Proposed Solution!

We propose a novel solution to reduce memory access latency by reducing redundant classic convolutional layers and applying integer quantization without significant loss of performance.

State of the art supervised learning models capture a hierarchical representation of image features using series of convolutional layers. These layers each take in an image tensor as an input, a three-dimensional array that typically consists of height, width, and image channels. Learnable filters or kernels are then slid over input tensors, performing pixelwise multiplication to capture specific features from tensors, which produces feature maps.

Through the use of PyTorch, an open-source machine learning Python library, we plan on existing layer ablation techniques, evaluating the redundancy of layers based on its performance, the features it captures, and its computation cost. Layer ablation involves first training the original neural network model on a given dataset. Then, a specified layer is ablated or pruned, which may be a setting of a layer's outputs to zero or completely removing a layer. After removing a layer, the model can be fine-tuned again to recover some lost accuracy. The performance of the model with and without the layer is benchmarked using procedures specified below.

Additionally, we propose the use of integer quantization to make models more memory efficient and faster to execute. Integer quantization is the reduction of precision of a model's parameters from floating-point numbers to integers. Floating-point numbers are typically represented by 32-bits and can be converted into integers of reduced bit width, typically 8 bits with varying loss of performance, which conserves memory. These parameters are converted into integers through multiplication by scale factors, which are learned. To reduce the loss of accuracy caused by integer quantization, a quantization-aware training process can be applied. This means that scaling factors are treated as learnable parameters and are updating during the training process to minimize loss.

We plan on applying integer quantization to the weights and activations of the aforementioned convolution layers, the loss functions of the genarator and discriminator of the Conditional Generative Adversarial Networks (GAN), activation functions, such as Leaky Rectified Linear Units, and batch normalization parameters. We would start by training the neural network on the given dataset with full precision (32-bit floating point numbers). Then, we would quantize the weights and activations by manually converting floating-point parameters into integers. Finally, we would fine-tune the quantized model on the original dataset with the aim of recovering loss of accuracy. This newly quantized model would also be benchmarked using processes outlined below.