Data Intensive Systems for Machine Learning Assignment-2

Upinderjit Singh
ASU ID: 1217042463
Master's in Computer Science
Arizona State University

Abstract

Studying the structures and features of **TensorFlow and PyTorch** systems in detail[1][2]. Listing the benefits and shortcomings of the two systems. Also, stating the common features and difference among the two.

1 Why are we reviewing the two systems?

TensorFlow and PyTorch are the two most popular machine learning systems. Although there are many other reasonable systems but why these two are the most widely used? What is there internal structure which make them unique? What are the similarities between the two? These questions made me choose these two topics for this assignment.
Research paper[1] on TensorFlow talks about the previous system - DistBelief and how TensorFlow is more flexible than it. Later the paper talks about the execution model and extensible case studies.
Research paper[2] on PyTorch talks about how it caters both usability and speed. What are the various computational advancements which helped deep learning. Later the paper gives in site to the PyTorch's design.
This detailed description in both the research papers gain my interest.

2 Benefits and shortcomings of TensorFlow

TensorFlow is an open-source machine learning system which is developed by Google Brain.

2.1 Benefits:

- **Scalability**: TensorFlow supports both large-scale training and running the learned model (inference). Further, it is capable of training models using hundreds of GPUs and performing inference ranging from large distributed clusters to local mobile applications.
- **Mutable State Updates:** Allows vertices to represent computations to own or update mutable states along with functional computation on immutable data.
- **Graphs:** It has a better **computational graph visualizations** in comparison to other libraries. It offers embedding projector, a tool for interactively visualize embeddings.
- **Debugging:** Execution of graph's sub-part allows to easy debugging. Moreover, it offers its own debugger "tfdbg".
- **Partial and concurrent execution:** As stated above, TF allows partial execution of the graph. It also supports concurrent executions on overlapping sub-graphs.
- **Distributed execution:** TF provides a simple dataflow-based programming abstraction which simplifies distributed execution as the communication between subcomputations is explicit.

- **Pipelining:** TF is highly parallel and can use various back-end software (GPU, etc.)
- **Major Usage:** TF is a good option for production, mobile platforms and for large-scale distributed model training.
- TF has a good community support and comprehensive **documentation**. **Tensorboard** is a surplus to visualize a graph.

2.2 Shortcomings:[3]

- **Benchmark tests:** TensorFlow lacks in both speed and usage as compared to other systems.
- **GPU and languages support :** Only NVIDIA GPUs are supported. Moreover, Python is the only full language supported.
- **No support for Windows:** No explicit support for windows but TF can be installed within conda environment.
- TF has a unique structure which makes it comparitively hard to find error and thus difficult to debug.

3 Benefits and shortcomings of PyTorch

PyTorch is an open-source machine learning library based on Torch library which is primarily developed by Facebook's AI Research lab (FAIR). It is majorly used for applications such as computer vision and natural language processing.

3.1 Benefits:

- **Pythonic in nature:** PyTorch is easily integratable with Python and it's libraries like NumPy. It also integratable with python's plotting, debugging and data processing tools.
- **Easy to learn:** As PyTorch is pythonic in nature, it's syntax and application are similar to Python. Thus, it increases developer's productivity.
- **Researcher's Friend:** With extremely easy syntax, it makes writing models, data loaders and optimizers comparitively easy and productive.
- **Debugging:** Python's pdb and ipdb tools can be used for debugging PyTorch code.
- **Dynamic Computational Graph Support:** This allows PyTorch to change network behaviour at runtime. Allowing to check each computation providing node-level knowledge of the graph.

3.2 Shortcomings:

- **Lacks Monitoring:** Unlike TF, PyTorch do not have TensorBoard like tool for visualization and monitoring purposes. It requires third party tools for visualizations.
- **Lacking features:** Features like flipping a tensor along a dimension, checking a tensor for NaN and infinity and fast fourier transforms are missing in PyTorch.

4 Common Features:

- **Custom Extensions:** C, C++ or CUDA written custom extensions are easily bind-able in both the frameworks.
- **Library Management**: Both the systems are maintained by big giants, Google and Facebook, thus have advantage of seamless performance and quick updates.
- **Documentation:** Proper documentation and training tutorials are easily available.
- **Support for Python:** Both systems provide support to most widely used ML programming language Python and does not support other deep learning growing languages like Lau. Though Torch, which is parent of PyTorch is written in Lau.

5 Differences:[3]

- PyTorch supports **dynamic computational** graph whereas TensorFlow supports **static computational** graph.
- As stated in benefits of PyTorch, it can use inbuilt python **debugging** tools such as pdb, ipdb, PyCharm or even the print statements whereas TensorFlow requires knowledge of tfdbg.
- One of the key advantages of PyTorch over TensorFlow is it's feature of **declarative data parallelism**.
- TensorFlow has a big advantage in terms of **visualization**, making visual debugging much easier in comparison to PyTorch.
- **Deployment** of models on a specialized servers is possible with TensorFlow.
- TensorFlow has advantage in **serialization**. The entire graph can be saved as a protocol buffer which can then be loaded to other supported languages like C++, Java.
- **Device management** in TensorFlow assumes to run on GPU if it is available whereas PyTorch requires explicit transferring onto the device.

6 Learning from the two research papers:

- I have previously worked with TensorFlow but was completely unaware of PyTorch's structure and features. Thus, this assignment gave me a good enough knowledge of PyTorch and how it is different from TensorFlow.
- Although I was aware of TensorFlow but the in-sites I got by the research paper made my understanding more strong.
- Got to know how TensorFlow is improvement over DistBelief.
- This assignment has motivated me to read the rest of the research papers provided by the Professor.

7 Acknowledgments

I, as the author of this report, would like to thank Professor Jia Zou for this informative and interesting assignment. Looking forward to the next assignments.

References

[1] Abadi, Martín, et al. 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16). 2016

[2] Paszke, Adam, et al. Advances in Neural Information Processing Systems. 2019.

[3] Online support: Site Link