LORA QLORA

Article

https://www.analyticsvidhya.com/blog/2023/08/lora-and-qlora/

IMPLEMENTATION'

Simple LoRA Implementation.ipynb

1. LoRA (Low-Rank Adaptation of Large Language Models)

Proposed Methods:

- **Motivation and Objective:**
  - Fine-tuning large language models (LLMs) traditionally requires modifying and retraining all parameters, which is computationally expensive and requires substantial memory. LoRA aims to reduce this overhead by introducing a method that focuses on updating only a small subset of parameters, making the fine-tuning process both resource-efficient and faster.
- **Low-Rank Adaptation Concept:**
  - LoRA's fundamental idea is based on the observation that the weight updates required during fine-tuning can be represented as low-rank updates. Instead of adjusting the full matrix of weights, LoRA proposes to approximate these updates by decomposing them into two smaller matrices, A and B.
  - **Mathematical Formulation:**

        - Suppose W is the weight matrix in a Transformer model. Instead of directly fine-tuning W, LoRA models the change in weights as ΔW=A×BT\Delta W = A \times B^TΔW=A×BT, where A and B are much smaller matrices with a predefined rank r.
        - This rank r is typically chosen to be much smaller than the dimensions of W, making A×BTA \times B^TA×BT a low-rank approximation of the desired weight update.
- **Efficiency and Benefits:**
    - **Parameter Reduction:**
        - In large models like GPT-3, which has 175 billion parameters, updating the entire weight matrix would require immense computational power. LoRA reduces the number of trainable parameters by orders of magnitude (often by a factor of 10,000), dramatically lowering the hardware requirements for fine-tuning.
    - **Training Process:**
        - During training, the original model weights are frozen. The low-rank matrices A and B are the only components that are learned, which significantly reduces the computational load.
    - **Adaptability:**
        - This method is adaptable to a variety of NLP tasks, showing strong performance comparable to full fine-tuning, despite the drastic reduction in trainable parameters.

Architecture:

- **Modification to Transformer Layers:**

    - LoRA is applied at the Transformer layer level, where each weight matrix in the Transformer's linear layers is augmented with a low-rank decomposition.
    - **Integration of Low-Rank Matrices:**
        - In each linear layer of the Transformer, instead of adjusting the weight matrix W, LoRA modifies the output using the low-rank approximation W+A×BTW + A \times B^TW+A×BT.
    - **Practical Implementation:**
        - The low-rank matrices A and B are typically initialized with random values and then fine-tuned on the specific task data. Since A and B have a significantly smaller size than W, this approach reduces both the memory footprint and the number of operations required during the forward and backward passes.
    - **Memory and Computation Benefits:**
        - Because the original weights are frozen and only the low-rank matrices are updated, the approach drastically reduces the amount of memory needed to store the model's parameters during training. Additionally, the computational burden is lessened since matrix multiplications involving smaller matrices are much less resource-intensive.
- **Application Scenarios:**
    - LoRA has been demonstrated to work effectively in tasks like text classification, translation, and summarization, where it matches or surpasses the performance of traditional fine-tuning methods with a fraction of the computational cost.

---

2. QLoRA (Quantized Low-Rank Adaptation)

Proposed Methods:

- **Motivation and Objective:**
    - Building on the efficiency gains of LoRA, QLoRA introduces quantization to further reduce memory usage and computational requirements. The method is particularly beneficial when adapting extremely large models on hardware with limited memory, such as GPUs with smaller VRAM.
- **Combining Low-Rank Adaptation with Quantization:**
    - QLoRA retains the low-rank adaptation method from LoRA but applies it to a quantized version of the model. Quantization involves representing model weights with lower precision, typically using fewer bits per parameter, thus significantly reducing the memory footprint.
    - **Quantization Details:**
        - QLoRA employs 4-bit quantization, where model weights, which are normally stored as 16-bit or 32-bit floating-point numbers, are reduced to 4-bit integers. This drastic reduction in precision allows the model to fit into smaller memory spaces.
        - **Quantization Process:**
            - The quantization process involves mapping the full-precision weights to a much smaller set of possible values, represented in a 4-bit space. This mapping is typically non-uniform, focusing on preserving the most important aspects of the weight distribution.
- **Adaptation with Low-Rank Matrices:**
    - Similar to LoRA, QLoRA introduces low-rank matrices AAA and BBB to adapt the quantized model to new tasks. These matrices are trained on top of the quantized weights,

ensuring that the adaptation process remains efficient even in the quantized domain.

    - **Training Process:**
        - During training, the quantized model weights are frozen, and only the low-rank matrices AAA and BBB are updated. This approach enables efficient fine-tuning even on hardware with limited memory capacity.
- **Efficiency Gains:**
    - **Memory Efficiency:**
        - By quantizing the model weights to 4-bit precision and only training low-rank matrices, QLoRA drastically reduces the memory required to store the model during training. This is especially advantageous for deploying large models on consumer-grade GPUs or in cloud environments with limited resources.
    - **Computational Efficiency:**
        - The reduction in precision also means that the computational cost of operations like matrix multiplications is lower, further speeding up the training process. Despite the lower precision, QLoRA maintains a high level of performance on NLP tasks.

Architecture:

- **Integration of Quantization and Low-Rank Adaptation:**
    - QLoRA modifies the architecture of large language models by integrating low-rank adaptation with quantized weights. The model is first quantized to 4-bit precision, and then the low-rank matrices are introduced to fine-tune the model.
    - **Quantized Transformer Layers:**
        - In QLoRA, each Transformer layer operates on quantized weight matrices. The low-rank matrices

AAA and BBB are added to these quantized layers, allowing the model to adapt to specific tasks while operating in a lower precision mode.

    - **Fine-Tuning Process:**
        - The fine-tuning process in QLoRA involves updating the low-rank matrices while keeping the quantized weight matrices fixed. This approach ensures that the model remains memory-efficient throughout the training process.
- **Quantization Challenges:**
    - One of the key challenges in QLoRA is maintaining model accuracy while operating on lower precision weights. The design of QLoRA carefully balances the trade-offs between quantization-induced performance degradation and the efficiency gains in memory and computation.
    - **Error Mitigation:**
        - Techniques such as scaling and careful initialization of the low-rank matrices help mitigate the potential accuracy loss due to quantization. The architecture is designed to be robust to the noise introduced by quantization, ensuring that the model's performance remains competitive.
- **Real-World Applications:**
    - QLoRA is particularly well-suited for scenarios where large models need to be deployed in resource-constrained environments, such as edge devices or instances with limited GPU memory. The method has shown strong performance in tasks like machine translation, question answering, and text summarization.

Both **LoRA** and **QLoRA** represent significant advancements in the field of model fine-tuning, particularly in the context of large language models. While LoRA focuses on reducing the number of trainable parameters through low-rank adaptation, QLoRA takes this a step further by integrating quantization, making it possible to adapt even the largest models on constrained hardware resources. These methods are instrumental in democratizing access to powerful LLMs, allowing them to be fine-tuned and deployed in a wider range of environments.