Language Model

Introduction

This project focuses on training a **word-level language model** using text from *Poirot Investigates* by Agatha Christie. The goal is to develop a model that can predict the next word in a given sequence and generate text that mimics the author's writing style. The dataset was preprocessed, tokenized, and used to train a **Recurrent Neural Network (RNN) with Long Short-Term Memory (LSTM) layers**. The final model was evaluated based on accuracy and text generation quality, demonstrating its ability to generate contextually relevant and stylistically consistent text.

Data Processing

- The dataset was extracted from the Project Gutenberg text file of *Poirot Investigates*.
- **Text Cleaning**: All special characters and punctuation marks were removed, and the text was converted to lowercase to ensure uniformity.
- **Metadata Removal**: Header and footer information from the Project Gutenberg file were stripped to retain only the main book content.
- **Tokenization**: The text was broken down into individual words using TensorFlow's `Tokenizer`, and a **vocabulary size of 30,000 words** was set to limit memory usage while preserving important words.
- **Pretrained Word Embeddings**: **GloVe 100-dimensional embeddings** were incorporated to improve word representation by leveraging knowledge from large-scale corpus training.
- **Sequence Creation**: The text was split into sequences of words, where each sequence was **padded to a maximum length of 100 words**, ensuring consistent input size for training.

Model Architecture & Training

The language model was built using an **RNN with LSTM layers**, which is well-suited for sequential text prediction. The architecture consisted of:

- **Embedding Layer**: Converts tokenized words into dense vector representations, initialized with **GloVe embeddings**.
- **LSTM Layers**:
  - First **LSTM layer with 256 units**, responsible for capturing long-term dependencies.
  - Second **LSTM layer with 128 units**, refining learned patterns from the first LSTM layer.
- **LayerNormalization & Dropout**:
  - **LayerNormalization** stabilizes training and speeds up convergence.
  - **30% dropout** was applied to reduce overfitting and improve generalization.
- **Dense Layers**:
  - A fully connected **128-unit ReLU layer** enhances feature extraction.
  - A **softmax output layer** predicts the next word from the vocabulary.

- **Loss Function**: Categorical cross-entropy was used since the model is predicting a probability distribution over words.
- **Optimizer**: Adam (**learning rate: 0.0003, clipnorm: 1.0**) was chosen due to its efficient adaptive learning.
- **Training Setup**:
  - The model was trained for **250 epochs**.
  - **Early stopping** was applied to halt training if no improvement was observed.
  - **Learning rate reduction** was used to fine-tune the model during training.

Hyperparameters

Choosing the right hyperparameters was essential for ensuring optimal model performance. The following values were selected:

- **Optimizer**: Adam (**learning rate = 0.0003**) was chosen for its adaptive learning rate and robust performance in deep learning models.
- **Batch Size**: **64 –** a balanced choice between learning stability and computational efficiency.
- **Sequence Length**: **100 –** this captures sufficient context for text generation while maintaining efficient memory usage.
- **Epochs**: **250** – model performance stabilized within this range, ensuring adequate training without overfitting.
- **Dropout Rate**: **30% –** to reduce overfitting and improve generalization.
- **Temperature for Text Generation**: **0.8 –** controls randomness in the model's predictions, ensuring variability in generated text while maintaining coherence.

Results

- **Final Training Accuracy**: **79.86%**
- **Final Training Loss**: **0.6725**
- **Generated Text Example:**
  "The great financier was perfectly right in the afternoon Poirot gave a policeman getting of his own flesh and cry the nephew."
- **Model, tokenizer, and best weights** have been saved to **Google Drive** for future testing and fine-tuning.

Alternative Approaches Considered

While an LSTM-based approach was chosen, other methods could have been explored:

- **GRU-Based RNN**: Using **Gated Recurrent Units (GRUs)** instead of LSTMs for faster training while maintaining sequence modeling capabilities.
- **Transformer-Based Models**: Implementing architectures such as **GPT-2, BERT, or T5** for improved long-term dependency handling.

- **Hybrid Models**: Combining **CNN + LSTM** to enhance feature extraction and sequential learning.
- **Pretrained Language Models**: Fine-tuning larger **pretrained transformer models** like **GPT-3.5/4**.
- **Character-Level Models**: Instead of word-level tokenization, predicting text at the **character level** for increased flexibility in handling unknown words.

Performance Enhancement Techniques

To improve text prediction, the following techniques could be applied in order of expected effectiveness:

1. **Use a Transformer-Based Model**: LSTMs struggle with long-range dependencies, while transformers handle them efficiently.
2. **Increase Training Data**: Using multiple books by Agatha Christie to train a more diverse and robust model.
3. **Hyperparameter Optimization**: Conducting a thorough **grid search** or using **Bayesian optimization** for better model tuning.
4. **Augment the Training Data**: Introducing **sentence paraphrasing** and restructured text data to improve generalization.
5. **Larger Embedding Size**: Increasing from **100D GloVe embeddings to 300D** could improve word representation.

Comparison with Large Language Models (LLMs)

- **LLMs such as GPT-3 and GPT-4** are trained on massive datasets and use transformers, making them superior in generating human-like text.
- **Our model is domain-specific**, trained exclusively on *Poirot Investigates*, while LLMs are general-purpose.
- **Prompt Engineering** can be used to guide LLMs in generating more Christie-like text without additional fine-tuning.
- **Bridging the Gap**: Fine-tuning a smaller transformer model like **GPT-2** could help our model generate more realistic sentences while maintaining computational efficiency.
- **Efficiency Considerations**: Our approach is lightweight and can be trained and deployed on local machines, unlike massive LLMs that require cloud-based infrastructure.

Conclusion

This project successfully trained an **LSTM-based language model** capable of generating text in the style of Agatha Christie. While LSTMs work well for sequential text modeling, **transformer-based architectures remain superior** in handling long-range dependencies and generating highly coherent text. The **trained model, tokenizer, and best weights**have been stored for potential future refinements, fine-tuning, or comparisons with more advanced architectures.