Report: Deep Learning Model for Derivative Prediction

Objective:
Design a deep learning model to predict the derivative of a function with respect to a specified variable. The model should be able to take the original function and predict its derivative. The primary focus is on showcasing machine learning capabilities while avoiding the usage of heuristic or rules.

Dataset:
The dataset is provided as a file named `train.txt` and is available on Google Drive. Each line of the file comprises a training sample alongside its ground truth result.

Sample Data:
For instance, given the input sample: `d(6exp^(9a)+4exp^(2a))/da`, the output is `54exp^(9a)+8exp^(2a)`, where the derivative is taken concerning variable `a`.

Model Architecture:
The chosen architecture is a Seq2Seq model, which consists of an Encoder and a Decoder module. The primary goal of the model is sequence-to-sequence prediction, which is an apt choice for translating the function into its derivative.

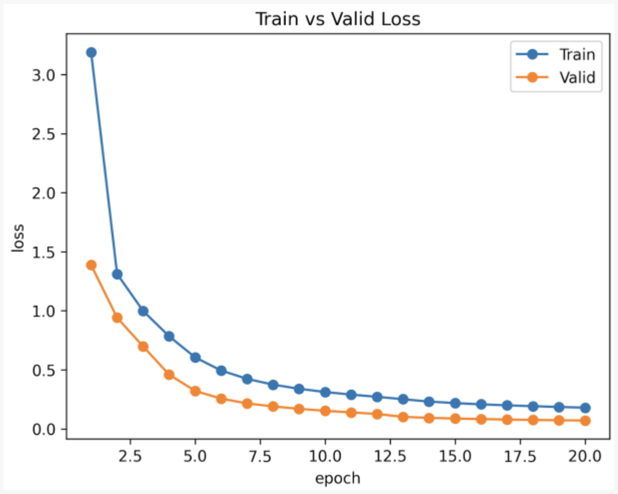
Model Design Choices:

1. **Network Size:**
   - The model consists of 434,133 trainable parameters. This size strikes a balance between model complexity and computational efficiency.
2. **Architecture:**
   - **Encoder:**
     - The encoder converts the input function into a fixed-size context vector.
     - It consists of token embeddings (`tok_embedding`) and position embeddings (`pos_embedding`).
     - Two encoder layers are used, each comprising self-attention mechanisms, position-wise feedforward networks, and layer normalization.
   - **Decoder:**
     - The decoder interprets the context vector to produce the derivative.
     - Like the encoder, it also has token and position embeddings.

    - Two decoder layers are incorporated. Each layer has self-attention, encoder-attention, position-wise feedforward networks, and layer normalization.
    - The final layer (`fc_out`) maps the output to one of the 1877 possible tokens.

3. **Regularization:**
    - Dropout of 0.3 is employed throughout the model, ensuring that the model doesn't overfit to the training data and generalizes well to unseen samples.
4. **Other Design Choices:**
    - **Embedding Sizes:** Both token and position embeddings are of size 64, ensuring the model captures the positional information of each token.
    - **MultiHeadAttention:** Used within both the encoder and decoder layers for capturing dependencies irrespective of their distance in the sequence.
    - **PositionwiseFeedforward:** Allows the model to focus on individual positions and apply a transformation to them.
    - **Criterion:** CrossEntropyLoss is used, which is suitable for classification tasks. In this case, predicting the next token in the sequence.

Loss Function Analysis: The model's training and validation losses were plotted over 20 epochs. Both losses decreased over time, indicating learning and improvement. The training loss showed a steep decline initially, slowing down as it progressed, while the validation loss remained consistently lower than the training loss throughout the epochs. The closeness of the validation loss to the training loss suggests that the model is not overfitting. By the 20th epoch, both losses appeared to converge, suggesting that further training might yield diminishing returns.



Model Performance: On a separate test set, the model demonstrates an accuracy of 81.887%. This level of accuracy, combined with the loss analysis, suggests that the model is performing well in terms of learning the relationships within the data and generalizing these to unseen data.

Conclusion: The Seq2Seq model, with its thoughtful architectural choices, successfully addresses the task of translating a function into its derivative. It demonstrates strong performance without overfitting, as evidenced by the close convergence of training and validation losses and a high accuracy rate on the test set. This model, with its robust regularization techniques and design choices, stands as a suitable candidate for derivative prediction tasks.