Deep Neural Networks
HW3 - The one with the Transformer

Adrian Dinu Urse

1. Data Preparation and Processing

The dataset provided consisted of lines from plays, where each line included metadata such as the play's name and the speaker. I built my corpus from the provided dataset, resulting in a list of lines with the following format:

<PLAY=Henry IV> <SPEAKER=WESTMORELAND> Without much shame retold or spoken of.

I preserved the metadata (<PLAY> and <SPEAKER>) as part of each line to ensure that the model could generate text contextually based on the play and speaker.
The lines were then tokenized using either a **character-level tokenizer** or a **subword-level tokenizer** (GPT-2)

Dataset Splits:

- The dataset was randomly shuffled to prevent any bias in the splits.
- The corpus was split into 80% for training, 10% for validation, and 10% for testing.

Special Tokens:

- Added <SOS> (start of sentence), <EOS> (end of sentence), and <PAD> (padding) tokens to the tokenizer vocabulary for processing input and target sequences.
- Input sequences were prefixed with <SOS> and target sequences were suffixed with <EOS>.

Padding and Truncation:

- Both input and target sequences were padded or truncated to a fixed max_len

2. Model Architectures

- **Small Model**
  - **Hidden Dimension**: 256
  - **Number of Layers**: 4
  - **Number of Attention Heads**: 4
  - **Feedforward Dimension**: 1024 (4× hidden dimension)

- **Medium Model**
  - **Hidden Dimension**: 384
  - **Number of Layers**: 6
  - **Number of Attention Heads**: 6
  - **Feedforward Dimension**: 1536 (4× hidden dimension)

- **Large Model**
  - **Hidden Dimension**: 512
  - **Number of Layers**: 6
  - **Number of Attention Heads**: 8
  - **Feedforward Dimension**: 1024 (4× hidden dimension)

3. Training Parameters for the Models

Small Model (Character-Level Tokenizer)

- **Optimizer**: Adam with a learning rate of 1e-3.
- **Loss Function**: Cross-Entropy Loss with ignore_index set to the <PAD> token ID, ensuring that padded tokens do not contribute to the loss calculation.
- **Batch Size**: 32.
- **Number of Epochs**: 50.

Large Model (Character-Level Tokenizer)

- **Optimizer**: AdamW with a learning rate of 1e-4 and a weight decay of 0.01 for regularization.
- **Loss Function**: Cross-Entropy Loss with ignore_index set to the <PAD> token ID.
- **Batch Size**: 32.
- **Number of Epochs**: 150.
- **Learning Rate Scheduler**:
  - **Warmup Steps**: 10% of the total training steps were used for a linear warmup.
  - **Scheduler**: get_linear_schedule_with_warmup for gradually increasing and then decreasing the learning rate.

Small Model (Subword -Level Tokenizer)

- **Optimizer**: Adam with a learning rate of 1e-3.
- **Loss Function**: Cross-Entropy Loss with ignore_index set to the <PAD> token ID, ensuring that padded tokens do not contribute to the loss calculation.
- **Batch Size**: 32.
- **Number of Epochs**: 20.
- **Learning Rate Scheduler**:
  - **Warmup Steps**: 10% of the total training steps were used for a linear warmup.
  - **Scheduler**: get_linear_schedule_with_warmup for gradually increasing and then decreasing the learning rate.

Medium Model (Subword-Level Tokenizer)

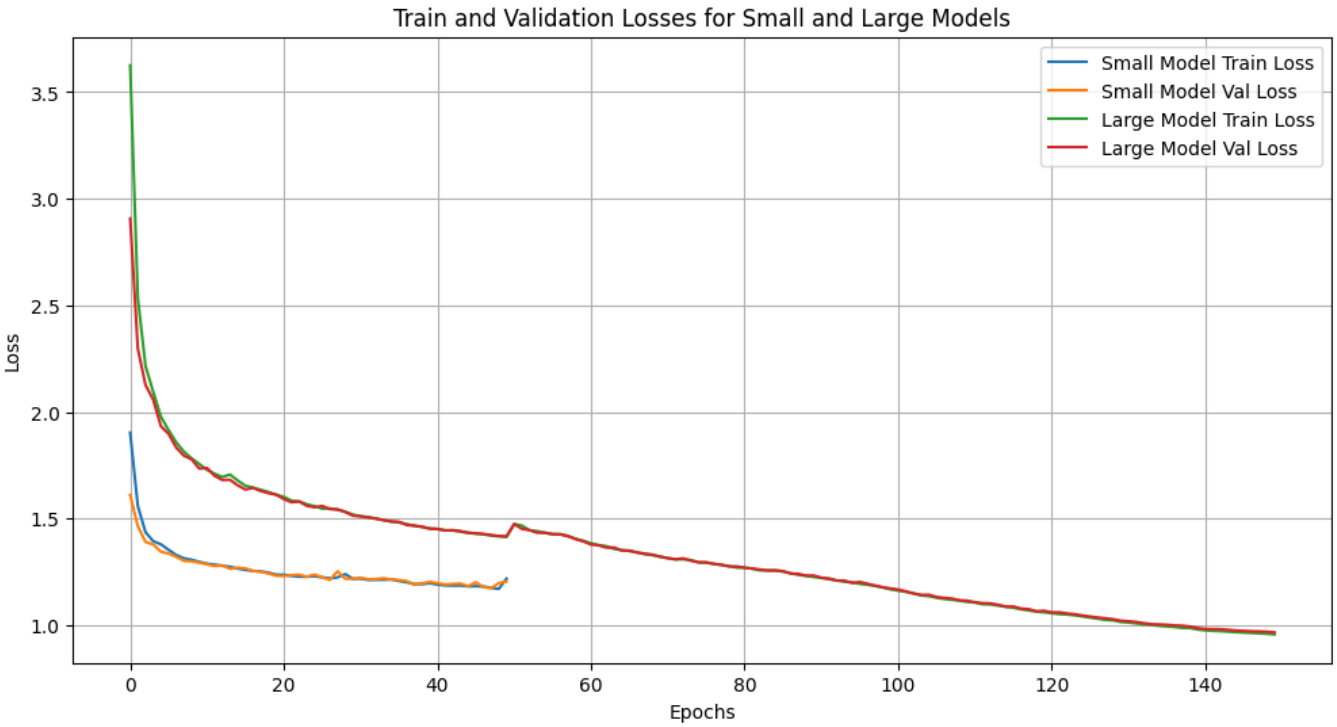
- **Optimizer**: AdamW with a learning rate of `1e-4` and a weight decay of `0.01`.
- **Loss Function**: Cross-Entropy Loss with `ignore_index` set to the `<PAD>` token ID from the tokenizer.
- **Batch Size**: `16` (reduced batch size due to larger computational requirements for the subword tokenizer).
- **Number of Epochs**: `8`.

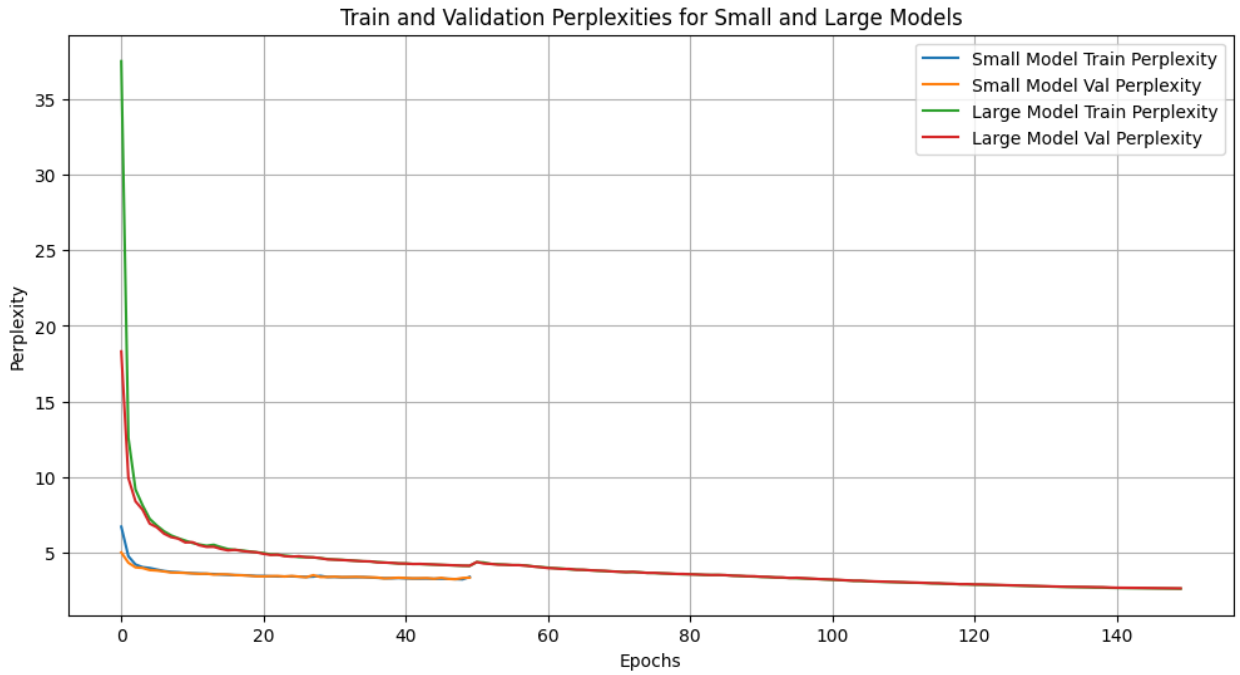
Large Model (Subword -Level Tokenizer)

- **Optimizer**: AdamW with a learning rate of 1e-4 and a weight decay of 0.01 for regularization.
- **Loss Function**: Cross-Entropy Loss with ignore_index set to the <PAD> token ID.
- **Batch Size**: 32.
- **Number of Epochs**:30.
- **Learning Rate Scheduler**:
    - **Warmup Steps**: 10% of the total training steps were used for a linear warmup.
    - **Scheduler**: get_linear_schedule_with_warmup for gradually increasing and then decreasing the learning rate.

4. Results

Character level tokenizer models

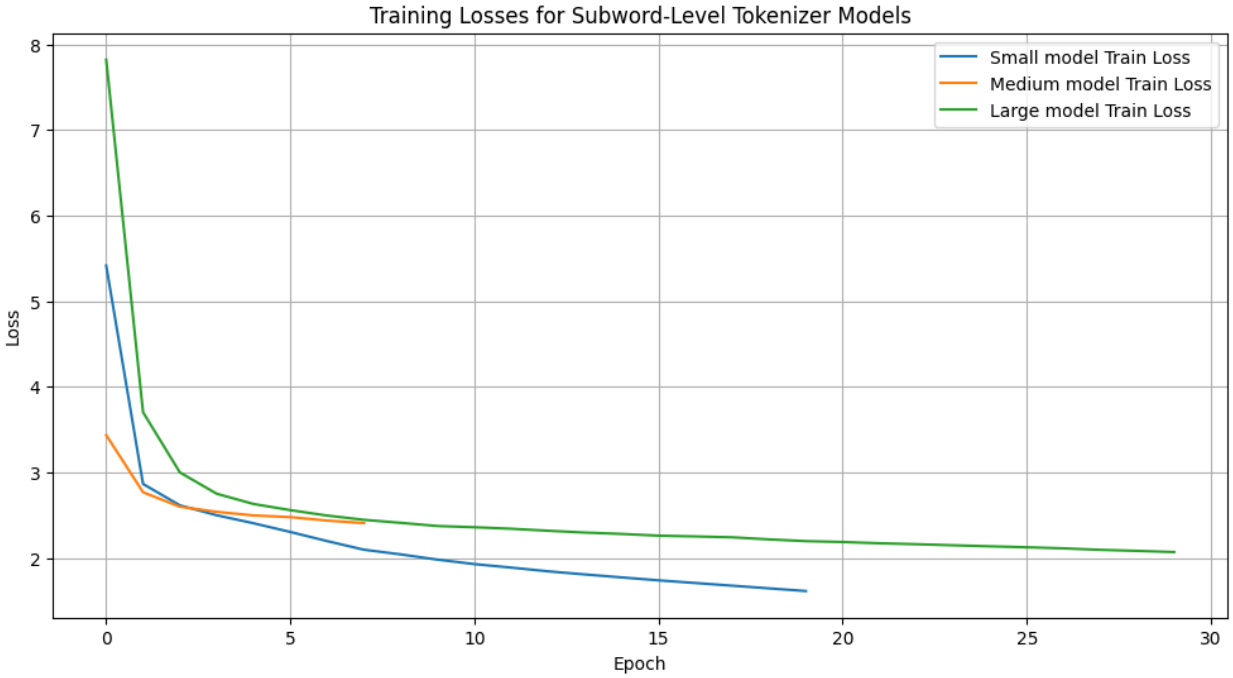


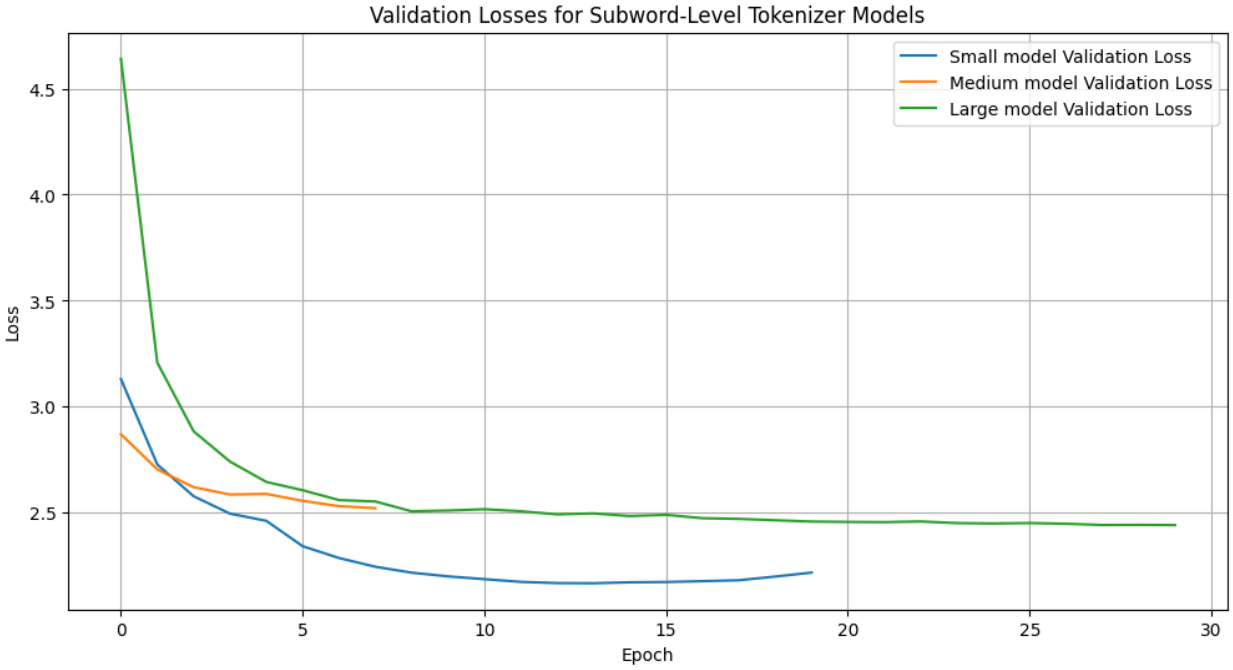


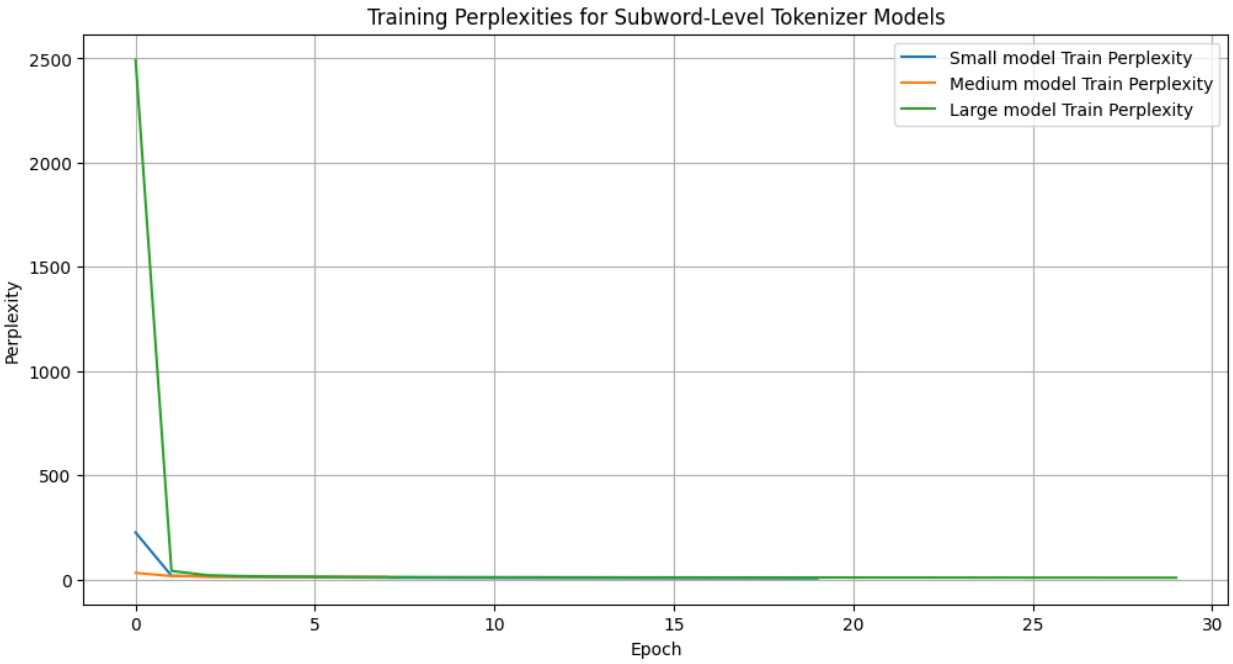
The **smaller model** with a learning rate of **1e-3** shows rapid initial loss reduction, outperforming the larger model during the first 50 epochs. However, its learning plateaus toward the end, struggling to further improve.

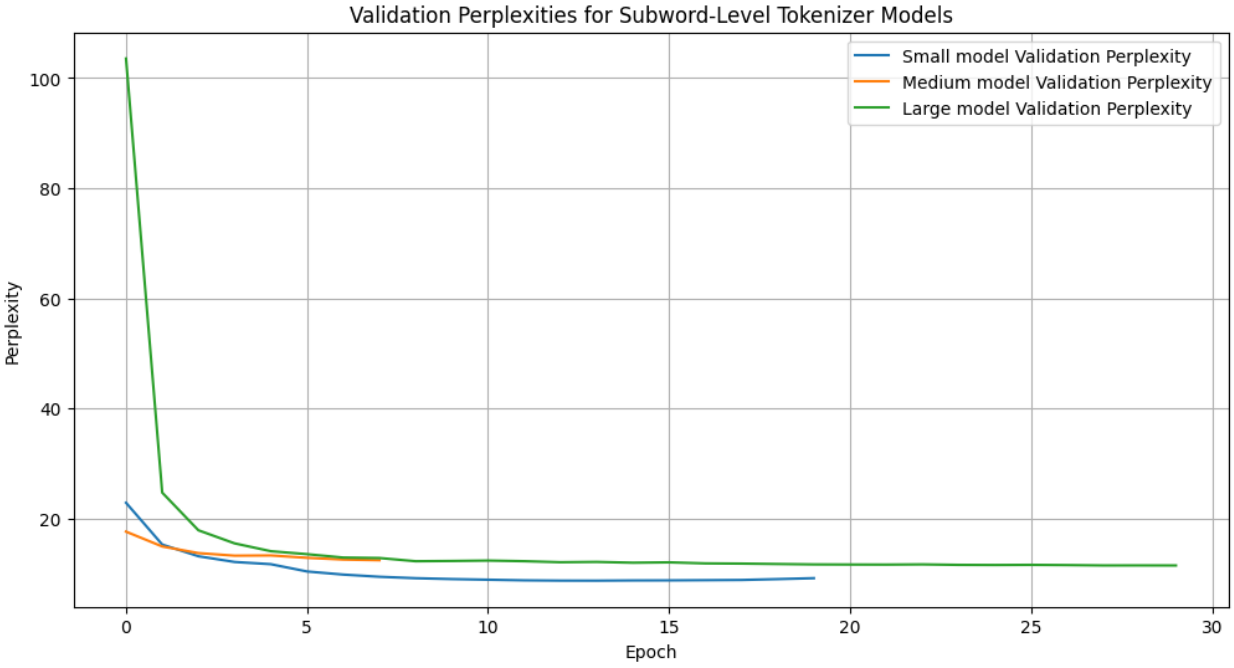
In contrast, the **larger model** with a lower learning rate of **1e-4** and a scheduler demonstrates steady loss reduction throughout training. By 150 epochs, it achieves lower final losses and perplexities, showcasing its greater capacity to learn and generalize over extended training. This highlights that smaller models are faster for short-term training, while larger models excel with longer training cycles due to their capacity and better convergence with lower learning rates.

Subword level tokenizer models






Training Perplexities for Subword-Level Tokenizer Models
Small model Train Perplexity
Medium model Train Perplexity
Large model Train Perplexity
Perplexity
Epoch
2500
2000
1500
1000
500
0
0
5
10
15
20
25
30


Validation Perplexities for Subword-Level Tokenizer Models
Small model Validation Perplexity
Medium model Validation Perplexity
Large model Validation Perplexity
Perplexity
Epoch
100
80
60
40
20
0
5
10
15
20
25
30

The **smaller model** with a learning rate of **1e-3** achieves a lower loss initially and converges faster than the medium and large models. However, towards the end of its learning it starts to overfit

.

The **medium** and **large** models, were trained using a learning rate of **1e-4**, showing they learn slower compared to the smaller model, achieving higher loss values and perplexity.

BLEU/ROUGE Comparison

Model	BLEU	ROUGE-1	ROUGE-2	ROUGE-L
Small (Character)	0.0043	0.029	0.0004	0.0277
Large (Character)	0.0064	0.037	0.0012	0.0343
Small (Subword)	**0.0081**	**0.042**	**0.0019**	**0.039**
Medium (Subword)	0.0055	0.038	0.00057	0.0359
Large (Subword)	0.0067	0.0401	0.00153	0.0376

Generally, subword tokenization outperforms character-level across all metrics, demonstrating its ability to better capture linguistic patterns. Larger models generally achieve higher BLEU and ROUGE scores, particularly with character-level tokenization. The small subword model achieves the highest BLEU and ROUGE scores.

Qualitative Samples

Input text: <PLAY=Loves <SPEAKER=FERDINAND> Labours Lost> Soft! whither away
Reference text: so fast?
Full line: <PLAY=Loves <SPEAKER=FERDINAND> Labours Lost> Soft! whither away so fast?
Generated text (small char model): <PLAY=Loves <SPEAKER=FERDINAND> Labours Lost> Soft! whither away, ind fa hof t,
Generated text (large char model): <PLAY=Loves <SPEAKER=FERDINAND> Labours Lost> Soft! whither away hand, spectan thou
Generated text (small subword model): <PLAY=Loves <SPEAKER=FERDINAND> Labours Lost> Soft! whither away, and,
Generated text (medium subword model): <PLAY=Loves <SPEAKER=FERDINAND> Labours Lost> Soft! whither away, my
Generated text (large subword model): <PLAY=Loves <SPEAKER=FERDINAND> Labours Lost> Soft! whither away,

---
Input text: <PLAY=As <SPEAKER=OLIVER> you like it> an end
Reference text: of him, for my soul, yet I know not why,
Full line: <PLAY=As <SPEAKER=OLIVER> you like it> an end of him, for my soul, yet I know not why,
Generated text (small char model): <PLAY=As <SPEAKER=OLIVER> you like it> an endowisth was. by, he wee tha st ieal commes
Generated text (large char model): <PLAY=As <SPEAKER=OLIVER> you like it> an endedeers faith-thowned, a bee them.
Generated text (small subword model): <PLAY=As <SPEAKER=OLIVER> you like it> an end,
Generated text (medium subword model): <PLAY=As <SPEAKER=OLIVER> you like it> an end the king, and,
Generated text (large subword model): <PLAY=As <SPEAKER=OLIVER> you like it> an end, by, sirUS

Input text: <PLAY=Henry <SPEAKER=MARGARET VI Part 2> JOURDAIN> By
Reference text: the eternal God, whose name and power
Full line: <PLAY=Henry <SPEAKER=MARGARET VI Part 2> JOURDAIN> By the eternal God, whose name and power
Generated text (small char model): <PLAY=Henry <SPEAKER=MARGARET VI Part 2> JOURDAIN> By ase, thee takinge at and,
Generated text (large char model): <PLAY=Henry <SPEAKER=MARGARET VI Part 2> JOURDAIN> By which such'd to the preleat her,
Generated text (small subword model): <PLAY=Henry <SPEAKER=MARGARET VI Part 2> JOURDAIN> By a prince,
Generated text (medium subword model): <PLAY=Henry <SPEAKER=MARGARET VI Part 2> JOURDAIN> By! and
Generated text (large subword model): <PLAY=Henry <SPEAKER=MARGARET VI Part 2> JOURDAIN> By,
---
Input text: <PLAY=Twelfth <SPEAKER=SIR Night> TOBY BELCH> Pourquoi, my
Reference text: dear knight?
Full line: <PLAY=Twelfth <SPEAKER=SIR Night> TOBY BELCH> Pourquoi, my dear knight?
Generated text (small char model): <PLAY=Twelfth <SPEAKER=SIR Night> TOBY BELCH> Pourquoi, my, sisand shea the s the.
Generated text (large char model): <PLAY=Twelfth <SPEAKER=SIR Night> TOBY BELCH> Pourquoi, my haven asst word.
Generated text (small subword model): <PLAY=Twelfth <SPEAKER=SIR Night> TOBY BELCH> Pourquoi, my niece, for my lord.
Generated text (medium subword model): <PLAY=Twelfth <SPEAKER=SIR Night> TOBY BELCH> Pourquoi, my.
Generated text (large subword model): <PLAY=Twelfth <SPEAKER=SIR Night> TOBY BELCH> Pourquoi, my niece,
---
Input text: <PLAY=Henry <SPEAKER=CADE> VI Part 2> Enter a
Reference text: Messenger
Full line: <PLAY=Henry <SPEAKER=CADE> VI Part 2> Enter a Messenger
Generated text (small char model): <PLAY=Henry <SPEAKER=CADE> VI Part 2> Enter athe owiccth, withithicheise s, th withat
Generated text (large char model): <PLAY=Henry <SPEAKER=CADE> VI Part 2> Enter all then to be with surpp. What, feach:
Generated text (small subword model): <PLAY=Henry <SPEAKER=CADE> VI Part 2> Enter a ship
Generated text (medium subword model): <PLAY=Henry <SPEAKER=CADE> VI Part 2> Enter a the, and the sun: the.

Generated text (large subword model): <PLAY=Henry <SPEAKER=CADE> VI Part 2> Enter aerer> Which, how to toe,, my, and, and, to

Character-Level Models (Small and Large):

- Struggle to generate coherent text, producing gibberish or nonsensical sequences.
- Larger models improve slightly but still fail to capture meaningful context due to the challenges of character-level tokenization.

Subword-Level Models (Small, Medium, Large):

- Generate more coherent and contextually aligned text.
- Small and medium models produce shorter, simpler outputs but lack completeness.
- Larger models handle context better, generating longer, more structured text, though issues with redundancy and accuracy persist.

Sampling Methods:

- With **argmax**, models tended to repeat themselves, leading to monotonous outputs.
- Using **top-k sampling** (k=10) with a **temperature of 0.7** resulted in better diversity and improved generation quality. All reported metrics were computed under these sampling conditions.

Subword tokenization outperforms character-level tokenization, enabling better text generation. Larger models improve coherence, but further refinement is needed for optimal results.