Model Development Phase

Date	17th June 2025
Team ID	SWTID1749820017
Project Name	Dog Breed Identification using Transfer Learning
Maximum Marks	10 Marks

Initial Model Training Code

The model training code is shown below. It includes model construction, compilation and training using the NASNetLarge architecture and Keras' ImageDataGenerator. Here we are training two models

a) NASNetLarge model that has a Flatten() layer at the second last layer (model1)
b) NASNetLarge model that has a GlobalAveragePooling2D() layer at the second last layer instead of a Flatten() layer at the end (model2)

NASNetLarge model with Flatten() layer (model1)

```
base_model = NASNetLarge(include_top=False,weights='imagenet',input_shape=(331, 331, 3))
base_model.trainable = True
for layer in base_model.layers[:-50]:
    layer.trainable = False
x = base_model.output
x = Flatten()(x)
output = Dense(NUM_CLASSES, activation='softmax')(x)
model1 = Model(inputs=base_model.input, outputs=output)
model1.compile(optimizer=Adam(learning_rate=1e-4),loss='categorical_crossentropy',metrics=['accuracy'])
model1.summary()
```

normal_concat_18 (Concatenate)	(None, 11, 11, 4032)	0	adjust_bn_18[0][… normal_add_1_18[… normal_add_2_18[… normal_add_3_18[… normal_add_4_18[… normal_add_5_18[…
activation_519 (Activation)	(None, 11, 11, 4032)	0	normal_concat_18…
flatten_1 (Flatten)	(None, 487872)	0	activation_519[0…
dense_1 (Dense)	(None, 120)	58,544,760	flatten_1[0][0]

Total params: 143,461,578 (547.26 MB)
Trainable params: 68,599,224 (261.69 MB)
Non-trainable params: 74,862,354 (285.58 MB)

NASNetLarge model with GlobalAveragePooling2D() layer (model1)

```
base_model = NASNetLarge(include_top=False,weights='imagenet',input_shape=(331, 331, 3))
base_model.trainable = True
for layer in base_model.layers[:-50]:
    layer.trainable = False
x = base_model.output
x = GlobalAveragePooling2D()(x)
output = Dense(NUM_CLASSES, activation='softmax')(x)
model2 = Model(inputs=base_model.input, outputs=output)
model2.compile(optimizer=Adam(learning_rate=1e-4),loss='categorical_crossentropy',metrics=['accuracy'])
model2.summary()
```

normal_concat_18 (Concatenate)	(None, 11, 11, 4032)	0	adjust_bn_18[0][… normal_add_1_18[… normal_add_2_18[… normal_add_3_18[… normal_add_4_18[… normal_add_5_18[…
activation_259 (Activation)	(None, 11, 11, 4032)	0	normal_concat_18…
global_average_poo… (GlobalAveragePool…	(None, 4032)	0	activation_259[0…
dense (Dense)	(None, 120)	483,960	global_average_p…

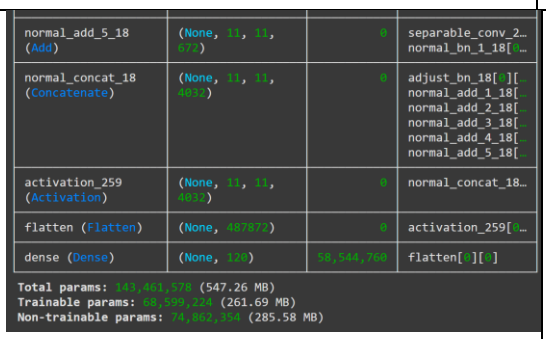
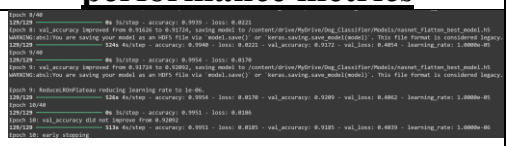
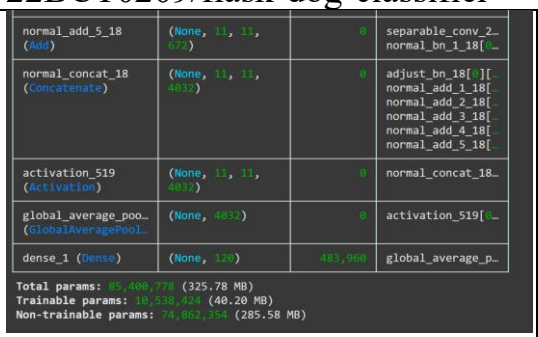
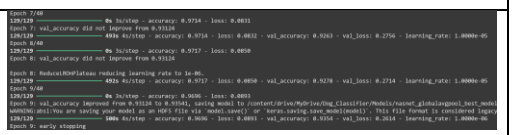
Total params: 85,400,778 (325.78 MB)
Trainable params: 10,538,424 (40.20 MB)
Non-trainable params: 74,862,354 (285.58 MB)

```
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import ModelCheckpoint, ReduceLROnPlateau, EarlyStopping
models = [model1, model2]
model_labels = ['NASNet with Flatten', 'NASNet with GlobalAvgPool']
model_filenames = ['nasnet_flatten_best_model.h5', 'nasnet_globalavgpool_best_model.h5']
histories = {}
for i, model in enumerate(models):
    print(f"\nTraining Model: {model_labels[i]}")
    model.compile(
        optimizer=Adam(learning_rate=1e-4),
        loss='categorical_crossentropy',
        metrics=['accuracy']
    )
    callbacks = [
        ModelCheckpoint(
            filepath=f'/content/drive/MyDrive/Dog_Classifier/Models/{model_filenames[i]}',
            monitor='val_accuracy',
            save_best_only=True,
            verbose=1
        ),
        ReduceLROnPlateau(
            monitor='val_loss',
            factor=0.1,
            patience=3,
            verbose=1,
            min_lr=1e-6
        ),
        EarlyStopping(
            monitor='val_loss',
            patience=7,
            restore_best_weights=True,
            verbose=1
        )
    ]
    history = model.fit(
        train_generator,
        validation_data=val_generator,
        epochs=40,
        callbacks=callbacks,
        verbose=1
    )
    histories[model_labels[i]] = history
```

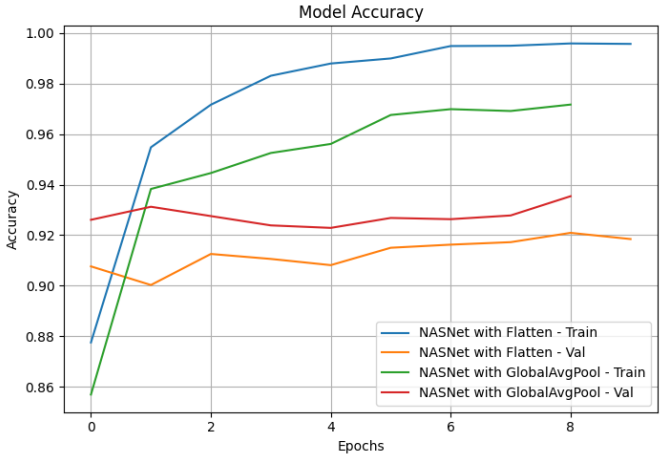
```
Training Model: NASNet with Flatten
/usr/local/lib/python3.11/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: UserWarning: Your `PyDataset` class should call `super().__init__(**kwargs)` in its constructor. `**kwargs` can include `workers`,
  self._warn_if_super_not_called()
Epoch 1/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 4s/step - accuracy: 0.7996 - loss: 1.0014/usr/local/lib/python3.11/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: UserWarning: Your `PyDataset` class should call `super().
  self._warn_if_super_not_called()

Epoch 1: val_accuracy improved from -inf to 0.90766, saving model to /content/drive/MyDrive/Dog_Classifier/Models/nasnet_flatten_best_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.k
129/129 ━━━━━━━━━━━━━━━━━━━━ 725s 5s/step - accuracy: 0.8002 - loss: 0.9980 - val_accuracy: 0.9077 - val_loss: 0.3814 - learning_rate: 1.0000e-04
Epoch 2/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9587 - loss: 0.1376
Epoch 2: val_accuracy did not improve from 0.90766
129/129 ━━━━━━━━━━━━━━━━━━━━ 503s 4s/step - accuracy: 0.9586 - loss: 0.1377 - val_accuracy: 0.9003 - val_loss: 0.4021 - learning_rate: 1.0000e-04
Epoch 3/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9736 - loss: 0.0923
Epoch 3: val_accuracy improved from 0.90766 to 0.91257, saving model to /content/drive/MyDrive/Dog_Classifier/Models/nasnet_flatten_best_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.k
129/129 ━━━━━━━━━━━━━━━━━━━━ 506s 4s/step - accuracy: 0.9736 - loss: 0.0923 - val_accuracy: 0.9126 - val_loss: 0.3778 - learning_rate: 1.0000e-04
Epoch 4/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9840 - loss: 0.0564
Epoch 4: val_accuracy did not improve from 0.91257
129/129 ━━━━━━━━━━━━━━━━━━━━ 516s 4s/step - accuracy: 0.9840 - loss: 0.0564 - val_accuracy: 0.9106 - val_loss: 0.4337 - learning_rate: 1.0000e-04
Epoch 5/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9884 - loss: 0.0421
Epoch 5: val_accuracy did not improve from 0.91257
129/129 ━━━━━━━━━━━━━━━━━━━━ 494s 4s/step - accuracy: 0.9884 - loss: 0.0421 - val_accuracy: 0.9082 - val_loss: 0.4270 - learning_rate: 1.0000e-04
Epoch 6/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9904 - loss: 0.0341
Epoch 6: val_accuracy improved from 0.91257 to 0.91503, saving model to /content/drive/MyDrive/Dog_Classifier/Models/nasnet_flatten_best_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.k

Epoch 6: ReduceLROnPlateau reducing learning rate to 9.999999747378752e-06.
129/129 ━━━━━━━━━━━━━━━━━━━━ 504s 4s/step - accuracy: 0.9904 - loss: 0.0341 - val_accuracy: 0.9150 - val_loss: 0.4138 - learning_rate: 1.0000e-04
Epoch 7/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9948 - loss: 0.0217
Epoch 7: val_accuracy improved from 0.91503 to 0.91626, saving model to /content/drive/MyDrive/Dog_Classifier/Models/nasnet_flatten_best_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.k
129/129 ━━━━━━━━━━━━━━━━━━━━ 533s 4s/step - accuracy: 0.9948 - loss: 0.0217 - val_accuracy: 0.9163 - val_loss: 0.4145 - learning_rate: 1.0000e-05
Epoch 8/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9939 - loss: 0.0221
Epoch 8: val_accuracy improved from 0.91626 to 0.91724, saving model to /content/drive/MyDrive/Dog_Classifier/Models/nasnet_flatten_best_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.k
129/129 ━━━━━━━━━━━━━━━━━━━━ 524s 4s/step - accuracy: 0.9940 - loss: 0.0221 - val_accuracy: 0.9172 - val_loss: 0.4054 - learning_rate: 1.0000e-05
Epoch 9/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9954 - loss: 0.0170
Epoch 9: val_accuracy improved from 0.91724 to 0.92092, saving model to /content/drive/MyDrive/Dog_Classifier/Models/nasnet_flatten_best_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.k

Epoch 9: ReduceLROnPlateau reducing learning rate to 1e-06.
129/129 ━━━━━━━━━━━━━━━━━━━━ 526s 4s/step - accuracy: 0.9954 - loss: 0.0170 - val_accuracy: 0.9209 - val_loss: 0.4062 - learning_rate: 1.0000e-05
Epoch 10/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9951 - loss: 0.0186
Epoch 10: val_accuracy did not improve from 0.92092
129/129 ━━━━━━━━━━━━━━━━━━━━ 513s 4s/step - accuracy: 0.9951 - loss: 0.0185 - val_accuracy: 0.9185 - val_loss: 0.4039 - learning_rate: 1.0000e-06
Epoch 10: early stopping
```

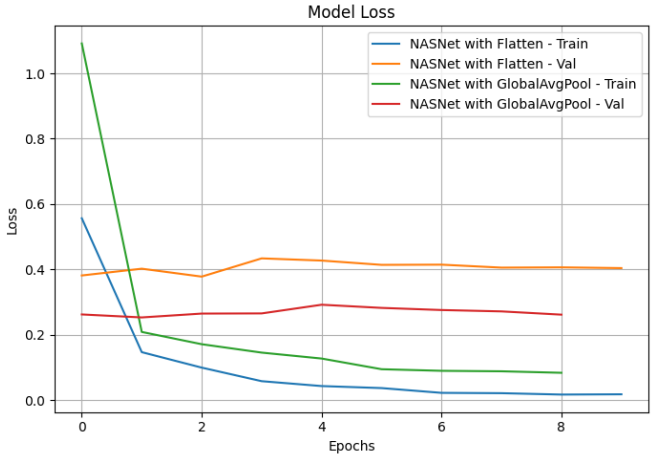
```
Training Model: NASNet with GlobalAvgPool
Epoch 1/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.6877 - loss: 2.3469
Epoch 1: val_accuracy improved from -inf to 0.92608, saving model to /content/drive/MyDrive/Dog_Classifier/Models/nasnet_globalavgpool_best_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.ker
129/129 ━━━━━━━━━━━━━━━━━━━━ 593s 4s/step - accuracy: 0.6890 - loss: 2.3372 - val_accuracy: 0.9261 - val_loss: 0.2621 - learning_rate: 1.0000e-04
Epoch 2/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9412 - loss: 0.2081
Epoch 2: val_accuracy improved from 0.92608 to 0.93124, saving model to /content/drive/MyDrive/Dog_Classifier/Models/nasnet_globalavgpool_best_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.ker
129/129 ━━━━━━━━━━━━━━━━━━━━ 502s 4s/step - accuracy: 0.9412 - loss: 0.2081 - val_accuracy: 0.9312 - val_loss: 0.2529 - learning_rate: 1.0000e-04
Epoch 3/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9470 - loss: 0.1654
Epoch 3: val_accuracy did not improve from 0.93124
129/129 ━━━━━━━━━━━━━━━━━━━━ 497s 4s/step - accuracy: 0.9470 - loss: 0.1654 - val_accuracy: 0.9276 - val_loss: 0.2647 - learning_rate: 1.0000e-04
Epoch 4/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9541 - loss: 0.1414
Epoch 4: val_accuracy did not improve from 0.93124
129/129 ━━━━━━━━━━━━━━━━━━━━ 493s 4s/step - accuracy: 0.9541 - loss: 0.1415 - val_accuracy: 0.9239 - val_loss: 0.2652 - learning_rate: 1.0000e-04
Epoch 5/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9594 - loss: 0.1211
Epoch 5: val_accuracy did not improve from 0.93124

Epoch 5: ReduceLROnPlateau reducing learning rate to 9.999999747378752e-06.
129/129 ━━━━━━━━━━━━━━━━━━━━ 493s 4s/step - accuracy: 0.9594 - loss: 0.1211 - val_accuracy: 0.9229 - val_loss: 0.2918 - learning_rate: 1.0000e-04
Epoch 6/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9679 - loss: 0.0955
Epoch 6: val_accuracy did not improve from 0.93124
129/129 ━━━━━━━━━━━━━━━━━━━━ 493s 4s/step - accuracy: 0.9679 - loss: 0.0954 - val_accuracy: 0.9268 - val_loss: 0.2821 - learning_rate: 1.0000e-05
Epoch 7/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9714 - loss: 0.0831
Epoch 7: val_accuracy did not improve from 0.93124
129/129 ━━━━━━━━━━━━━━━━━━━━ 493s 4s/step - accuracy: 0.9714 - loss: 0.0832 - val_accuracy: 0.9263 - val_loss: 0.2756 - learning_rate: 1.0000e-05
Epoch 8/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9717 - loss: 0.0850
Epoch 8: val_accuracy did not improve from 0.93124

Epoch 8: ReduceLROnPlateau reducing learning rate to 1e-06.
129/129 ━━━━━━━━━━━━━━━━━━━━ 492s 4s/step - accuracy: 0.9717 - loss: 0.0850 - val_accuracy: 0.9278 - val_loss: 0.2714 - learning_rate: 1.0000e-05
Epoch 9/40
129/129 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9696 - loss: 0.0893
Epoch 9: val_accuracy improved from 0.93124 to 0.93541, saving model to /content/drive/MyDrive/Dog_Classifier/Models/nasnet_globalavgpool_best_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.ker
129/129 ━━━━━━━━━━━━━━━━━━━━ 500s 4s/step - accuracy: 0.9696 - loss: 0.0893 - val_accuracy: 0.9354 - val_loss: 0.2614 - learning_rate: 1.0000e-06
Epoch 9: early stopping
```

Model valuation and evaluation report

Model	Summary	Training and validation performance metrics
Model 1: NASNetLarge with Flatten() layer	 Note: As NASNetLarge has several layers it is not possible to take a screenshot of the entire model summary. So a screenshot of the last 4 layers have been displayed here. To get the full model summary please visit the training_notebook folder in the following github repository: https://github.com/HCN-22BCT0209/flask-dog-classifier	 Note: As the screenshot of all the epochs is too large to fit here we have attached the screenshot of the last 3 epochs. To view the full training history checkout the training_notebook folder in the following github repository: https://github.com/HCN-22BCT0209/flask-dog-classifier
Model 2: NASNetLarge with GlobalAvgPool() layer	 Note: As NASNetLarge has several layers it is not possible to take a screenshot of the entire model summary. So a screenshot of the last 4 layers have been displayed here. To get the full model summary please visit the training_notebook folder in the following github repository: https://github.com/HCN-22BCT0209/flask-dog-classifier	 Note: As the screenshot of all the epochs is too large to fit here we have attached the screenshot of the last 3 epochs. To view the full training history checkout the training_notebook folder in the following github repository: https://github.com/HCN-22BCT0209/flask-dog-classifier

Accuracy and Loss graphs:





Confusion Matrix of Model 1 and Model 2: