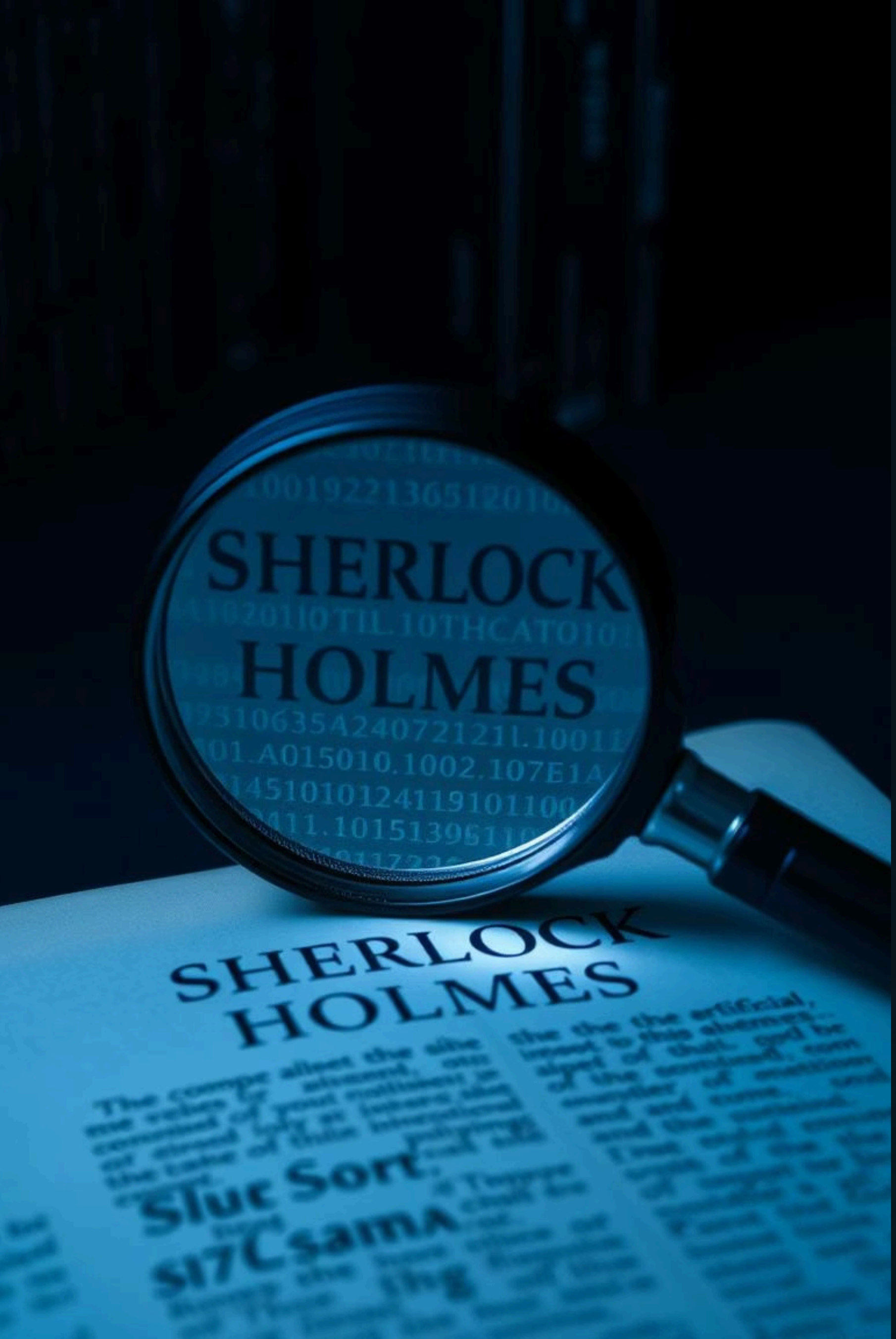


Tokenization Impact on Language Models

This presentation compares word-level and subword-level (BPE) tokenization effectiveness in predicting the next token using an LSTM-based language model. We will explore the dataset, model architecture, training setup, and key results from our analysis.

M by Mai Mohd

Dataset and Preprocessing

Dataset Details

- Source: Kaggle, Sherlock Holmes.txt
- Size: 610.92 kB
- Language: Classic 19th-century English
- Preprocessed Word Count: ~536,000 words
- Unique Words: 7,901
- Total Sentences: 7,278

Data Preprocessing

- Removed special characters, digits, and excessive whitespace.
- Lowercased all text (approx. 1.5% words removed).
- Tokenization performed at:
  - Word-level (Keras Tokenizer)
  - Subword-level (SentencePiece BPE, 8000 merge operations)

Model Architecture and Training

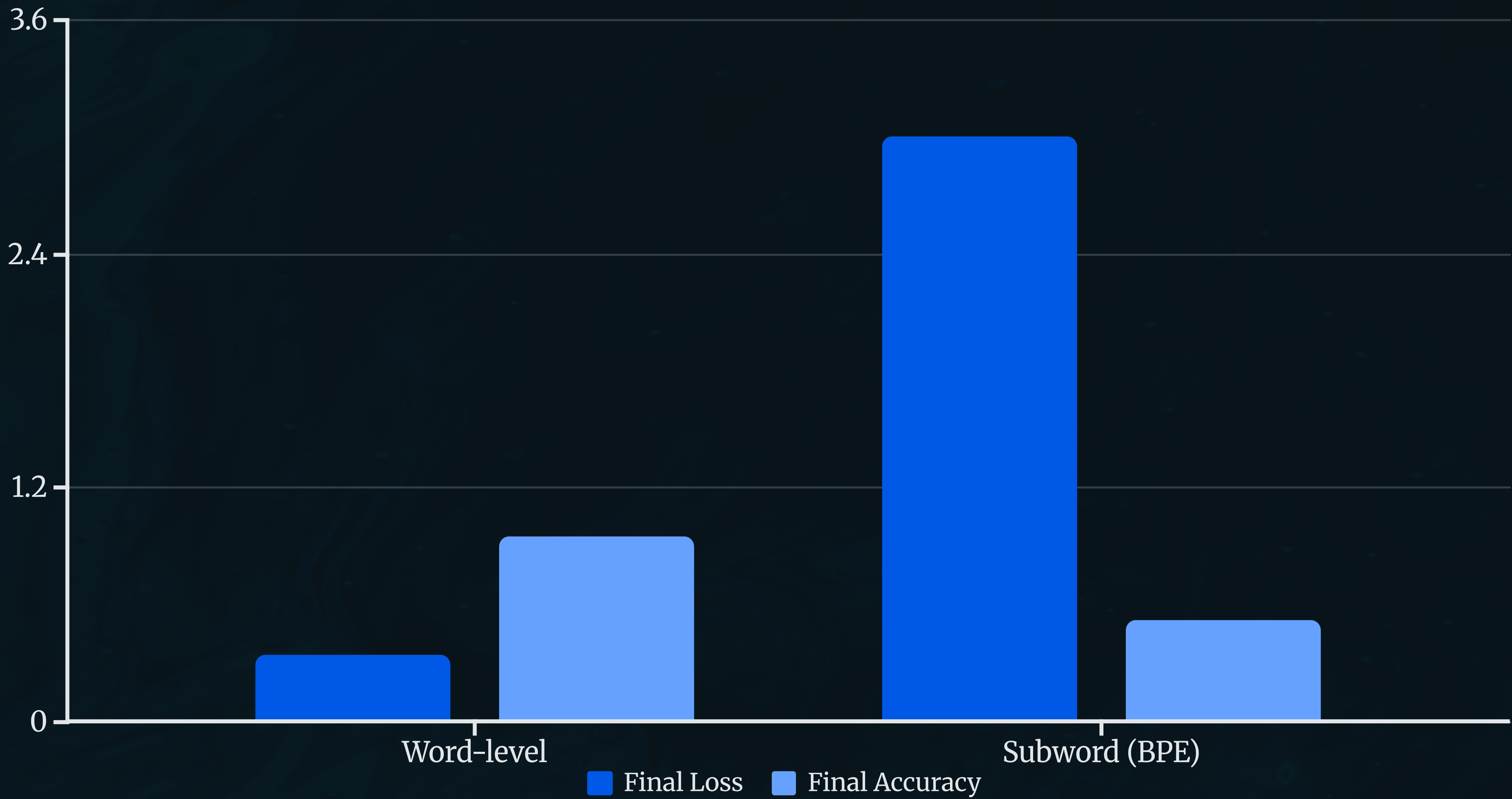
Model Architecture

- Model Type: Bidirectional LSTM
- Embedding Dimension: 100
- Hidden Units: 128
- Dropout: 0.2
- Loss Function: Categorical Crossentropy
- Optimizer: Adam
- Epochs: 20

Training Setup

Both models were trained on input-output pairs formed using a sliding window over the token sequences, predicting the next token given the previous ones.

Results and Conclusion



Word-level tokenization significantly outperformed BPE, achieving a final accuracy of 0.95 compared to BPE's 0.52. This highlights how tokenization method impacts model performance, especially with text like Sherlock Holmes that aligns well with full-word processing. While BPE is useful for rare words, word-level proved superior here.