**Assignment Number 4**

**CS-4063 Natural Language Processing**

**Fall 2024**

**Instructions:**

* Submission will be **Online** with **Screenshots** of running code
* Late submissions are not allowed.
* Solve all the questions in the given order.
* Write the programs with clarity and add comments where necessary.
* Each question carries 10 marks.
* The evaluation will be viva based with binary marking.

**Title: Designing a Custom Transformer-Based LLM for Text Summarization**

**Objective:**  
In this assignment, you will design and implement your own Large Language Model (LLM) from scratch using a Transformer-based architecture to perform text summarization. Unlike the previous assignment, you will not use any pre-trained models and will be responsible for building the model architecture, training the model, and evaluating its performance.

**Public Dataset:**  
Use the following text summarization dataset. https://www.kaggle.com/code/lusfernandotorres/text-summarization-with-large-language-models/input

### Requirements:

#### 1. Designing the Transformer-Based LLM:

You are required to design and implement a custom transformer-based model for text summarization. The model should be built from scratch, meaning you should:

* Implement the Transformer architecture:
  + Implement the core components of the Transformer model, including the encoder, decoder, multi-head attention mechanisms, and position-wise feed-forward networks.
  + Implement positional encoding to preserve the sequence order of the input text.
  + Use a suitable number of layers (e.g., 6 or more layers) for both the encoder and decoder, with appropriate attention heads (e.g., 8 or more heads).
* Custom modifications (optional):
  + You may explore adding custom modifications or optimizations to the standard Transformer architecture to improve the summarization task, such as:
    - Custom attention mechanisms (e.g., hierarchical attention, sentence-level attention).
    - Variations of the feed-forward networks.
    - Techniques like layer normalization, dropout, or residual connections to improve model performance and generalization.

#### 2. Text Summarization:

* Training for Summarization Task:
  + Use the Transformer model to generate summaries from input text. Implement both extractive and abstractive summarization techniques, depending on your design.
  + Define the loss function for training (e.g., cross-entropy loss for sequence-to-sequence generation).
  + Ensure that the model is capable of learning to summarize long texts by leveraging the self-attention mechanism.

#### 3. Dataset:

* Perform data preprocessing, including tokenization, padding, and batching, as needed.

#### 4. Training the Model:

* Train the model using an appropriate optimizer (e.g., Adam, AdamW) and set up a learning rate schedule (e.g., warm-up followed by a decay).
* Use an appropriate batch size based on your computational resources and optimize the model’s hyperparameters.
* Implement techniques to avoid overfitting, such as early stopping or validation loss monitoring.

#### 5. Evaluation Metrics:

After training your model, evaluate its performance using the following metrics:

* Loss: Report the training and validation loss during the training process (e.g., cross-entropy loss).
* ROUGE Scores: Evaluate the quality of your generated summaries using ROUGE metrics (ROUGE-N, ROUGE-L).
  + ROUGE scores compare the overlap between the model’s generated summary and the reference summary, measuring recall, precision, and F1 scores.

Additionally, if possible, you may consider evaluating the quality of summaries manually based on:

* Relevance: Does the summary capture the main ideas of the text?
* Coherence: Is the summary logically structured and clear?
* Conciseness: Does the summary contain only the essential information?

#### 6. Sample Text and Summarization:

After training your model, provide:

* A sample input text from your dataset.
* The predicted summary generated by your custom Transformer model.
* Compare your model’s summary with the reference summary to analyze its quality.

#### 7. Analysis and Comparison:

* Results Comparison:
  + Report and analyze the loss during training and validation for the model.
  + Compare the ROUGE scores of your custom model with any baseline models or standard summarization systems (e.g., GPT-2, BERT-based summarizers).
* Summary Quality Evaluation:
  + Provide a comparative analysis of the predicted summaries in terms of coherence, relevance, and conciseness.
  + Highlight any potential areas of improvement for your model and any challenges faced during model design and training.