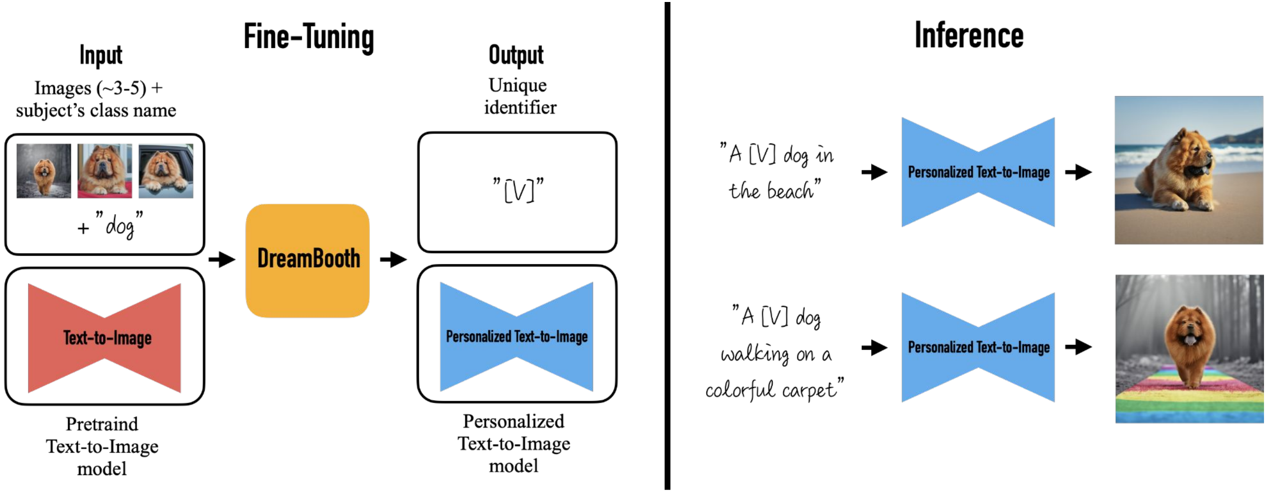
Hardware-Aware Neural Network Optimization

Mid-Term Project Review | MSML-605 | Spring 2025
Group-21

Harshit Singh
Krishna Vamsi
Prakhar Tiwari

Overview and Motivation



- Generative AI (LLMs & text-to-image diffusion models) demands high compute resources
- Fine-tuning these models on custom data increases resource usage
- Need to deploy across various hardware (CPUs, GPUs, TPUs)
- Motivation: Optimize performance and hardware efficiency

Problem Statement

CPU

GPU

TPU

- **Goal**: Benchmark different parameter-efficient fine-tuning strategies by varying different hardware resources.

- Compare efficiency and fidelity across hardware resources

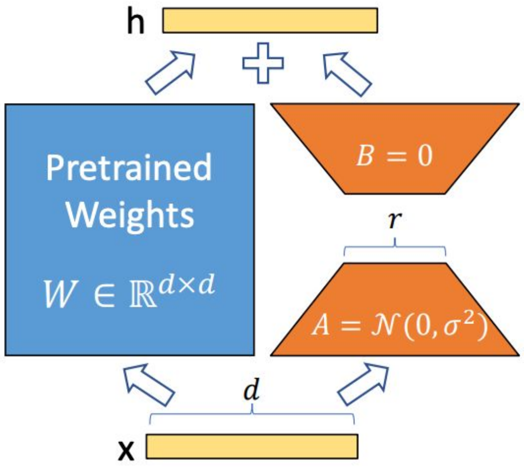
- **Analyze trade-offs**:
  - Training time and inference latency
  - Memory and compute utilization
  - Fidelity metrics (BLEU/ROUGE for LLMs; FID/CLIP-I/CLIP-T for diffusion models)

Proposed Approach

- **Apply popular methods**:
  - LoRA and QLoRA for low-rank adaptation
  - INT8/INT4 Quantization, quantization-aware training

- Implement on open-source LLMs (e.g., LLaMA, Mistral) & diffusion models (Stable Diffusion v2.1, SDXL)

Fine-Tuning Techniques



- **LoRA & Q-LoRA**:
  - Adapt LLMs with minimal extra parameters
  - Efficient fine-tuning even with limited resources

  ΔW = BA

    - B: d * r matrix
    - A: r * k matrix
    - r: rank, much smaller than d and k (e.g., 4, 8, 16)
    - ΔW: low-rank approximation of weight update

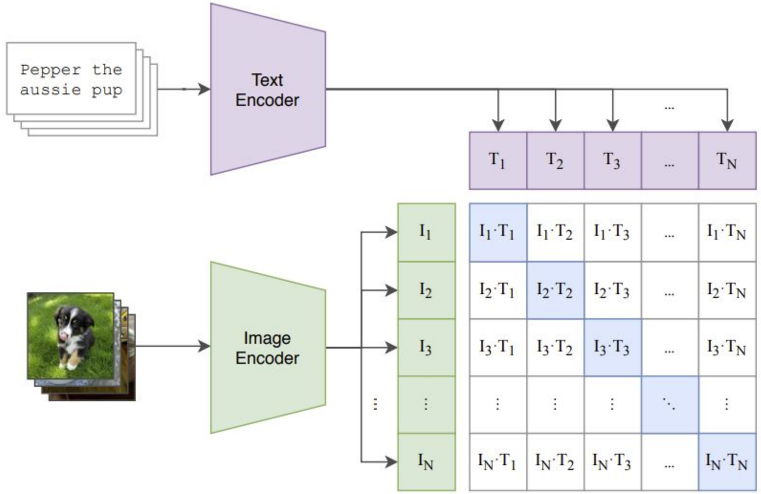
- **Quantization Methods**:
  - Post-training quantization
  - Quantization-aware training for INT8/INT4 precision
  - Reducing precision (e.g., from F16 to INT8) cuts storage significantly but comes at the cost of performance, requiring a balance between the two.

Hardware Environments for Benchmarking

- **CPU Clusters**: High-Performance Computing (HPC)
- **Nvidia GPUs**: V100 and A100 performance comparisons
- **TPUs**: Evaluated via cloud platforms (Google Cloud / Colab)
- Focus on tailoring fine-tuning for optimal resource use per hardware type

Benchmarking & Evaluation Metrics

- **Training Time**: Speed of fine-tuning processes
- **Inference Latency**: Response time during model deployment
- **Resource Utilization**: Memory and compute monitoring
- **Fidelity Metrics**:
    - LLMs: BLEU/ROUGE scores
    - Diffusion Models: FID, CLIP-I, CLIP-T scores



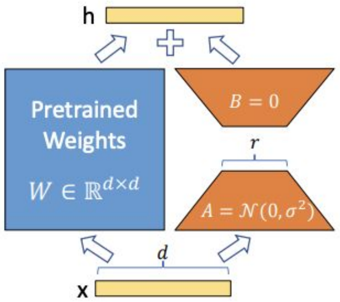
Programming Tools & Frameworks

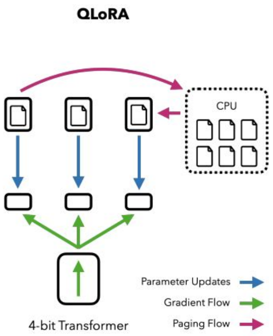
- **Languages & Libraries**:
  - Python, PyTorch
  - Hugging Face Transformers & Diffusers
- **Optimization Libraries**:
  - LoRA, Q-LoRA modules
  - Hugging Face optimization libraries for quantization
- **Monitoring**:
  - Weights & Biases for logging performance metrics

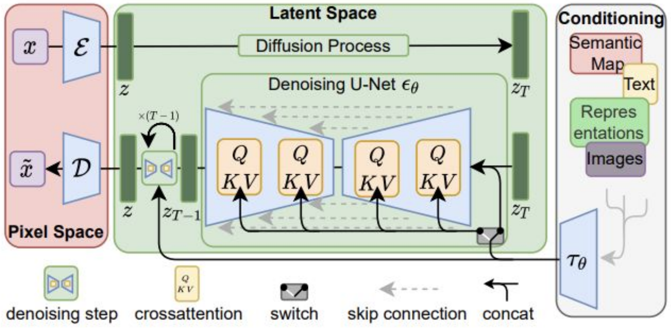
Weights & Biases

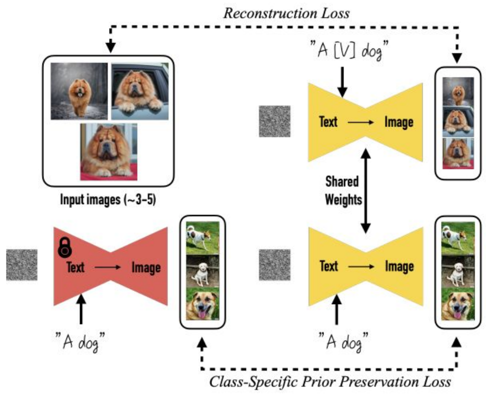
Literature Survey & References

- Hu et al., "LoRA: Low-Rank Adaptation of Large Language Mod [Link]
- Dettmers et al., "QLoRA: Efficient Fine Tuning of Quantized LLMs [Link]
- Rombach et al., "High-Resolution Image Synthesis with Latent Diffusion Models[Link]
- Ruiz et al., "DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation [Link]









Thank You