Annglady feat: Completed theoretical answers and debugged TensorFlow script

18fadde · 1 hour ago

55 lines (45 loc) · 2.51 KB

Code Blame Raw

```python
# debug_tensorflow.py (CORRECTED VERSION)
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Flatten, Input # Import Input layer
from tensorflow.keras.optimizers import Adam
import numpy as np

print("Attempting to run corrected code...")

# --- FIXED BUGS ---

# Bug 1 & 5 FIXED: Correct data generation/types and shape for MNIST-like structure
# Assuming we want 28x28 grayscale images, 10 classes, and float32 type
num_samples = 100
image_height = 28
image_width = 28
num_channels = 1 # Grayscale images have 1 channel
num_classes = 10 # Model output will be 10 classes (digits 0-9)

# Generate dummy image data (pixel values between 0 and 1)
X_train_corrected = np.random.rand(num_samples, image_height, image_width, num_channels).a
# Generate dummy integer labels (0 to 9)
y_train_corrected = np.random.randint(0, num_classes, num_samples)

print(f"Generated X_train_corrected shape: {X_train_corrected.shape}")
print(f"Generated y_train_corrected shape: {y_train_corrected.shape}")

# Bug 2 FIXED: Model architecture with explicit Input layer and correct output Dense layer
model = Sequential([
    Input(shape=(image_height, image_width, num_channels)), # Explicitly define input shap
    Flatten(), # Flatten the 28x28x1 image into a 784-element vector
    Dense(128, activation='relu'), # Hidden layer with ReLU activation
    Dense(num_classes, activation='softmax') # Output layer for 10 classes with softmax fo
])

```

```
36  # Bug 3 & 4 FIXED: Correct loss function for integer labels (sparse_categorical_crossentro
37  # Adam optimizer and accuracy metric are appropriate
38  model.compile(optimizer='adam',
39                loss='sparse_categorical_crossentropy', # Corrected: use sparse_categorical_
40                metrics=['accuracy'])
41
42  print("\nModel compiled successfully.")
43  model.summary() # Print model summary to verify architecture
44
45  # Attempt training with corrected data
46  try:
47      print("\nStarting dummy training for 1 epoch...")
48      # Use verbose=0 to suppress per-batch output, just show epoch summary
49      model.fit(X_train_corrected, y_train_corrected, epochs=1, batch_size=32, verbose=1)
50      print("\nCorrected model training attempted and succeeded for 1 epoch!")
51  except Exception as e:
52      print(f"\nError during training (this should not happen with corrected code): {e}")
53      print("Please double-check the pasted code and ensure your venv is active.")
54
55  print("\n--- End of Corrected Code Execution ---")
```