TensorFlow Tech Review

Creon Creonopoulos
CS410 Fall 2021

This paper is a brief Technology Review of the Machine Learning System called **TensorFlow**. TensorFlow is an open source project, with an enormous community and is also heavily used by a multitude of machine learning teams including several ones at Google. It has been developed to allow flexibility and automation for ML developers and is the leading ML system as of this technical review's writing. It is a successor of the highly used system, DistBelief[3], and was mostly developed by the same teams.

This system's architectural definition is as follows: "TensorFlow uses dataflow graphs to represent computation, shared state, and the operations that mutate that state."[1] This means it allows for different machines in a machine cluster to utilize their hardware to run operations on the same mutable state at the same time without the need for one operation to wait for another to complete. This allows developers to optimize different parts of the ML Algorithm they are trying to implement without having to wait for all operations to complete, before implementing an optimization to each operation.

The most important advantage of TensorFlow over other ML systems is that it allows for multiple processes to utilize smaller parts of the the overall data graph, for parallel execution and mutate the overall data with their results, giving each other updated, more recent and more accurate input on each algorithm they iterate through[1]. In my opinion, that is the most ingenious implementation piece of this system. Allowing for multiple processes to run on the same data set and mutate it dynamically for each other process to consume as fast as possible, not only reduces the amount of time for input data to be corrected and siphoned back into the training algorithms, but allows for the system to run on multiple different hardware, that feeds back into the central data graph.

The overall TensorFlow graph consists of vertices and edges. Vertices represent operations on data and each vertex's edge represents either input or output data. Data is represented as an "n-dimensional" array which is called a tensor. Tensors are also entered or removed from special queues that allow TensorFlow to stream them to and from different vertices, or even incorporate them in subgraphs of the overall graph. Through this process, each operation can be hosted on a different device (GPU or CPU), on any machine on the cluster, thus allowing operations to be specially written for the machine type they are being assigned to (i.e. a mobile device vs a computer gpu). TensorFlow is even capable of recognizing and deciding which device is better suited for each operation and assigning it appropriately.

Implementation wise, TensorFlow is built using C++, making it highly performant and portable. It does, however, expose multi language client libraries so that multiple different types of projects can interact with the core library [Figure 6]. Almost all major operating systems can interface with this system and run different kinds of algorithms, depending on the problem the application is trying to solve. The system's user input consists of a data graph and a steps definition, meaning the algorithms and their optimization techniques. Tensor flow then runs all sorts of optimizations on the graph, the steps and the distribution of operations throughout the hardware.

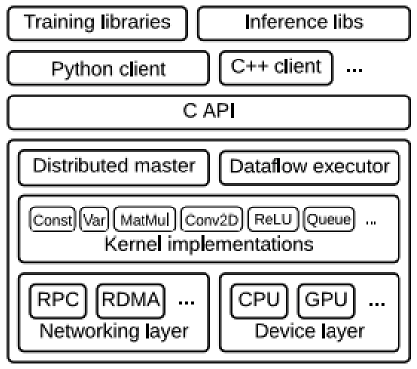

Figure 6: The layered TensorFlow architecture.

TensorFlow, although not the highest performing ML system in general, is one of the leading ML systems in two areas. Image Classification, meaning recognizing objects in photographs and Language Modeling in the sense of predicting what the next probable word is, given a sequence of words.

For Image Classification, the greatest advantage comes from the speed of processing training data. Being able to asynchronously read training images and update the data graph with results fast enough for each worker, TensorFlow is able to achieve marginally better performance than its competitor [1]. It is interesting to note here, that adding additional operation workers only helps up to a point, before the performance takes a hit due to more workers adding more traffic for the operation tasks. "Although adding 4 backup workers achieves the shortest overall step time (1.93 s), adding 3 achieves the highest normalized speedup (9.5%), and hence uses less aggregate GPU-time to reach the same quality." [1]. In fact, in my own professional career, I have utilized TensorFlow on mobile devices to allow for image recognition for governmental projects and it has had great success identifying objects it was trained to identify, even with small training samples.

As far as Language Modeling goes, the greatest use for TensorFlow comes from the need for speech recognition, predictive text and translation tasks. It's ability to use "distributed model parallelism"[2] allows the generation of word predictions to happen at significantly faster speeds. However, in contrast to Image Classification, instead of using more workers for more efficient data processing, the optimization comes from adding more operations per worker and improves the speed of throughput of data, maximizing its calculations. "Adding a second PS task is more effective than increasing from 4 to 32, or 32 to 256 workers" [1].

To recap, TensorFlows strengths come not only from its open source nature, allowing it to be optimized and expanded, but also from its unique architecture of distributed processing. It has been developed with the idea of allowing it to expand its capabilities and, given the opportunity, evolve into a system that can utilize machines that are not in use (which the world is in abundance of) to solve real world problems. Even though it's highest efficiency are focused around Image classification and Language Modeling, the system is not solely capable of only those two, giving it the opportunity to solve other ML problems and provide great advancements to technology and society as a whole.

References:

[1] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y. and Zheng, X., 2021. TensorFlow: A System for Large-Scale Machine Learning. [online] Usenix.org. Available at: <https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi> [Accessed 6 November 2021].

[2] A. Krizhevsky. One weird trick for parallelizing convolutional neural networks. arXiv preprint, 1404.5997, 2014. arxiv.org/abs/1404.5997.

[3] https://github.com/ucla-labx/distbelief