Assignment-1

Deconstructing the Transformer

Course: Foundation of Large Language Models (AI60213)

Due date: September-6-2025

[**Total marks: 20**]

Advisory

Using GenAI tools for solving assignment is discouraged as this will increase the probability of plagiarism with others who may choose the same path.

Submission Instructions

1. Only one member from each group needs to submit the assignment through moodle to avoid any duplicate submissions.
2. Submit the Solution file as: **Group_(number)_assignment_1_solution.ipynb** and a **.py** file as well.

Plagiarism Policy

1. Code plagiarism will be checked on the designated blocks which the students have to code.
2. No penalty till 30% (Baseline).
3. Each 10% increase over baseline will cost 1 mark, Beyond 70% absolute -2 (negative).
4. Two groups having similar code will be penalized with the same amount. No arbitration will be done.
5. Late submission policy: One week window with a penalty of 3 marks.

Setup Instructions

1. Use Google Colab for all experiments (free GPU tier is sufficient)
2. Install required libraries:

```
!pip install transformers torch matplotlib seaborn pandas numpy
!pip install bertviz  # For attention visualization
```

3. Use the provided starter code (`Assignment1.ipynb`)
4. Dataset: We'll use simple and small dataset, information will be available in the Colab notebook.

Questions

Part-1: Building Tiny Transformer Model from Scratch and do the following analysis.

[**Total: 5 marks**]

You are given a set of Implementation instructions, needs to be followed in order to implement Tiny Transformer and do the required Analysis.

Task to be Performed:

- Implement sinusoidal positional encoding [The Transformer has no notion of word order, so you need to inject positional information], and all the different modules like Multi-head attention, scaled-dot product attention, Feed-forward layer, Encoder, Decoder Layer. All the relevant information is given in the code notebook. [**1 mark**]
- Implement a Tiny Transformer with the following specifications: Embedding dimension: 128, Transformer Layers 2, Configurable number of attention heads (1,2,4..8 etc), Feed-Forward dim: 512, Max-Token Seq Length: 128.

  [**1 mark**]
- Train the `TinyTransformer` for appropriate number of epochs (approx 100-200) on the given dataset on code notebook for 'Language Translation Task'. [**1 mark**]
- Train multi head model (4 heads) and single head (1) model, by keeping the number of parameters same, adjust attention head dimension accordingly. [**1 mark**]
- Implement visualization for different attention types like Encoder self-attention, Decoder self-attention and Decoder cross-attention. Visualize the attentions for multi-head and single-head both for given test sentences. [**1 mark**]

Part-2: Architectural Ablation Studies

[**Total: 10 marks**]

This section explores the impact of key architectural components. For each task, you will modify your baseline 4-head model from Question 1, retrain it, and analyze the results.

Study the Role of Residual Connections:

- **The Impact of Removal:** Modify your Transformer architecture by removing all residual (or "skip") connections within the Encoder and Decoder layers. Train this modified model under the same conditions as your baseline. Plot the training and validation loss curves for both the baseline and the modified model on the same graph. Does the model still train effectively? [**2.5 marks**]
- **Learnable skip weights:** Instead of the `standard x + Sublayer(x)`, modify the connection to be `(w * x) + Sublayer(x)`, where `w` is a learnable scalar parameter for each layer. Initialize `w` to 1.0 and plot its value for each layer over the course of training. Does the model learn to down-weight the skip connection (identity path) in certain layers?

  [**2.5 marks**]
- **Long-Range Skip Connections:** In a standard Transformer, the output of layer L is the input to layer $L+1$. Modify this so that the input to layer $L+1$ is `Output(`L`) + Output(`$L-1$`)`. This creates a short-range `skip` over one layer. A more extreme version could be connecting the output of the first encoder layer directly to the input of the last encoder layer. Train this modified model under the same conditions as your baseline. Plot

the training and validation loss curves for both the baseline and the modified model on the same graph. Does the model still learns effectively? [**2.5 marks**]

Study the Role of Feed-Forward Layers:

[**2.5 marks**]

- Modify your `EncoderLayer` and `DecoderLayer` to completely bypass the FFN sub-layer. The output from the multi-head attention's `Add & Norm` step should now be the final output of the entire layer.
  1. Calculate and report the percentage reduction in total model parameters after removing the FFNs.
  2. Train this model. Plot its validation loss curve on the same graph as your baseline model.
  3. Discuss the performance difference. What is the role of the FFN in introducing non-linearity, and why is this critical for the model's learning capacity?

Part-3: Exploring Attention Modulation

[**Total: 5 marks**]

Token Distance as an Attention Bias:

- Propose a technique to modulate attention scores based on the relative distance between tokens. For example, you could create a distance-based penalty or bias that is added to the `pre-softmax attention scores` QK^T. Formulate the mathematical modification to the standard scaled dot-product attention equation. Formulate the mathematical modification to the standard scaled dot-product attention equation that implements your proposed technique. [**3 marks**]
- **Implementation and Evaluation:** Implement this new `"distance-aware"` attention mechanism in your baseline 4-head model. Retrain the model and evaluate its performance against the original baseline. Visualize the attention patterns from this new model. How do they differ from the patterns you observed in Question 1? [**2 marks**]