**Chatbot using Seq2Seq Model with GRU**

**Project Overview**

This project implements a sequence-to-sequence (Seq2Seq) chatbot using TensorFlow. The chatbot is trained on a dataset containing conversational pairs from movies. The model uses an encoder-decoder architecture with GRU (Gated Recurrent Units) to learn how to respond to user queries.

**Dataset:**

* **Source**: /kaggle/input/cleaned-data-for-the-chatbot-collected-from-movies/dialogs\_expanded.csv
* **Content**: Pairs of conversational sentences (question, answer)
* **Preprocessing**:
  + Removal of unnecessary columns: Unnamed: 0, question\_as\_int, answer\_as\_int, question\_len, answer\_len
  + Text cleaning:
    - Lowercasing
    - Removing HTML tags, URLs, special characters, and digits
    - Stripping extra spaces
  + Tokens <start> and <end> are added to each sentence to mark boundaries.

**Tokenization:**

* Tokenization is performed using tf.keras.preprocessing.text.Tokenizer.
* Out-of-vocabulary tokens handled using <OOV>.
* Sequences are padded to ensure consistent input length using pad\_sequences.

**Model Architecture:**

**1. Encoder**

* Input: Tokenized question
* Layers:
  + **Embedding Layer**: Converts word indices to dense vectors.
  + **GRU Layer**: Processes sequences and returns both outputs and hidden state.
* Output: Final encoder state passed to the decoder.

**2. Decoder**

* Input: Tokenized answer (with teacher forcing)
* Layers:
  + **Embedding Layer**
  + **GRU Layer**
  + **Dense Layer**: Produces probability distribution over vocabulary.
* Output: Predicted word token for each timestep.

**Configuration:**

* **Vocabulary Size**:
  + Input: vocab\_inp\_size = len(question\_tokenizer.word\_index) + 1
  + Target: vocab\_tar\_size = len(answer\_tokenizer.word\_index) + 1
* **Hyperparameters**:
  + embedding\_dim = 256
  + units = 1024
  + batch\_size = 32
* **Loss Function**: Sparse Categorical Crossentropy (masked to ignore padding)
* **Optimizer**: Adam
* **Metrics**: Average batch loss for both training and testing

**Data Splitting:**

* train\_test\_split is used to divide the dataset:
  + 90% for training
  + 10% for testing

**Training Loop:**

* **Epochs**: 10
* **Teacher Forcing**: The target token at time *t* is fed as input to time *t+1*.
* **Steps**:
  1. Encode the input sequence
  2. Initialize the decoder with <start> token
  3. Generate predictions step-by-step
  4. Compute and accumulate loss
  5. Apply backpropagation and update weights using gradient tape

**Evaluation:**

* The model is evaluated using a similar step-by-step method as training but without updating weights.
* Loss is calculated and tracked per batch.

**Sample Outputs:**

During the development, a sample from the training dataset is printed to show:

* Shape and content of tokenized question and answer
* Mapping between token indices and actual words

**Summary:**

· This chatbot project provides a strong baseline for building conversational agents using TensorFlow and movie-based dialogue data. With additional enhancements like attention and real-time inference, it can be transformed into a deployable chatbot application.