RNN with LSTM for ABC Music Generation - Architecture and Hyperparameters

Model Architecture:

1. **Embedding Layer**
    - Converts integer-encoded characters to dense vector representations.
    - Input Dimension: vocab_size
    - Output Dimension: embedding_dim = 256
2. **LSTM Layer**
    - Long Short-Term Memory layer with 1024 units.
    - Parameters:
        - return_sequences=True: Required for stacked RNNs or sequence output.
        - stateful=True: Maintains state across batches for better sequence learning.
        - recurrent_initializer='glorot_uniform': Weight initializer.
        - recurrent_activation='sigmoid': Activation for gates.
3. **Dense Output Layer**
    - Fully connected layer projecting the LSTM output to vocabulary size.
    - Output Shape: (batch_size, sequence_length, vocab_size)

Loss Function:

- SparseCategoricalCrossentropy with from_logits=True.
- Measures how far the predicted token distribution is from the actual token.

Optimizer:

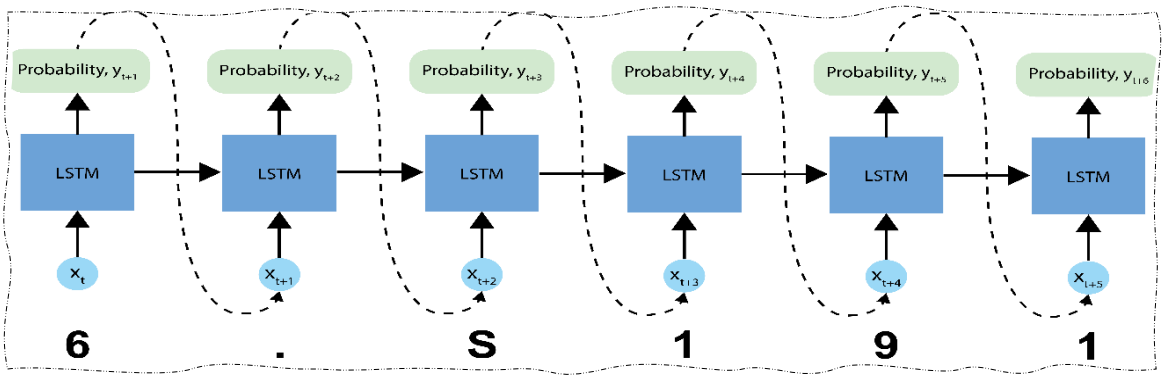
- Adam Optimizer
- Learning Rate: 5e-3

Training Hyperparameters:

- Training Iterations: 3000
- Batch Size: 8
- Sequence Length: 100
- Embedding Dimension: 256
- RNN Units: 1024

Training Procedure:

- Use tf.GradientTape() to compute gradients of the loss with respect to trainable parameters.
- Apply gradients using the optimizer.
- Checkpoints are saved every 100 iterations.



Text Generation Function Modifications:

- Batch size is fixed to 1 for inference.
- Model state is reset using:

```
for layer in model.layers:
    if hasattr(layer, 'reset_states'):
        layer.reset_states()
```

- Predictions are sampled using a categorical distribution:

```
predicted_id = tf.random.categorical(tf.expand_dims(predictions, 0), num_samples=1)[-1, 0].numpy()
```

- Generated characters are appended to the output string using:

```
text_generated.append(idx2char[predicted_id])
```

Interesting Modifications:

- Used tqdm for tracking generation progress.
- Custom model builder with LSTM helper function.
- Integration with Comet ML (via experiment.log_metric) to track training progress.
- Periodic plotting of training loss during training for better monitoring.

Note: An InvalidArgumentError was encountered and resolved by ensuring that predictions passed to tf.random.categorical() are 2D matrices of shape [batch_size, vocab_size] using tf.expand_dims.