Word Completion using LSTM Neural Networks on Shakespeare's Works

***Abstract*—This paper presents a word-level Long Short-Term Memory (LSTM) model for sentence completion trained on Shakespeare's works. We implement a neural network that can predict the next word in a sequence given a partial sentence, capturing the unique vocabulary and linguistic patterns of Shakespearean text. The model achieves 8.97% validation accuracy on the large vocabulary of over 22,500 words. We demonstrate how bidirectional LSTM layers, strategic preprocessing techniques, and vocabulary management significantly improve prediction quality. The system successfully generates contextually appropriate continuations of Shakespearean quotes while avoiding the prediction of unknown tokens. This implementation provides insights into both the challenges and techniques for developing language models for specific literary styles.**

***Index Terms*—LSTM, natural language processing, word prediction, Shakespeare, bidirectional neural networks, language modeling**

I. Introduction

Language models capable of predicting the next word in a sequence have become increasingly important in various natural language processing applications. This paper focuses on building a word-level Long Short-Term Memory (LSTM) model specifically trained on Shakespeare's plays. The objective is to create a model that can predict the next word in a sentence given a partial sequence, capturing the unique vocabulary, grammar, and style of Shakespearean English.

Shakespeare's works present a fascinating challenge for language models due to their distinct vocabulary, archaic expressions, and complex linguistic patterns. Training a neural network to understand and generate Shakespearean text requires addressing several challenges, including handling a specialized vocabulary, managing limited training data, and capturing the unique syntactic structures characteristic of Shakespeare's writing.

Our implementation uses bidirectional LSTM layers, which allow the model to learn from both past and future context, providing a more comprehensive understanding of the text. We explore various techniques to optimize the model's performance, including vocabulary filtering, sequence length optimization, and temperature scaling for generation diversity.

The model is evaluated on its ability to:

1) Accurately predict the next word in a Shakespearean sentence
2) Generate coherent and contextually appropriate continuations
3) Avoid predicting unknown tokens
4) Capture aspects of Shakespeare's unique writing style

II. Methodology

A. Dataset

We utilized a dataset of Shakespeare's complete works from Kaggle, containing all of Shakespeare's plays in two formats:

- `Shakespeare_data.csv`: A CSV file containing 110,793 lines with structured information including the play, act-scene-line, player, and the spoken lines
- `alllines.txt`: A text file containing 111,396 lines of dialogue from Shakespeare's plays

The combined dataset provided approximately 1.66 million words with 49,662 unique words before preprocessing. This represents the complete corpus of Shakespeare's dramatic works, offering a comprehensive sample of his distinctive vocabulary and writing style.

B. Data Preprocessing

Several preprocessing steps were applied to prepare the text data for training:

1) **Text Cleaning**: We removed special characters while preserving important punctuation (commas, periods, question marks, etc.) to maintain sentence structure. Stage directions enclosed in square brackets were filtered out. All text was converted to lowercase for consistency.
2) **Vocabulary Management**: To balance model size and performance, we filtered out rare words that appeared fewer than 2 times in the corpus. This reduced the vocabulary size while retaining important Shakespearean words.
3) **Shakespeare-Specific Vocabulary**: We explicitly included a curated list of specialized Shakespearean words (e.g., "thee," "thou," "wherefore," "discontent," "anon") to ensure the model could handle key terms even if they appeared infrequently.
4) **Tokenization**: We used Keras' Tokenizer to convert words to numerical indices, including an out-of-vocabulary "`<UNK>`" token for unseen words.
5) **Sequence Generation**: We created input sequences of 8 consecutive words, with the goal of predicting the 9th word. This sequence length provides sufficient context for accurate prediction while remaining computationally manageable.

After preprocessing, our final vocabulary contained 22,586 unique tokens, capturing both common English words and Shakespeare-specific terminology.

C. Model Architecture

We designed a deep neural network architecture optimized for capturing the nuances of Shakespearean language:

Layer	Details
Embedding	Vocabulary size: 22,586, Dimensions: 200
Bidirectional LSTM	Units: 256, Return sequences: True
Dropout	Rate: 0.3
Bidirectional LSTM	Units: 192, Return sequences: True
Dropout	Rate: 0.3
Bidirectional LSTM	Units: 128
Dropout	Rate: 0.3
Dense	Units: 512, Activation: ReLU
Dropout	Rate: 0.3
Dense (Output)	Units: Vocabulary Size (22,586), Activation: Softmax

TABLE I
MODEL ARCHITECTURE

Key features of the architecture include:

- **Bidirectional LSTM Layers**: Allowing the model to learn context from both preceding and following words, which is crucial for understanding Shakespeare's complex sentence structures.
- **Multiple LSTM Layers**: Three stacked bidirectional LSTM layers with decreasing unit counts (256→192→128) to capture hierarchical patterns in the text.
- **Dropout Regularization**: Applied after each LSTM layer and the dense layer at a rate of 0.3 to prevent overfitting.
- **Dense Layer with ReLU**: A 512-unit dense layer with ReLU activation provides additional representational capacity before the final prediction layer.
- **Softmax Output**: The final layer outputs a probability distribution over the entire vocabulary.

D. Training Procedure

The model was compiled with the following parameters:

- **Loss Function**: Sparse categorical cross-entropy
- **Optimizer**: Adam with an initial learning rate of 0.001
- **Metrics**: Accuracy

To address memory constraints and improve training efficiency, we implemented a batch-based training approach:

1) Sequences were generated and processed in batches of 100,000
2) Each batch was split into training (90%) and validation (10%) sets
3) The model was trained for up to 25 epochs per batch, with early stopping based on validation accuracy
4) We used a learning rate reduction schedule to improve convergence
5) Four batches (approximately 400,000 sequences) were processed in total

III. RESULTS

A. Training Performance

The model's training performance showed steady improvement across batches:

TABLE II
TRAINING PERFORMANCE METRICS

Metric	Initial Value	Final Value
Training Accuracy	3.2%	10.7%
Validation Accuracy	3.84%	8.97%
Training Loss	7.42	5.51
Validation Loss	6.68	6.53

The learning curves (Figure **??**) show that the model steadily improved over the training epochs, with validation accuracy plateauing at approximately 9%. This level of accuracy is reasonable given the large vocabulary size (22,586 words) and the complexity of Shakespearean language.

B. Prediction Examples

Table III shows examples of the model's ability to predict the next word for various Shakespearean prompts:

Additionally, the model could generate extended sequences by repeatedly predicting the next word. For example:

- **Input**: "to be or not to"
  Extended output: "to be or not to be your malice but you have so comfort"
- **Input**: "shall i compare thee to"
  Extended output: "shall i compare thee to be you from their face in a cardinal"

IV. DISCUSSION

A. Prediction Quality Analysis

Our model successfully learned to predict contextually appropriate continuations for Shakespearean text. For the famous opening of Hamlet's soliloquy "to be or not to," the model correctly predicted "be" with high confidence (17.46%), demonstrating its ability to capture well-known phrases from the corpus.

However, for some other famous quotes, the model did not always predict the historically accurate continuation. For example, "all the world's a" should continue with "stage" from "As You Like It," but the model predicted "king." Similarly, "now is the winter of our" should continue with "discontent" from Richard III, but the model predicted "hand." This suggests that while the model captures grammatical patterns well, it may not always recognize specific famous quotations, especially if they appear infrequently in the corpus.

B. Model Limitations and Challenges

Several factors affected the model's performance:

1) **Vocabulary Size**: Shakespeare's works contain many rare words and unique phrases. Managing this large vocabulary while maintaining computational efficiency was challenging.

TABLE III
EXAMPLE PREDICTIONS AND CONFIDENCE SCORES

Input Prompt	Top Predicted Word	Confidence
”to be or not to”	”be”	17.46%
”all the world’s a”	”king”	5.64%
”now is the winter of our”	”hand”	1.17%
”if music be the food of”	”the”	5.79%
”the quality of mercy is”	”a”	11.99%
”to sleep perchance to dream”	”to”	11.50%
”alas poor yorick i knew him”	”to”	7.29%
”romeo romeo wherefore art thou”	”a”	11.14%
”shall i compare thee to”	”be”	13.55%

2) **Data Sparsity**: Despite having over 1.6 million words, many specific word combinations occur infrequently, making it difficult for the model to learn certain patterns reliably.
3) **Complex Syntax**: Shakespeare’s sentence structures often involve complex inversions and archaic forms that are difficult for the model to capture consistently.
4) **Memory Constraints**: Training a model with a large vocabulary on full sequences required careful memory management, including batch processing and sparse cross-entropy loss.

C. Architectural Improvements

Throughout the development process, we made several key improvements to the model architecture:

1) **Bidirectional Processing**: Adding bidirectional LSTM layers significantly improved the model’s ability to understand context from both directions.
2) **Increased Network Depth**: Moving from a two-layer to a three-layer LSTM architecture improved the model’s capacity to capture complex patterns.
3) **Sequence Length Optimization**: Increasing the input sequence length from 5 to 8 words provided more context for accurate predictions.
4) **Specialized Vocabulary Management**: Adding specific Shakespearean terms to the vocabulary ensured the model could handle key phrases.

D. Loss Function Analysis

We chose sparse categorical cross-entropy as our loss function, which proved crucial for the model’s performance. This loss function offers several advantages over regular categorical cross-entropy:

1) **Memory Efficiency**: It avoids one-hot encoding the target words, which would be prohibitively memory-intensive with our large vocabulary of 22,586 words.
2) **Computational Performance**: Working directly with class indices rather than sparse matrices accelerated training.
3) **Mathematical Equivalence**: It provides the same gradients as categorical cross-entropy but with better numerical stability.

This choice allowed us to handle the large vocabulary while maintaining training stability.

V. CONCLUSION

In this paper, we have presented a word-level LSTM model for predicting the next word in Shakespearean text. Our model successfully learns patterns from Shakespeare’s unique writing style and generates contextually appropriate continuations for input prompts.

Key findings include:

1) Bidirectional LSTM layers significantly improve the model’s ability to capture Shakespeare’s complex sentence structures.
2) Careful vocabulary management is crucial for balancing between coverage of Shakespeare’s rich vocabulary and model performance.
3) A sequence length of 8 words provides sufficient context for accurate next-word prediction in most cases.
4) The sparse categorical cross-entropy loss function enables efficient training with large vocabularies.

The final model achieved 8.97% validation accuracy on a vocabulary of 22,586 words, which is promising given the complexity and uniqueness of Shakespearean language. The model successfully eliminated unknown token predictions while maintaining grammatically correct and stylistically appropriate continuations.

Future work could explore incorporating attention mechanisms, transformer architectures, or character-level modeling to further improve prediction accuracy. Additionally, fine-tuning the model on specific plays or character dialogues could enable more targeted text generation.

VI. PROMPTS

The model was tested with the following Shakespearean prompts:

- ”to be or not to” (from Hamlet)
- ”all the world’s a” (from As You Like It)
- ”friends romans countrymen lend me” (from Julius Caesar)
- ”now is the winter of our” (from Richard III)
- ”if music be the food of” (from Twelfth Night)
- ”the quality of mercy is” (from The Merchant of Venice)
- ”to sleep perchance to dream” (from Hamlet)
- ”alas poor yorick i knew him” (from Hamlet)
- ”romeo romeo wherefore art thou” (from Romeo and Juliet)

- ”shall i compare thee to” (from Sonnet 18)

REFERENCES

[1] I. Landuzgun, “Next Word Prediction Using LSTM with TensorFlow,” Medium, April 2023. [Online]. Available: https://medium.com/@ilaslanduzgun/next-word-prediction-using-lstm-with-tensorflow-e2a8f63b613c

[2] H. Singh, “Next-Word-Prediction-using-LSTM,” GitHub repository, 2022. [Online]. Available: https://github.com/Hritikahere/Next-Word-Prediction-using-LSTM/blob/main/Next$_{w}ord_{p}rediction.ipynb$