Collaborative Optimizations

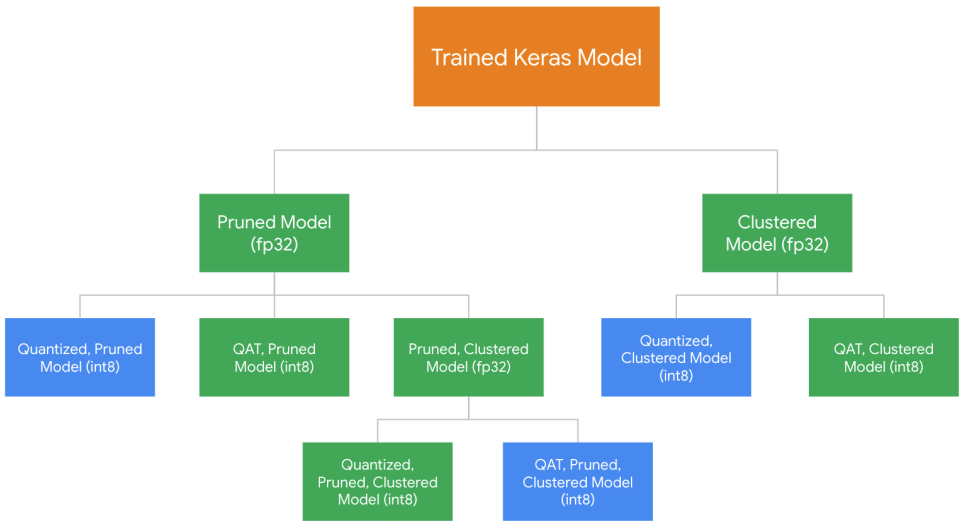
About This Reading

Previously, we have talked about optimizing our models for size by pruning our models, performing weight clustering, quantizing our weights from 32-bit floating point values to 8-bit integer values, and also touched on quantization-aware training (QAT). However, trying to perform all of these in such a way that we are able to optimize our model performance without removing the benefits of past optimizations is difficult. In this reading we will explore Collaborative optimization, a relatively new technique that allows us to get the best of both worlds, optimizing for multiple characteristics simultaneously.

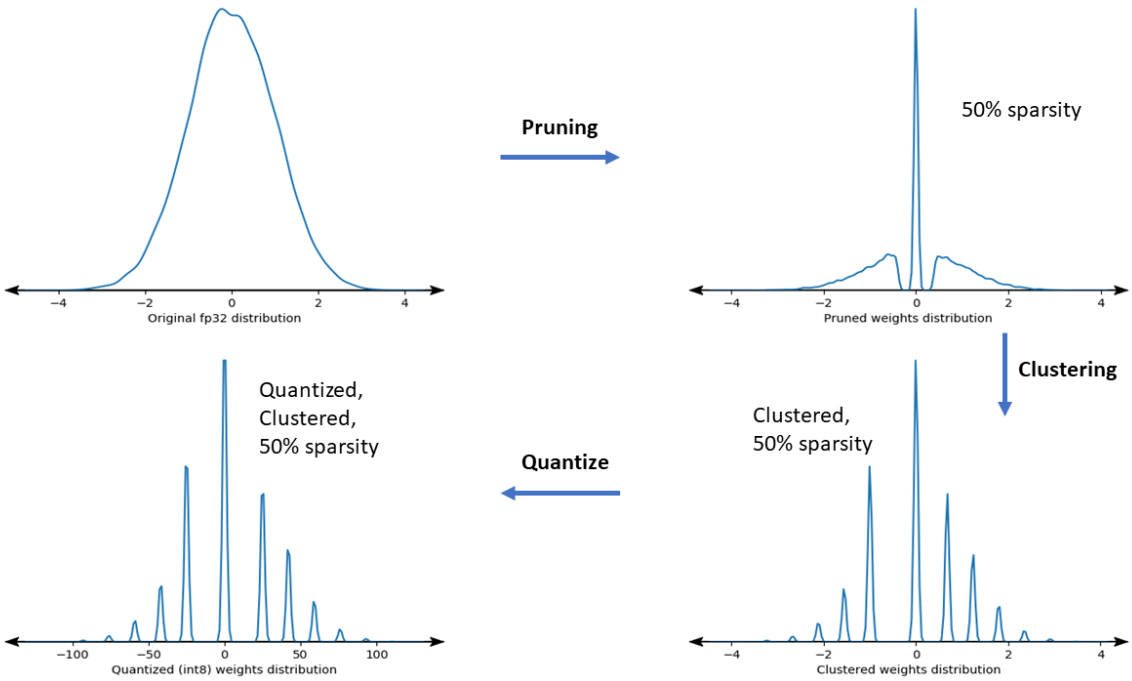
Overview

Collaborative optimization takes the optimization techniques that we were using previously and chains them together in such a way that the advantages gained by applying one method are preserved upon the application of another method. Previously, we might apply weight pruning (flattening weights that were close to zero) followed by weight clustering (clustering the existing weights into a few weights such as small, medium, and large weights). However, in applying the clustering, we might inadvertently remove some of the advantages of the pruning stage. Collaborative optimization takes this into account by introducing optimization stages that are preservable. The optimizations offered by TensorFlow currently are:

- Sparsity-preserving clustering
- Sparsity-preserving quantization aware training (PQAT)
- Cluster preserving quantization aware training (CQAT)
- Sparsity and cluster preserving quantization aware training



Using these optimization techniques in a specific sequence (outlined in the above image) allows the compression of a machine learning model to be “optimally” optimized. For example, applying sparsity-preserving clustering after pruning a model helps to prevent the loss of performance gained from the pruning stage. The leaf nodes in the above diagram are deployment-ready models, green nodes represent stages that require retraining or fine-tuning before deployment, and the red border represents collaborative optimization steps.



Using this optimization tree may not be required for a specific problem. For example, where model performance is already sufficient from a pruned and quantized model, we don’t need to consider PCQAT or CQAT as an alternative. A suitable procedure might be to go through the standard optimization pathway, see if this is sufficient and, if not, proceed with collaborative optimization pathways. Alternatively, all pathways could be tried and the output model with optimal characteristics is selected for deployment.

Summary

While collaborative optimizations may not always be necessary, knowing that they exist might be helpful for applications that are particularly demanding or where optimizing metrics is very important, such as if the model is being deployed on a large number of devices.