**NLP Project: Next Token Prediction**

This project compares two tokenization methods for language modeling: word-level and subword-level (BPE).

I used **Sherlock Holmes.txt**, size: **(610.92 kB)**, Source: **Kaggle.** The dataset is a single plain-text file, containing classic 19th-century English prose, it offers a clean, well-punctuated narrative structure, so it’s ideal for language-model experiment.

The corpus holds 106032 words distributed across 7278 sentences, Vocabulary richness is moderate, with 7901 unique words.

After preprocessing and cleaning the data about 1.5% of the words were removed which gives us about 536K.

This project benchmarks two tokenization strategies for next-token language modelling. word-level tokens VS sub-word units produced by Byte-Pair Encoding (BPE)

For the first model I used **Keras Tokenizer** to convert words into tokens. Word-level Tokenizer (Keras) mapped each unique word to an integer ID.

For the sub-word experiment, I built a **SentencePiece** learned an 8000-merge vocabulary that splits rare words into frequent sub-units.

For both schemes, sliding **N-grams** were generated to create input-output pairs (context → next-token) pairs to train a model to predict the next word.

Both models were trained with LSTM architecture, they were fed into an identical model.

A single-layer bidirectional LSTM (embedding = 100, hidden = 128, dropout = 0.2) was trained with categorical-cross-entropy, using Adam optimizer and train for 20 epochs.

Word-level loss fell from 6.6 → 0.27. Accuracy rose from 0.06 → 0.95 over the 20 epochs, confirming steady learning despite the large vocabulary.

BPE tokenizer loss fell from 6.6 → 3. Accuracy rose from 0.05 → 0.52 over the 20 epochs.

This project showed how the way we break down text (tokenization) can really affect how well a model predicts the next word. The word-level method gave the best results because it works directly with full words, which fits well with the dataset. The BPE method didn’t perform as well, but it’s still useful for handling rare or unknown words.