Project 2. Design Space Exploration of Distributed Machine Learning Systems:

The choice of configuration parameters in distributed machine learning and their impact on training time and overall model performance.

Denise S. Davis, MS and Pooyan Jamshidi,PhD

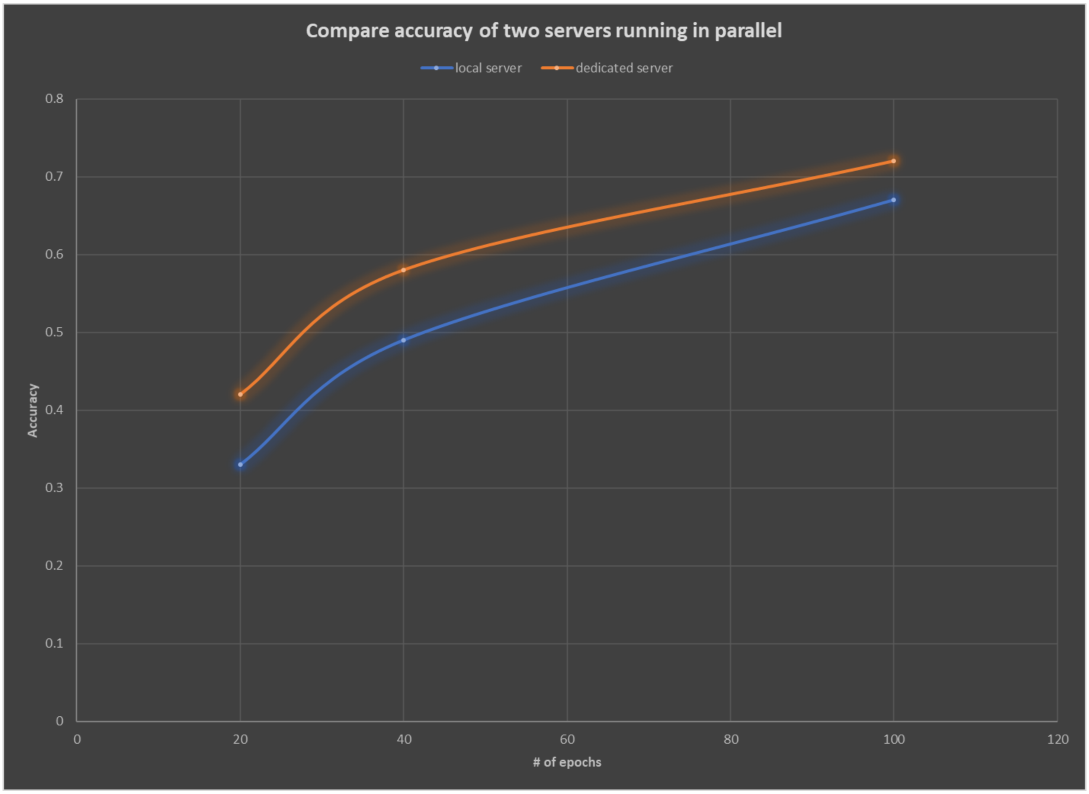
College of Engineering and Computing, Department of Computer Science, University of South Carolina.

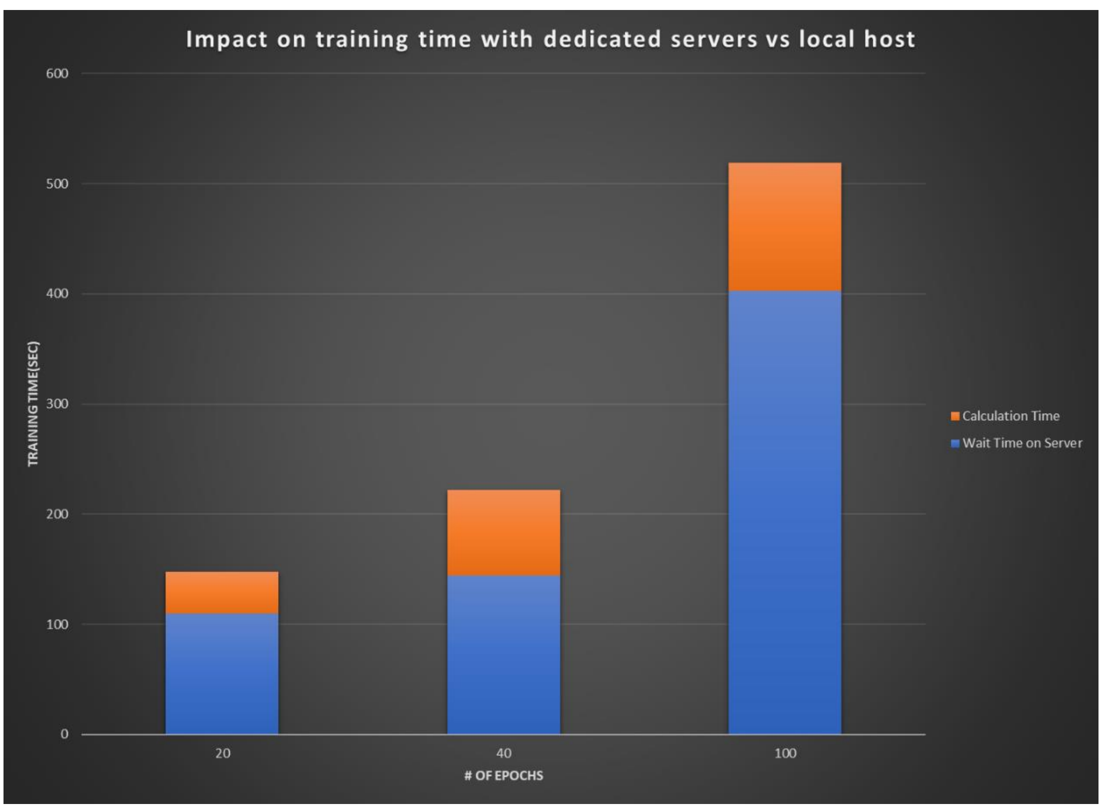
We propose a simplistic approach to evaluate the training time and performance of distributed machine learning systems. Finding the optimal configuration parameters for Distributed Machine Learning, ML, Systems is an exploratory process and unique to each system. In practice, a parameter server, a separate TensorFlow server on the local host or a dedicated server/node, manages and coordinates receiving and updating configuration parameters for training the model. The worker servers, separate TensorFlow servers on local host or dedicated servers/nodes, perform calculations of the model on separate shards of data and communicate with the parameter server with updates to their models. In this study, we investigate the impact of using the local host for parameter servers as well as the worker servers versus dedicated servers/nodes for each. We measure training time and model accuracy under various conditions: update schemas, asynchronous and synchronous, learning rates, batch size and the number of epochs on a fixed training model.

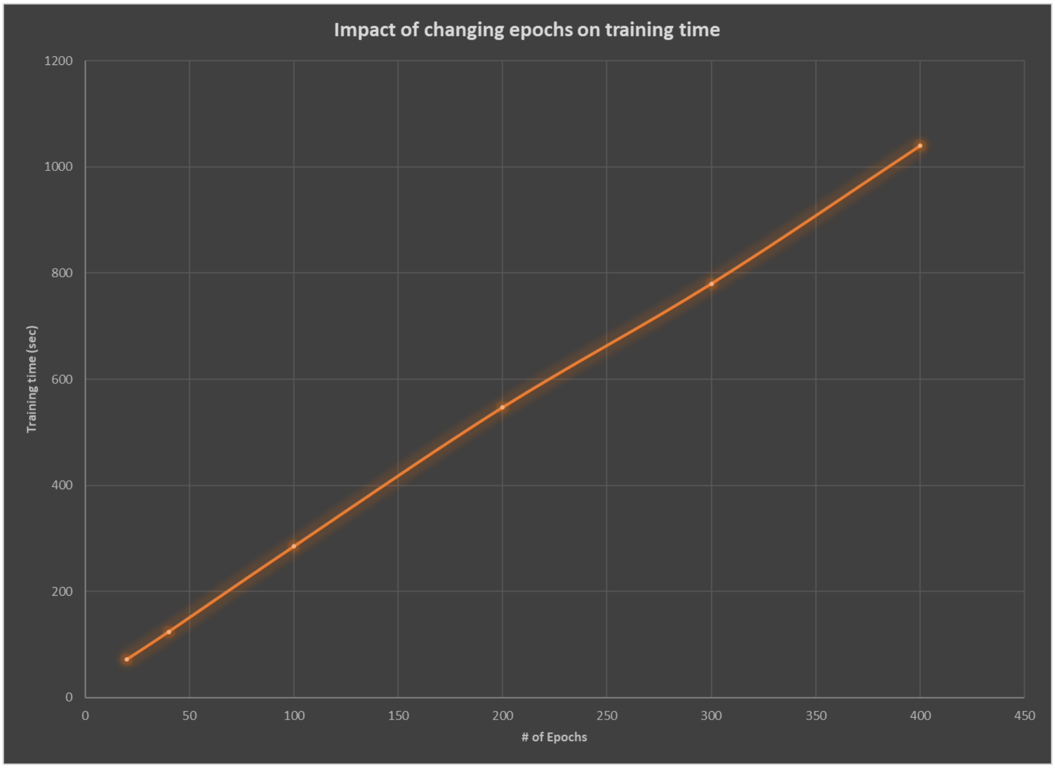
Our study was performed in a research environment with 2 Quadpro M6000 24 Gbyte GPU nodes and two types of data: MNIST, CIFAR10 to train simple neural networks and convolutional neural networks for image classification. (Time permitting, a third set will include TensorFlow CNN benchmark datasets.) Generalizations of configuration parameters will be presented. These results can be extrapolated to scalable real world systems, having been ruled out on the simplest of systems.

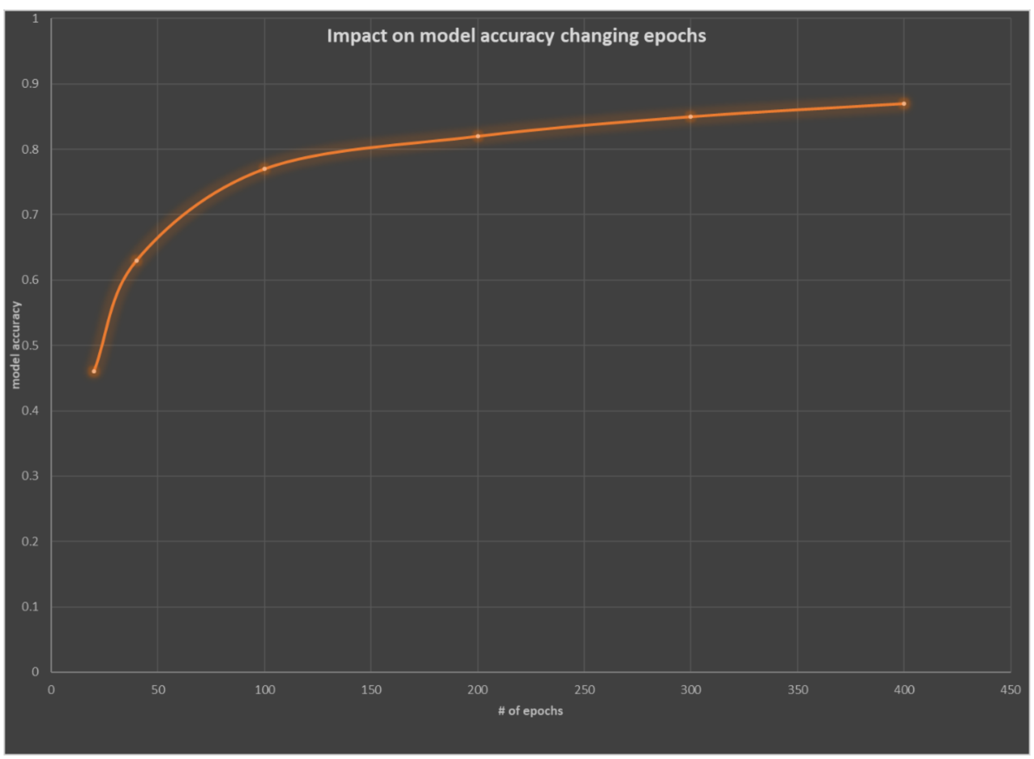
Summary of findings:

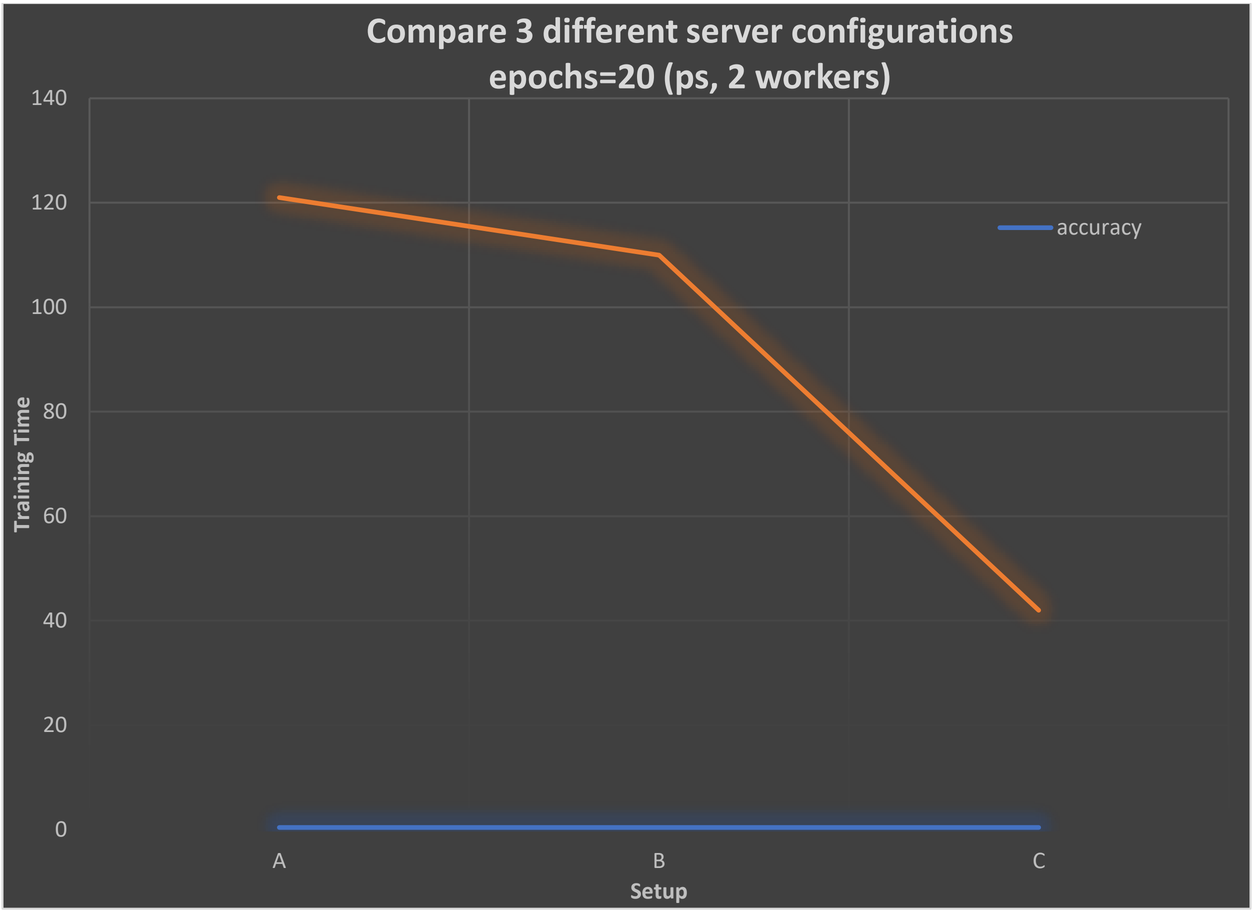
Tensorflow Distributed Machine Learning is not only intended for setup on dedicated servers, the architecture can also be used to encapsulate jobs on the same server and run independently or communicate together and update parameters. Clearly, running on the same server and not exceeding the resources of that server will run more efficiently than having to communicate over the internet/network with multiple servers. We measured a significant increase on training time going through gRPC protocol to communicate parameter updates and receive updates, ranging from two to up to four times the training time on a local server. When designing a Distributed ML systems, the placement of the parameter server and all of the worker servers, impacts training time. From our measurements to date, the first step in designing Distributed ML system is to determine which calculations can be run independently, encapsulated, and shared with a parameter server. This step will improve and save training time. If the model is split onto dedicated servers, separate from the parameter server, training time will be considerably slower compared to that with the parameter server located at least one of the worker servers.


Compare accuracy of two servers running in parallel
local server
dedicated server
Accuracy
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
20
40
60
80
100
120
# of epochs


Impact on training time with dedicated servers vs local host
TRAINING TIME(SEC)
600
500
400
300
200
100
0
20
40
100
OF EPOCHS
Calculation Time
Wait Time on Server


Impact of changing epochs on training time
Training time (sec)
# of Epochs
0
200
400
600
800
1000
1200
0
50
100
150
200
250
300
350
400
450


Impact on model accuracy changing epochs
model accuracy
# of epochs
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
50
100
150
200
250
300
350
400
450


Compare 3 different server configurations
epochs=20 (ps, 2 workers)
Training Time
140
120
100
80
60
40
20
0
accuracy
A
B
C
Setup