Deep Learning Q3 Report

Introduction

This report outlines the experimentation with different architectures and strategies to solve a sentiment classification task using the SST-2 dataset. The goal was to progressively enhance the model's ability to classify sentences as positive or negative. Starting with simpler architectures and progressing toward more advanced setups, the experiments aimed to evaluate the effectiveness of different strategies. Finally, a zero-shot classification approach was implemented using a pre-trained language model.

Steps in the Experiment

Step 1: Importing a Decoder-Only Model

- The initial step involved loading a pre-trained decoder-only language model (GPT-2) and its tokenizer.
- The SST-2 dataset was prepared and tokenized to create input data compatible with the model.

Step 2: Generating Outputs from the Model

- The model generated textual outputs for various inputs using different decoding configurations. These included hyperparameters like `temperature`, `top_k`, and `top_p`.
- While this step demonstrated the generative capabilities of GPT-2, it was not directly tied to the classification task.

Step 3: Loading and Preparing the SST-2 Dataset

- The SST-2 dataset, part of the GLUE benchmark, was loaded and tokenized. Padding and truncation were applied to ensure uniform input lengths, enabling efficient processing during training.

Step 4: Using the CLS Token for Classification

- The decoder's final layer was removed, and a classification head was added. The embedding of the first token (CLS token) was used as input for the classification layer.
- **Results**: This setup struggled with generalization, with modest training accuracy and a clear bias toward the positive class. Validation performance indicated limited capacity to distinguish between the classes effectively. The confusion matrix highlighted the model's tendency to over-predict positive samples while under-predicting negative ones.

Step 5: Adding a Linear Layer for Aggregation

- Instead of relying solely on the CLS token, a linear layer aggregated embeddings from the entire input sequence.

- **Results**: While the training performance slightly improved, validation accuracy declined, suggesting that the linear layer diluted useful contextual information. The model remained heavily biased toward the positive class, as reflected in its confusion matrix, which showed high false positive rates for the negative class.

Step 6: Introducing Bidirectional Attention

- A bidirectional multi-head attention layer replaced the linear layer. This mechanism allowed the model to learn relationships between all tokens in the input sequence, improving contextual understanding.
- **Results**: This architecture significantly enhanced performance. Both training and validation accuracy improved, and the confusion matrix showed better balance between classes. The model demonstrated robust generalization compared to earlier setups. The bidirectional attention mechanism effectively captured relationships across the entire sequence, reducing both false positives and false negatives.

Step 7: Implementing Unidirectional Attention

- Two variants of unidirectional attention were tested:
    - **Left-to-Right Attention**: The model only considered preceding tokens for each token in the sequence.
    - **Right-to-Left Attention**: The model only considered succeeding tokens for each token in the sequence.
- **Results**:
    - The **Right-to-Left Attention** setup achieved the highest validation accuracy among fine-tuned models, outperforming both Left-to-Right and Bidirectional setups. It demonstrated better balance in the confusion matrix, with reduced false positives and a slight increase in false negatives compared to the bidirectional model.
    - **Left-to-Right Attention** performed well but showed slightly more class imbalance compared to Right-to-Left, with higher false positive rates for the negative class.

Step 8: Zero-Shot Classification

- A pre-trained zero-shot classification pipeline was employed using the `facebook/bart-large-mnli` model.
- **Results**:
    - This approach achieved the highest overall accuracy without requiring any fine-tuning, showcasing the power of pre-trained models for general-purpose tasks.
    - The classification metrics were well-balanced, with high precision and recall for both classes. The model exhibited a slight bias toward the negative class, as indicated by its higher recall for negatives compared to positives.

Key Observations

1. **Performance Improvements**:

  - Each architectural enhancement improved the model’s ability to generalize.
  - Bidirectional and Right-to-Left attention mechanisms stood out as the most effective among the fine-tuned models.
2. **Class Balance**:
  - Simpler architectures (Steps 4 and 5) struggled with class imbalance, heavily favoring the positive class.
  - Attention-based models (Steps 6 and 7) and the zero-shot classifier (Step 8) addressed this issue effectively, with Right-to-Left Attention providing the best balance among the fine-tuned models.
3. **Zero-Shot Strength**:
  - The zero-shot accuracy was the highest, surpassing all fine-tuned models. This demonstrates the capability of pre-trained models to perform complex tasks without additional training.
  - Despite its lack of training, the zero-shot model produced a well-balanced confusion matrix and robust metrics across both classes.
4. **Training Efficiency**:
  - Fine-tuning required significant computational effort compared to the zero-shot approach, which achieved high performance with minimal setup.
  - However, fine-tuned models allowed for experimentation with task-specific architectural changes, demonstrating the benefits of attention mechanisms.

Conclusion

- **Bidirectional and Right-to-Left Attention** models demonstrated excellent fine-tuning performance, with the latter achieving the best results among the trained models.
- The **zero-shot classifier** achieved the highest overall accuracy with no training effort, showcasing the power of large pre-trained models.
- This task highlights the trade-off between fine-tuning complexity and zero-shot simplicity, with attention mechanisms providing robust intermediate solutions for sentiment classification.

Recommendations

1. **Use Zero-Shot Models**:
  - For quick deployment with minimal effort, zero-shot classification models like `facebook/bart-large-mnli` should be the go-to choice.
2. **Leverage Attention Mechanisms**:
  - When fine-tuning is necessary, attention-based architectures such as Bidirectional or Right-to-Left setups should be prioritized.
3. **Further Exploration**:
  - Explore hybrid models that combine fine-tuned attention mechanisms with pre-trained zero-shot models for enhanced performance.

4. **Address Class Imbalance**:
    - Use weighted loss functions or oversampling techniques to mitigate class imbalance in future experiments.

This report provides a comprehensive overview of the task, showcasing how various architectural choices affect performance and generalization in sentiment classification.