EXPERIMENT 4

EXPERIMENT OBJECTIVE

To build or implement a Recurrent Neural Network (RNN) Model that can Predict and Generate Poetry based on a Dataset of 100 Poems.

DATA PREPROCESSING

Loading the Dataset

- The dataset is read from a CSV file and combined into a single text corpus.
- All lines are concatenated to form a continuous sequence of words.

Tokenization and Vocabulary Creation

- The text is split into individual words (tokens).
- A dictionary maps each unique word to a unique index.
- A reverse mapping is also maintained for generating text.

Generating Training Sequences

- A sequence length of 10 words is used.
- Input sequences are created by sliding over the tokenized text.
- The target word for each sequence is the word immediately following the sequence.
- The sequences and targets are converted into PyTorch tensors.

NEURAL NETWORK IMPLEMENTATION

Architecture

- **Embedding Layer:** Converts word indices into dense vectors.
- **Recurrent Layer:** Processes the sequence of word embeddings.
- **Fully Connected Layer:** Maps the RNN output to the vocabulary size for prediction.

Activation Functions

- **Softmax:** Applied to the output layer for probability distribution.

Regularization

- **None**: No regularization techniques mentioned for this model.

TRAINING CONFIGURATION

Training the Model

- **Loss Function:** Cross-entropy loss.
- **Optimizer:** Adam optimizer with a Learning Rate of 0.001.
- **Epochs:** 250.
- **Batch Size:** 32.
- **Training Process:**
    - Batches of sequences are processed.
    - Forward and backward propagation is performed.
    - Weights are updated using the optimizer.

TRAINING AND VALIDATION RESULTS

Key Performance Metrics

- Loss decreased from 7.1773 in the first epoch to 0.0125 in the final epoch, indicating effective learning.

POEM GENERATION

Process

- A function is implemented to generate poetry from a given seed text.
- The model predicts the next word based on previously generated words.
- The generated poem extends for a fixed number of words.

Example Output

Input Seed:

"I wandered lonely as a"

Generated Poem:

"I wandered lonely as a man, Stuff'd with the stuff that is coarse and stuff'd with the stuff that is fine, One of the Nation of many nations, the smallest the same and the largest the same, A Southerner soon as a Northerner, a planter nonchalant and hospitable down by the Oconee I live..."

OBSERVATIONS AND CONCLUSIONS

- The model effectively captures poetic structures and themes.
- Training with more data and fine-tuning hyperparameters can further enhance generation quality.
- More advanced architectures like LSTMs or Transformers may improve coherence and creativity in generated poems.