Approach

The Sherlock Holmes corpus was used to train an LSTM-based model for next-word prediction. A total of approximately 500,000 individual tokens were processed with a unique vocabulary of around 15,000 words. Word-level tokenization was implemented using regex patterns to extract tokens, punctuation, and whitespace.

The text was split into sequences of 40 tokens, with the model trained to predict the subsequent word given an input sequence. A hybrid architecture was designed combining a single-layer LSTM to capture sequential dependencies with a single-head additive attention mechanism to focus on relevant parts of the input sequence. The model leverages pre-trained sentence embeddings from HuggingFace's all-MiniLM-L6-v2 model rather than training embeddings from scratch.

The training setup used the Adam optimizer with OneCycleLR scheduling, gradient clipping and early stopping with epoch patience to prevent overfitting. The model was trained for up to 100 epochs with a batch size of 256. Key evaluation metrics included cross-entropy loss, accuracy, and perplexity, providing comprehensive assessment of both predictive accuracy and text generation performance.

Metrics:

	Accuracy	Perplexity	Loss
Train	92.85%	1.46	0.4296
Validation	79.9%	2.98	1.0916
Test	80.43%	3.02	

Approach Comparison (v0 vs v1) :

v0 Approach (Initial Implementation):

- Attention Mechanism: Multi-head attention allowing simultaneous attention to different representation subspaces, but resulted in increased complexity, compute requirements, and pattern memorization
- Tokenization Strategy: Subword tokenization using SentencePiece for better handling of rare words and OOV words

- Embedding Approach: Trained embeddings from scratch to optimize representations specifically for the given text corpus
- Loss Computation: Calculated loss only on the last timestep, resulting in limited context for weight updates and poor generalization
- Training Complexity: Higher computational overhead with longer training times due to architectural complexity

v1 Approach (Current Implementation)

- Attention Mechanism: Single-head attention computing unified attention weights across the sequence, providing simpler and more stable learning
- Tokenization Strategy: Word-level tokenization using regex patterns, simpler but sufficient for clean literary text due to limited corpus
- Embedding Approach: Leveraged pre-trained embeddings from **all-MiniLM-L6-v2**, providing rich semantic representations without extensive training
- Loss Computation: Utilized all timesteps in sequences for loss calculation and weight updates, enabling better contextual learning and generalization
- Training Efficiency: Reduced computational requirements with faster convergence and more stable training metrics

Performance Impact

The v1 approach achieved superior results (**80.43%** test accuracy, **3.02** perplexity) primarily due to:

- Better contextual learning through full-sequence loss computation
- High-quality pre-trained embeddings providing semantic knowledge that the limited corpus lacked
- More stable training with simpler attention mechanism and proven optimization techniques

On the other hand in the v0 approach , training accuracy plateaued at **~25%** and validation at **~21%** indicating very limited learning.

Sample Generations (30-Word Generations)

Example 1

Prompt: "I asked"

Generated: i asked upon him. "perhaps it looks against me how well?' "yes, mr-dozen, but what about them when something had left you for the purpose of our

Example 2

Prompt: "The silence of"

Generated: the silence of the room was arthur outside who held round it yesterday night and smoking him to death within the centre of the hall with great mystery

Example 3

Prompt: "Every precaution has"

Generated: every precaution has to explain some terrible mysterious subject because he should vanish little for the purpose of our art in great life, waiting for some interesting

Example 4

Prompt: "Informed me"

Generated: informed me that i have done nothing as well and ill-natured man when something had left you. now for it was an inquiry from the american problem

Example 5

Prompt: "The assistant answered it"

Generated: the assistant answered it. we have had an american problem. " it was an inquiry? " gasped the man. " who gave an interesting one from your

Example 6

Prompt: “He looked at the letter and said”

Generated: He looked at the letter and said she glanced at seeing it up for half wages,

Prediction Trace:

```
Step 1 | Top 5 predictions: [(' ', '0.986'), (',', '0.010'), ('.', '0.003'), ('?', '0.000'), ('at', '0.000')]
Step 2 | Top 5 predictions: [('he', '0.362'), ('the', '0.188'), ('at', '0.137'), ('that', '0.111'), ('and',
'0.045')]
Step 3 | Top 5 predictions: [(' ', '0.999'), (',', '0.001'), ('.', '0.000'), ('!', '0.000'), ('?', '0.000')]
Step 4 | Top 5 predictions: [('glanced', '0.516'), ('looked', '0.189'), ('at', '0.040'), ('earnestly',
'0.031'), ('said', '0.028')]
Step 5 | Top 5 predictions: [(' ', '1.000'), ('.', '0.000'), ('glanced', '0.000'), ('"', '0.000'), ('at',
'0.000')]
Step 6 | Top 5 predictions: [('at', '0.689'), ('across', '0.205'), ('down', '0.013'), ('looking', '0.009'),
('again', '0.008')]
Step 7 | Top 5 predictions: [(' ', '1.000'), ('.', '0.000'), (',', '0.000'), ('looking', '0.000'), ('-', '0.000')]
Step 8 | Top 5 predictions: [('seeing', '0.576'), ('the', '0.193'), ('looking', '0.118'), ('it', '0.019'),
('whoever', '0.018')]
Step 9 | Top 5 predictions: [(' ', '1.000'), ('-', '0.000'), (',', '0.000'), ('.', '0.000'), ('it', '0.000')]
Step 10 | Top 5 predictions: [('it', '0.582'), ('the', '0.157'), ('me', '0.098'), ('him', '0.043'), ('an',
'0.010')]
Step 11 | Top 5 predictions: [(' ', '0.983'), ('.', '0.009'), (',', '0.008'), ('!', '0.000'), ('?', '0.000')]
Step 12 | Top 5 predictions: [('at', '0.349'), ('up', '0.086'), ('nearly', '0.071'), ('the', '0.066'),
('about', '0.051')]
Step 13 | Top 5 predictions: [(' ', '0.982'), (',', '0.017'), ('.', '0.001'), ('!', '0.000'), ('now', '0.000')]
Step 14 | Top 5 predictions: [('at', '0.482'), ('the', '0.136'), ('nearly', '0.076'), ('for', '0.074'),
('from', '0.043')]
Step 15 | Top 5 predictions: [(' ', '1.000'), ('.', '0.000'), (',', '0.000'), ('?', '0.000'), ('-', '0.000')]
```