Sample paper

The following is from:

http://papers.nips.cc/paper/4687-large-scale-distributed-deep-networks.pdf

Introduction

Deep learning and unsupervised feature learning have shown great promise in many practical applications. State-of-the-art performance has been reported in several domains, ranging from speech recognition [1, 2], visual object recognition [3, 4], to text processing [5, 6]. It has also been observed that increasing the scale of deep learning, with respect to the number of training examples, the number of model parameters, or both, can drastically improve ultimate classification accuracy [3, 4, 7]. These results have led to a surge of interest in scaling up the training and inference algorithms used for these models [8] and in improving applicable optimization procedures [7, 9]. The use of GPUs [1, 2, 3, 8] is a significant advance in recent years that makes the training of modestly sized deep networks practical. A known limitation of the GPU approach is that the training speed-up is small when the model does not fit in GPU memory (typically less than 6 gigabytes). To use a GPU effectively, researchers often reduce the size of the data or parameters so that CPU-to-GPU transfers are not a significant bottleneck. While data and parameter reduction work well for small problems (e.g. acoustic modeling for speech recognition), they are less attractive for problems with a large number of examples and dimensions (e.g., high-resolution images). In this paper, we describe an alternative approach: using large-scale clusters of machines to distribute training and inference in deep networks. We have developed a software framework called DistBelief that enables model parallelism within a machine (via multithreading) and across machines (via 1 message passing), with the details of parallelism, synchronization and communication managed by the framework. In addition to supporting model parallelism, the DistBelief framework also supports data parallelism, where multiple replicas of a model are used to optimize a single objective. Within this framework, we have designed and implemented two novel methods for large-scale distributed training: (i) Downpour SGD, an asynchronous stochastic gradient descent procedure which leverages adaptive learning rates and supports a large number of model replicas, and (ii) Sandblaster L-BFGS, a distributed implementation of L-BFGS that uses both data and model parallelism.1 Both Downpour SGD and Sandblaster L-BFGS enjoy significant speed gains compared to more conventional implementations of SGD and L-BFGS. Our experiments reveal several surprising results about large-scale nonconvex optimization. Firstly, asynchronous SGD, rarely applied to nonconvex problems, works very well for training deep networks, particularly when combined with Adagrad [10] adaptive learning rates. Secondly, we show that given sufficient resources, L-BFGS is competitive with or faster than many variants of SGD. With regard to specific applications in deep learning, we report two main findings: that our distributed optimization approach can both greatly accelerate the training of modestly sized models, and that it can also train models that are larger than could be contemplated otherwise. To illustrate the first point, we show that we can use a cluster of machines to train a modestly sized speech model to the same classification accuracy in less than 1/10th the time required on a GPU. To illustrate the second point, we trained a large neural network of more than 1 billion parameters and used this network to drastically improve on state-of-the-art performance on the ImageNet dataset, one of the largest datasets in computer vision.