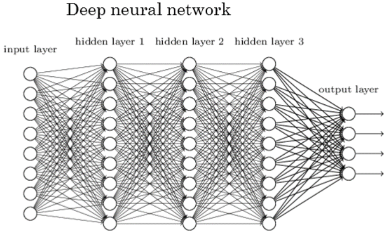
Distributed Stochastic Gradient Descent

Kevin Yang and Michael Farrell

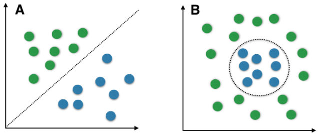
April 27, 2016

Motivation - Deep Learning

- Deep-Learning
  - Objective: Learn a complicated, non-linear function that minimzes some loss function
- Why do we need deep models?
  - The class of linear functions is inadequate for many problems.

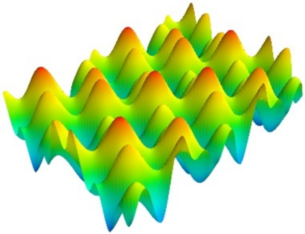


http://www.rsipvision.com/exploring-deep-learning/



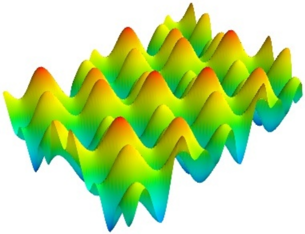
http://sebastianraschka.com/Articles/2014_naive_bayes_1.html

Motivation - Deep Learning



- How do we learn these deep models?
  - Choose a random example
  - Run the neural network on the example
  - Adjust the parameters of the network such that our loss function is minimized more than it was before
  - Repeat
- Difficulties?
  - Local Minima
  - Non-convexity
  - Neural Networks can have millions or even billions of parameters

Motivation - Deep Learning



- How do we learn these deep models?
  - Choose a random example
  - Run the neural network on the example
  - Adjust the parameters of the network such that our loss function is minimized more than it was before
  - Repeat
- Difficulties?
  - Local Minima
  - Non-convexity
  - Neural Networks can have millions or even billions of parameters

Motivation - SGD

- How do we maximize our reward function?
  - One common technique is Stochastic Gradient Descent
  - $\mathbf{w}$ is the vector of parameters for the model
  - η is the learning rate
  - $\mathbf{f}(\mathbf{w})$ is the loss function evaluated with the current parameters $\mathbf{w}$
  - $\mathbf{w} \leftarrow \mathbf{0}$

    while $\mathbf{f}(\mathbf{w})$ is not minimized **do**

        for $i = 1, n$ **do**

            $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla f(\mathbf{w})$
  - As the number of training examples, n, and the number of parameters, $|\mathbf{w}|$, increases, this algorithm quickly becomes very slow...

Motivation - Distributed SGD

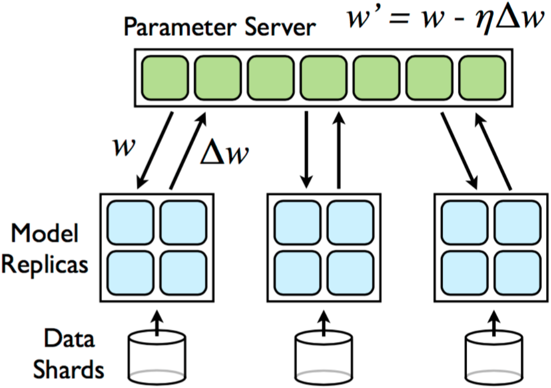
- Since some of these models take days/weeks/months to run, we would hope that we could use a distributed computing cluster parallelize this process.
- Learn from Google!
  - DistBelief- 2012
    - Downpour SGD
    - Sandblaster L-BFGS
  - TensorFlow -2015
    - gRPC

Motivation - Distributed SGD

- Since some of these models take days/weeks/months to run, we would hope that we could use a distributed computing cluster parallelize this process.
- Learn from Google!
  - DistBelief- 2012
    - Downpour SGD
    - Sandblaster L-BFGS
  - TensorFlow -2015
    - gRPC

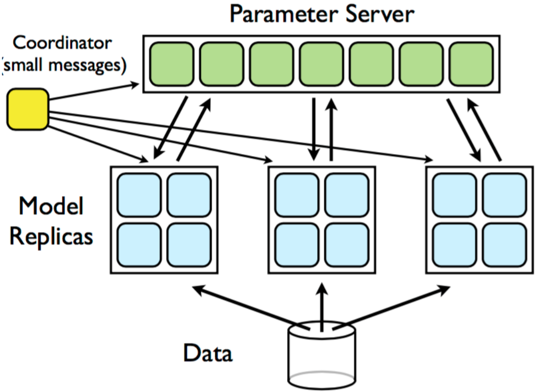
DistBelief - Downpour SGD

- "An asynchronous stochastic gradient descent procedure supporting a large number of model replicas."



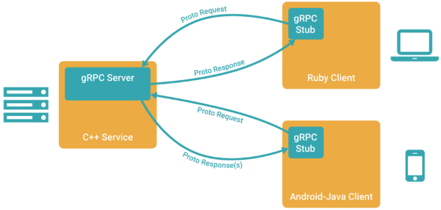
DistBelief - Sandblaster L-BFGS

- "A framework that supports a variety of distributed batch optimization procedures, including a distributed implementation of L-BFGS"



TensorFlow-GRPC

- Second Generation ML Model focused on distributing models to CPUs and GPUs
- Uses the high performance RPC framework (GRPC) to communicated between separate processes
  - Uses Protocol Buffers -v3
  - C-based
  - Client-server stubs in 10+ languages and counting



DistBelief/TensorFlow Summary

- TensorFlow is basically the second version of DistBelief that is approximately twice as fast and much more user-friendly.
- Results from DistBelief:

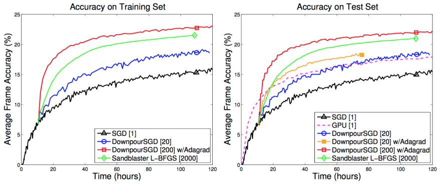


Figure 4: Left: Training accuracy (on a portion of the training set) for different optimization methods. Right: Classification accuracy on the hold out test set as a function of training time. Downpour and Sandblaster experiments initialized using the same ∼10 hour warmstart of simple SGD.

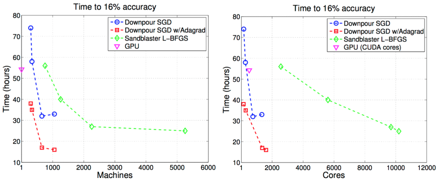


Figure 5: Time to reach a fixed accuracy (16%) for different optimization strategies as a function of number of the machines (left) and cores (right).

Our Project

- We hope to build our own Distributed SGD model modeled based on both DistBelief and TensorFlow
  - From TensorFlow- Use GRPC with Protocol Buffers to communicated between processes
  - From DistBelief -Implement Downpour-SGD (the most effective model with limited resources) first, and then attempt to implement Sandblaster

Our Example

Talk about image dataset

Exploration of Downpour-SGD

- The Downpour-SGD requires the passing of parameters between processes
- Bottleneck here is bandwidth
- Parameters in the range of 100MB-50GB

Large Data Sets

Protocol Buffers are not designed to handle large messages. As a general rule of thumb, if you are dealing in messages larger than a megabyte each, it may be time to consider an alternate strategy.

That said, Protocol Buffers are great for handling individual messages *within* a large data set. Usually, large data sets are really just a collection of small pieces, where each small piece may be a structured piece of data. Even though Protocol Buffers cannot handle the entire set at once, using Protocol Buffers to encode each piece greatly simplifies your problem: now all you need is to handle a set of byte strings rather than a set of structures.

Exploration of Downpour-SGD Algorithm

-