Binary Classification of Skin Lesions using a Custom ResNet-152 Model

CS22B2020 SriRam

Date: 12 Feb, 2025

Abstract

This report details the implementation of a custom ResNet-152 model for classifying skin lesions as benign or malignant. The model was built from scratch using basic TensorFlow layers (convolution, batch normalization, ReLU, pooling, etc.) rather than prebuilt architectures. We describe the data processing, model architecture, and hyperparameter experiments. The final model achieved a test accuracy of 0.9794, and further metrics such as precision, recall, and F1-score were computed. Insights gained during the experiments are also discussed.

1 Introduction

Early diagnosis of skin cancer through accurate classification of skin lesions is crucial. In this assignment, we implement a deep convolutional neural network (ResNet-152) from scratch for the binary classification task of determining whether a skin lesion is benign or malignant. The approach involves creating a custom dataset loader, building the ResNet-152 architecture using basic building blocks, and evaluating the model on standard metrics.

2 Methodology

2.1 Dataset and Preprocessing

The provided dataset includes a CSV file containing image names and corresponding labels. A custom dataset class was implemented to:

- Load and preprocess images (resize to 224x224, normalize pixel values).
- Filter out any records with missing image files.
- Split the data into training (70%), validation (15%), and test (15%) sets.

2.2 Model Architecture

The ResNet-152 architecture was built using:

- An initial 7×7 convolutional layer followed by max pooling.
- Four residual blocks (conv2_x, conv3_x, conv4_x, conv5_x) implemented via bottleneck blocks.
- Global average pooling and a final dense layer with a sigmoid activation for binary classification.

The model's parameters and architecture were printed to ensure it matched the standard ResNet-152 design.

2.3 Training Strategy and Hyperparameter Tuning

- **Hardware:** Training was conducted on both single and dual GPU setups using TensorFlow's `MirroredStrategy`. Training on a single GPU took approximately 20 minutes, while dual GPU training reduced the time to about 10 minutes.
- **Batch Size:** Experimentation revealed that a batch size of 16 was optimal. Larger batch sizes led to memory errors, whereas smaller batch sizes underutilized the hardware.
- **Learning Rate:** A learning rate of 1×10^{-4} was found to be stable. Higher learning rates caused NaN issues during training, and lower values resulted in slower convergence.

3 Experimental Results

The model achieved the following metrics:

- **Training Data:**
  Accuracy: 0.9646, Loss: 0.1262
  Validation Accuracy: 0.9804, Validation Loss: 0.1422
- **Test Data:**
  Test Accuracy: 0.9794

The overall test metrics were:

- **Accuracy:** 0.9794
- **Precision:** 0.2308
- **Recall:** 0.0690
- **F1 Score:** 0.1062

3.1 Representative Code

Below is a snippet of the key code used for training and evaluation:

```
# Setup multi-GPU training
strategy = tf.distribute.MirroredStrategy()
with strategy.scope():
    model = build_resnet152(input_shape=(224,224,3), num_classes=1)
    model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=1e-4),
                  loss='binary_crossentropy',
                  metrics=['accuracy'])
model.summary()

# Train the model
history = model.fit(train_ds,
                    validation_data=val_ds,
                    epochs=NUM_EPOCHS,
                    callbacks=[checkpoint_cb, earlystop_cb])

# Evaluate on test data and compute overall metrics
predictions = model.predict(test_ds)
# Batch-wise evaluation for accuracy, precision, recall, and F1 score
```

Listing 1: Training and Evaluation Code

4 Discussion

- **Training Efficiency:** Using two GPUs reduced the training time by half compared to a single GPU.
- **Hyperparameter Tuning:** The experiments revealed that a batch size of 16 is optimal for memory and resource utilization. A learning rate of 1×10^{-4} ensured stable convergence without NaN issues.
- **Performance Metrics:** While the overall test accuracy is high (0.9794), the lower precision (0.2308) and recall (0.0690) suggest that further tuning or handling class imbalance may be necessary.

5 Conclusion

This assignment successfully demonstrates the implementation of a custom ResNet-152 model for the binary classification of skin lesions. The model, built from fundamental layers, achieved high accuracy on the test dataset. Experimentation with hyperparameters provided valuable insights into optimizing training efficiency and stability. Future work may include addressing class imbalance and exploring additional regularization techniques to further enhance model performance.