**NBD(NIST Big Data) Requirements WG Use Case Template Aug 11 2013**

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case Title** | | Large-scale Deep Learning | |
| **Vertical (area)** | | Machine Learning/AI | |
| **Author/Company/Email** | | Adam Coates / Stanford University / [acoates@cs.stanford.edu](mailto:acoates@cs.stanford.edu) | |
| **Actors/Stakeholders and their roles and responsibilities** | | Machine learning researchers and practitioners faced with large quantities of data and complex prediction tasks. Supports state-of-the-art development in computer vision as in automatic car driving, speech recognition, and natural language processing in both academic and industry systems. | |
| **Goals** | | Increase the size of datasets and models that can be tackled with deep learning algorithms. Large models (e.g., neural networks with more neurons and connections) combined with large datasets are increasingly the top performers in benchmark tasks for vision, speech, and NLP. | |
| **Use Case Description** | | A research scientist or machine learning practitioner wants to train a deep neural network from a large (>>1TB) corpus of data (typically imagery, video, audio, or text). Such training procedures often require customization of the neural network architecture, learning criteria, and dataset pre-processing. In addition to the computational expense demanded by the learning algorithms, the need for rapid prototyping and ease of development is extremely high. | |
| **Current**  **Solutions** | **Compute(System)** | | GPU cluster with high-speed interconnects (e.g., Infiniband, 40gE) |
| **Storage** | | 100TB Lustre filesystem |
| **Networking** | | Infiniband within HPC cluster; 1G ethernet to outside infrastructure (e.g., Web, Lustre). |
| **Software** | | In-house GPU kernels and MPI-based communication developed by Stanford CS. C++/Python source. |
| **Big Data  Characteristics** | **Data Source (distributed/centralized)** | | Centralized filesystem with a single large training dataset. Dataset may be updated with new training examples as they become available. |
| **Volume (size)** | | Current datasets typically 1 to 10 TB. With increases in computation that enable much larger models, datasets of 100TB or more may be necessary in order to exploit the representational power of the larger models. Training a self-driving car could take 100 million images. |
| **Velocity**  **(e.g. real time)** | | Much faster than real-time processing is required. Current computer vision applications involve processing hundreds of image frames per second in order to ensure reasonable training times. For demanding applications (e.g., autonomous driving) we envision the need to process many thousand high-resolution (6 megapixels or more) images per second. |
| **Variety**  **(multiple datasets, mashup)** | | Individual applications may involve a wide variety of data. Current research involves neural networks that actively learn from heterogeneous tasks (e.g., learning to perform tagging, chunking and parsing for text, or learning to read lips from combinations of video and audio). |
| **Variability (rate of change)** | | Low variability. Most data is streamed in at a consistent pace from a shared source. Due to high computational requirements, server loads can introduce burstiness into data transfers. |
| **Big Data Science (collection, curation,**  **analysis,**  **action)** | **Veracity (Robustness Issues, semantics)** | | Datasets for ML applications are often hand-labeled and verified. Extremely large datasets involve crowd-sourced labeling and invite ambiguous situations where a label is not clear. Automated labeling systems still require human sanity-checks. Clever techniques for large dataset construction is an active area of research. |
| **Visualization** | | Visualization of learned networks is an open area of research, though partly as a debugging technique. Some visual applications involve visualization predictions on test imagery. |
| **Data Quality (syntax)** | | Some collected data (e.g., compressed video or audio) may involve unknown formats, codecs, or may be corrupted. Automatic filtering of original source data removes these. |
| **Data Types** | | Images, video, audio, text. (In practice: almost anything.) |
| **Data Analytics** | | Small degree of batch statistical pre-processing; all other data analysis is performed by the learning algorithm itself. |
| **Big Data Specific Challenges (Gaps)** | | Processing requirements for even modest quantities of data are extreme. Though the trained representations can make use of many terabytes of data, the primary challenge is in processing all of the data during training. Current state-of-the-art deep learning systems are capable of using neural networks with more than 10 billion free parameters (akin to synapses in the brain), and necessitate trillions of floating point operations per training example. Distributing these computations over high-performance infrastructure is a major challenge for which we currently use a largely custom software system. | |
| **Big Data Specific Challenges in Mobility** | | After training of large neural networks is completed, the learned network may be copied to other devices with dramatically lower computational capabilities for use in making predictions in real time. (E.g., in autonomous driving, the training procedure is performed using a HPC cluster with 64 GPUs. The result of training, however, is a neural network that encodes the necessary knowledge for making decisions about steering and obstacle avoidance. This network can be copied to embedded hardware in vehicles or sensors.) | |
| **Security & Privacy**  **Requirements** | | None. | |
| **Highlight issues for generalizing this use case (e.g. for ref. architecture)** | | Deep Learning shares many characteristics with the broader field of machine learning. The paramount requirements are high computational throughput for mostly dense linear algebra operations, and extremely high productivity. Most deep learning systems require a substantial degree of tuning on the target application for best performance and thus necessitate a large number of experiments with designer intervention in between. As a result, minimizing the turn-around time of experiments and accelerating development is crucial.  These two requirements (high throughput and high productivity) are dramatically in contention. HPC systems are available to accelerate experiments, but current HPC software infrastructure is difficult to use which lengthens development and debugging time and, in many cases, makes otherwise computationally tractable applications infeasible.  The major components needed for these applications (which are currently in-house custom software) involve dense linear algebra on distributed-memory HPC systems. While libraries for single-machine or single-GPU computation are available (e.g., BLAS, CuBLAS, MAGMA, etc.), distributed computation of dense BLAS-like or LAPACK-like operations on GPUs remains poorly developed. Existing solutions (e.g., ScaLapack for CPUs) are not well-integrated with higher level languages and require low-level programming which lengthens experiment and development time. | |
| **More Information (URLs)** | | Recent popular press coverage of deep learning technology:  <http://www.nytimes.com/2012/11/24/science/scientists-see-advances-in-deep-learning-a-part-of-artificial-intelligence.html>  <http://www.nytimes.com/2012/06/26/technology/in-a-big-network-of-computers-evidence-of-machine-learning.html>  <http://www.wired.com/wiredenterprise/2013/06/andrew_ng/>  A recent research paper on HPC for Deep Learning: <http://www.stanford.edu/~acoates/papers/CoatesHuvalWangWuNgCatanzaro_icml2013.pdf>  Widely-used tutorials and references for Deep Learning:  <http://ufldl.stanford.edu/wiki/index.php/Main_Page>  <http://deeplearning.net/> | |
| **Note:** <additional comments> | | | |