```
! git clone https://github.com/coderhim/Guitar-Tablature-Classification.git
```

```
fatal: destination path 'Guitar-Tablature-Classification' already exists and is
```

```
%run Guitar-Tablature-Classification/engine.py
```

```
Accuracy for string 6: 83.13%
Model saved with validation loss: 0.8282
```

```
def test_model(model, test_loader, device):
    model.eval()
    correct = [0] * 6
    total = [0] * 6

    with torch.no_grad():
        for inputs, labels in test_loader:
            inputs = inputs.to(device)

            # Ensure correct input shape
            if inputs.dim() == 3:
                inputs = inputs.unsqueeze(1)

            outputs = model(inputs)

            # Process labels
            target_indices = []
            for label in labels:
                if label.dim() > 1 and label.shape[1] > 1:
                    indices = torch.argmax(label, dim=1).to(device)
                else:
                    indices = label.to(device).long()
                target_indices.append(indices)

            # Evaluate predictions
            for i, (output, target) in enumerate(zip(outputs, target_indices)):
                _, predicted = torch.max(output, 1)
                correct[i] += (predicted == target).sum().item()
                total[i] += target.size(0)

    # Report accuracy
    for i in range(6):
        accuracy = 100 * correct[i] / total[i] if total[i] > 0 else 0
        print(f"Test Accuracy for string {i + 1}: {accuracy:.2f}%")
```

```
model.eval()
model.to(device)
```

```
GuitarTabNet(
  (conv1): Conv2d(1, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (conv2): Conv2d(32, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (conv3): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (pool): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
 ceil_mode=False)
  (branches): ModuleList(
    (0-5): 6 x Sequential(
      (0): Linear(in_features=12288, out_features=152, bias=True)
      (1): ReLU()
      (2): Dropout(p=0.5, inplace=False)
      (3): Linear(in_features=152, out_features=76, bias=True)
      (4): ReLU()
      (5): Dropout(p=0.2, inplace=False)
      (6): Linear(in_features=76, out_features=19, bias=True)
```

```
            )
          )
        )
```

```
test_model(model, test_loader, device)
```

```
Test Accuracy for string 1: 82.03%
Test Accuracy for string 2: 75.96%
Test Accuracy for string 3: 71.65%
Test Accuracy for string 4: 72.49%
Test Accuracy for string 5: 78.02%
Test Accuracy for string 6: 83.64%
```

```
Start coding or generate with AI.
```