Detailed Report: Reducing Loss in Cosine Similarity Between Parent and Student Embeddings

Introduction

Core Idea

The primary goal of this project was to improve the efficiency of Large Language Models (LLMs) by reducing their precision, making them faster and more memory-efficient. This process is known as quantization. Specifically, we aimed to maintain the similarity between the original (parent) model's embeddings and those from the quantized (student) model. We used backpropagation, a fundamental technique in machine learning, to minimize the difference (loss) between the two sets of embeddings.

Definitions

Quantization

Quantization is a technique used in machine learning to reduce the precision of the model's parameters from higher-bit (like 32-bit) to lower-bit (like 8-bit) representations. This process helps in reducing the model size and improving inference speed without significantly sacrificing accuracy. It is particularly useful for deploying models on resource-constrained devices.

Backpropagation

Backpropagation is a key algorithm in training neural networks. It involves calculating the gradient of the loss function with respect to each weight by the chain rule, and then updating the weights to minimize the loss. This process helps the model learn from the errors and improve its predictions.

Llama.cpp

Llama.cpp is a library used for quantizing and optimizing models. It streamlines the process of converting model weights and activations to lower bit-widths, enhancing deployment speed and memory usage without sacrificing model accuracy. This makes it crucial for efficient model deployment in real-world applications.

Cosine Similarity

Cosine similarity measures the cosine of the angle between two vectors, providing a metric for their similarity. It ranges from -1 (completely different) to 1 (exactly the same). In this project, it was used to compare the similarity between the parent and student embeddings.

Comparing Parent and Student Models

The parent model represents the original, high-precision model, while the student model is the quantized, lower-precision version. Comparing their embeddings using cosine similarity helps ensure that the quantization process retains the essential characteristics and performance of the original model.

Experiment Setup

Embedding Generation

1. **Parent Embeddings**: Generated using the `BAAI/bge-large-en-v1.5` model.
2. **Student Embeddings**: Generated using the quantized version of the model.

Cosine Similarity Calculation

Cosine similarity measures how similar two vectors (embeddings) are, ranging from -1 (completely different) to 1 (exactly the same). We computed the mean cosine similarity between the parent and student embeddings.

Optimization

We used backpropagation to minimize the Mean Squared Error (MSE) loss between the cosine similarities of the parent and student embeddings. The `llama.cpp` library facilitated the quantization and embedding generation processes.

Performance Timings

Below are the timings recorded for each operation:

1. **Load Time**: Time taken to load the model.
2. **Sample Time**: Time taken to sample tokens.
3. **Prompt Evaluation Time**: Time taken to evaluate the prompt tokens.
4. **Total Time**: Overall time taken for the operation.

Example timing results:

–yaml

```
llama_print_timings:        load time =    4917.40 ms
llama_print_timings:      sample time =       0.00 ms /     1 runs
llama_print_timings: prompt eval time =    4901.16 ms /   104 tokens
llama_print_timings:       total time =    4916.77 ms /   105 tokens
```

Results

Despite running the optimization for multiple epochs, the loss remained constant, indicating that the optimization was not effective.

Loss results for each epoch:

1. **Epoch 1**: Loss = 1.3021
2. **Epoch 2**: Loss = 1.3021
3. **Epoch 3**: Loss = 1.3021
4. **Epoch 4**: Loss = 1.3021
5. **Epoch 5**: Loss = 1.3021
6. **Epoch 6**: Loss = 1.3021
7. **Epoch 7**: Loss = 1.3021

Analysis

Several factors might have contributed to the constant loss:

1. **Learning Rate**: The learning rate might have been too low.
2. **Optimizer Choice**: The Adam optimizer may not have been the best choice.
3. **Initialization**: The initial embeddings might not have been optimal.
4. **Model Complexity**: The quantized model might have lost significant information.
5. **Gradient Flow**: Potential issues with gradient flow during backpropagation.

Conclusion

Summary

The experiment aimed to reduce the loss between the cosine similarity of the parent and student embeddings through backpropagation, but it was not successful. The loss remained constant across multiple epochs, indicating ineffective optimization.

Recommendations

1. **Hyperparameter Tuning**: Experiment with different learning rates and optimizers.
2. **Model Analysis**: Analyze the quantized model to understand information loss.
3. **Advanced Techniques**: Explore techniques like knowledge distillation or transfer learning.

Further research and experimentation are needed to achieve the desired optimization.