Interactive Distributed Deep Learning with Jupyter Notebooks

Extended Abstract

STEVEN FARRELL, MATTHEW HENDERSON, OLIVER EVANS, SHREYAS CHOLIA, FERNANDO PEREZ, WAHID BHIMJI, SHANE CANON, ROLLIN THOMAS, PRABHAT, Lawrence Berkeley National Laboratory

AARON VOSE, Cray Inc.

Deep learning researchers are increasingly using Jupyter notebooks to implement interactive, reproducible workflows. Such solutions are typically deployed on small-scale (e.g. single server) computing systems. However, as the sizes and complexities of datasets and associated neural network models increase, distributed systems become important for training and evaluating models in a feasible amount of time. In this poster we describe our work on Jupyter notebook solutions for distributed training and hyper-parameter optimization of deep neural networks on high-performance computing systems.

ACM Reference Format:

Steven Farrell, Matthew Henderson, Oliver Evans, Shreyas Cholia, Fernando Perez, Wahid Bhimji, Shane Canon, Rollin Thomas, Prabhat and Aaron Vose. 2018. Interactive Distributed Deep Learning with Jupyter Notebooks: Extended Abstract. 1, 1 (October 2018), 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Deep learning (DL) [LeCun et al. 2015], the sub-field of machine learning which uses multi-layer neural networks (NNs) to solve complex tasks with data, has gained a great deal of popularity in recent years in part due to the availability of large datasets and increasingly powerful computing resources. Meanwhile, Jupyter [Kluyver et al. 2016] notebooks enable code, graphical results, and rich documentation to be combined into an interactive computational narrative, and have become the de facto standard for data science collaboration, development and pedagogy. Notebook-based DL workflows are widely used but typically deployed on small-scale computing systems, but as the sizes and complexities of datasets and associated neural network models increase, distributed computing systems become essential for generating the best results in a reasonable time. In particular, models that are slow to train can benefit from distributed data-parallel training, and model-selection tasks can be accelerated with distributed hyper-parameter optimization (HPO). In this poster we demonstrate several distributed notebook-based deep learning workflows on HPC systems, interfacing to the Cori supercomputer at NERSC. We describe our architecture to effectively use these HPC resources via Jupyter, as shown in Figure 1, taking into account the hardware and policy restrictions that are common with other large HPC machines. We then present our notebook-based approaches, including scaling results for distributed training showing no overhead from the notebook infrastructure (Figure 2). We take advantage of and build on features available in the Jupyter and Python ecosystem including JupyterHub, IPyWidgets, IPython "magic" commands, Dask, and IPyParallel

Authors' addresses: Steven Farrell, Matthew Henderson, Oliver Evans, Shreyas Cholia, Fernando Perez, Wahid Bhimji, Shane Canon, Rollin Thomas, Prabhat, Lawrence Berkeley National Laboratory, Berkeley, CA, wbhimji@lbl.gov; Aaron Vose, Cray Inc. Seattle, WA, avose@cray.com.

Manuscript submitted to ACM

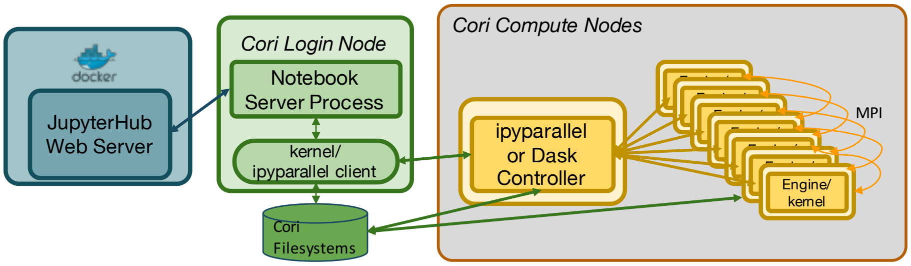
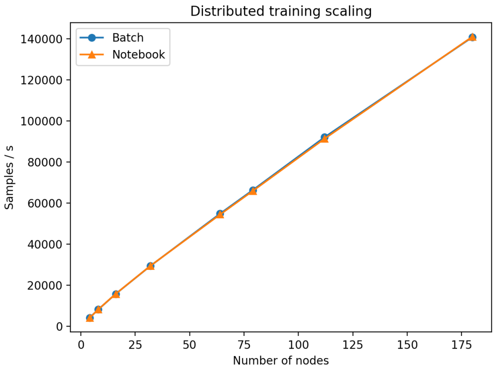
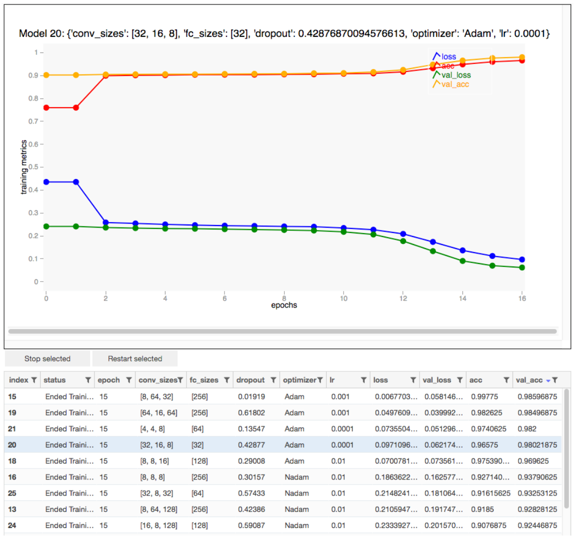


Fig. 1. Diagram of the distributed workload system




Model 20: {'conv_sizes': [32, 16, 8], 'fc_sizes': [32], 'dropout': 0.42876870094576613, 'optimizer': 'Adam', 'lr': 0.0001}

index	status	epoch	conv_sizes	fc_sizes	dropout	optimizer	lr	loss	val_loss	acc	val_acc
15	Ended Traini...	15	[8, 64, 32]	[256]	0.01919	Adam	0.001	0.0067703...	0.058146...	0.99775	0.98596875
19	Ended Traini...	15	[64, 16, 64]	[256]	0.61802	Adam	0.001	0.0497609...	0.039992...	0.982625	0.98496875
21	Ended Traini...	15	[4, 4, 8]	[64]	0.13547	Adam	0.0001	0.0735504...	0.051296...	0.9740625	0.982
20	Ended Traini...	15	[32, 16, 8]	[32]	0.42877	Adam	0.0001	0.0971096...	0.062174...	0.96575	0.98021875
18	Ended Traini...	15	[8, 8, 16]	[128]	0.29008	Adam	0.01	0.0700781...	0.073561...	0.975390...	0.969625
16	Ended Traini...	15	[8, 8, 8]	[256]	0.30157	Nadam	0.01	0.1863622...	0.162577...	0.927140...	0.93790625
25	Ended Traini...	15	[32, 8, 32]	[64]	0.57433	Nadam	0.01	0.2148241...	0.181064...	0.91615625	0.93253125
13	Ended Traini...	15	[8, 64, 128]	[256]	0.42386	Nadam	0.01	0.2105947...	0.191747...	0.9185	0.92828125
24	Ended Traini...	15	[16, 8, 128]	[128]	0.59087	Nadam	0.01	0.2333927...	0.201570...	0.9076875	0.92446875

Fig. 2. Scaling of distributed training with and without the notebook infrastructure (left). Screenshot of interactive Jupyter widget for HPO (right).

which provide us with tools to interact with the backend tasks directly through a visual interface (Figure 2). As well as human interaction we couple this infrastructure with a powerful automated hyper-parameter framework incorporating genetic algorithms. We demonstrate these methods with a science use-case and dataset from High Energy Physics which applies convolutional neural networks (CNNs) to classify images formed from Large Hadron Collider events, and is described in detail in [Bhimji et al. 2017]. We find that we can improve on state-of-the-art results by using distributed HPC resources to improve model parameters. Example notebooks and recipes for running these at NERSC are provided.

REFERENCES

Wahid Bhimji, Steven Andrew Farrell, Thorsten Kurth, Michela Paganini, Evan Racah, and Prabhat. 2017. Deep Neural Networks for Physics Analysis on low-level whole-detector data at the LHC. *arXiv preprint arXiv:1711.03573* (2017).

Thomas Kluyver, Benjamin Ragan-Kelley, Fernando Pérez, Brian Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, Jessica Hamrick, Jason Grout, Sylvain Corlay, Paul Ivanov, Damián Avila, Safia Abdalla, and Carol Willing. 2016. Jupyter Notebooks – a publishing format for reproducible computational workflows. In *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, F. Loizides and B. Schmidt (Eds.). IOS Press, 87 – 90.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436.