Ikhlas Attarwala
MSAI 339 – Data Science Seminar
11-07-18 – Dr. Jennie Rogers

Summarizing “TensorFlow: A System for Large-Scale Machine Learning” by Martín Abadi et al.

TensorFlow (TF) is an open-source library that was released by Google in late 2015. It was originally created for machine learning practices on tasks that required very numerical-heavy computations. The Google Brain team wrote a paper on the powerful and unparalleled system they had developed, starting with the limitations of their previous DistBelief, Google’s project from 2011 for training neural nets. They discussed how TF was an optimal system in heterogeneous environments (containing different kinds of hardware/software to work in tandem with one another), that it used dataflow graphs to depict computations and those computations’ current state, mapped nodes across many machines and multiple devices to allow for novel optimizations and experimentation, and non-traditionally allows nodes to access mutable input data, which was previously a restriction in dataflow systems. In terms of the paper, the authors begin with their motivation for this project by relating their interest in wanting to make a more flexible version of DistBelief, illustrated the execution of TF and its various jargon, their experience with training cases and fault tolerance, and comparisons with other systems that execute and compute machine learning methods.

As related to our class and recent lectures, if we were to take a look at large-scale computation ML methods such as deep learning, for example, TF would be an excellent tool to consider. In this case, while it not only has built-in support for the creation and use of neural nets, it also has employable mathematical functions that can be useful for them. This also assists with gradient-based ML algorithms. The versatility of TF is incredible as it allows you to modify the network’s structure and (activation) functions used for processing. We could also see great use of TF with other non-linear models to learn and make interpretations on data sets. I personally demo’d into TF’s embedding visualizer to examine higher-dimensional data-sets, and I can say with confidence this blows my mind!

I’m impressed by how they begin their paper with limitations. Prior to reading the content and simply knowing the index of the paper, there’s already an impression that the author is building off of known issues and providing solutions for it. This is a very strong stance of any paper. The authors do indeed do this by discussing what directions they wanted to head in after DistBelief. The article also does well to provide various suggestions and examples, which is helpful for new and experienced programmers. For example, when talking about training methods, they use phrases such as “To train a model on high-dimensional data, …it is *common* to use a distributed representation, …” (273), and this helps the reader because if they are unfamiliar to this concept, jumping straight into the uses of distributed representation may misdirect them into trying to learn by asking “what?” as opposed to quickly answering the “why”. There are many other strengths of this paper incl. the section on fault tolerance, or the various effectiveness and evaluation comparisons made, but their choice of example applications is arguably more interesting. In order to better market TF, they chose to attack the two points that stress the library to show its sturdiness. They run applications on image classification and language modeling to illustrate TF’s range of accommodations and assess its capacity for efficient learning.

While there aren’t many points that I can critique them on, I can say that from what I know so far on the topic of information science, it does seem like the paper is a little rich in detail, possibly more than is necessary. While they do a phenomenal job at providing a great background and, they glance over a lot of topics that almost appear like thoughts in someone’s head and disappear in topic from the entire paper immediately after. The following is a strength and weakness, but while I think it’s a great idea to discuss small scale projects when emphasizing the efficiency of larger scale works with TF, they only briefly touch the subject in section 6.1. Finally, while they enjoyed enlightening us with their motivation for moving on from DistBelief to TF, even so far as starting this article with the limitations of DistBelief, their conclusion only skims over a few problems users have such as the use of static dataflow graphs, and don’t consider any future possibilities of improvement.

It’s not such a great idea to end their paper on a less enthusiastic note (comparing conclusion to section 2.1), but they could have suggested a handful of long-term interests for the future potential of TF. I think a great option for further improvement would be to find a way to collect and implement data at incredible speeds. They said mobile devices can use TF, and if so, that means every person that has a smartphone can do amazing things with it on the go. If TF was more user-friendly and quicker to experiment on, simply on a platform as small as a smartphone, experimentation would skyrocket. Invention of some ingenious collection methods could really change the front of who could use TF and how quickly everyone could learn from it.