Epoch 5/10
53/53 — 21s 392ms/step - accuracy: 0.9149 - loss: 0.2542
Epoch 6/10
53/53 — 41s 401ms/step - accuracy: 0.9332 - loss: 0.2249
Epoch 7/10
53/53 — 21s 391ms/step - accuracy: 0.9466 - loss: 0.1468
Epoch 8/10
53/53 — 20s 378ms/step - accuracy: 0.9460 - loss: 0.1697
Epoch 9/10
53/53 — 21s 398ms/step - accuracy: 0.9529 - loss: 0.1673
Epoch 10/10
53/53 — 20s 379ms/step - accuracy: 0.9668 - loss: 0.1092

```