**Communication Cost in Distributed Deep Learning: A Comprehensive Survey**

Vinayak Kothari, Anirudh Kaushik, Hasan Ahmed Faisal

**Problem Foundation:**

Deep learning (DL) has attained a lot of advances in recent years and it is being used by the world's leading researchers in Artificial Intelligence, Computer Vision , NLP. The explosion of data and an increase in model size has led to great interest in training deep neural networks on distributed systems. In particular, the data parallel approach is a widely used technique in the field of distributed computing.

The way data parallel paradigm works is by decomposing gradient computations - which are often time consuming - into sub-tasks, and assigning them to separate worker machines for execution. The data is distributed among multiple workers. Each worker also gets its own individual copy of the model parameters. For every iteration, every worker computes the local gradient for the mini-batch generated from the local copy of it’s training data. These M local gradients, thus computed are then synchronously aggregated to form the global stochastic gradient, which is then used to update the weights and other model parameters like the biases.

While the data parallelism approach lets us to speed up the convergence of the model by distributing the training among the workers and utilizing each worker’s compute power, but it suffers from two major drawbacks which can slow down the convergence -

* **Communication Cost** - With increase in the size and complexity of deep learning models, the workers can send gradients in the order of a million per each iteration. Thus, communication overhead has emerged as one of the biggest performance bottlenecks in the data parallel approach to distributed deep learning.
* **Synchronization Cost** - For every iteration, the master node has to wait for every worker to send its gradient. Due to improper load balancing among the workers or difference in their computing powers, some workers may be slow as compared to their peers. This will lead to the master node waiting idly for the slow workers to send their gradients.

In our project we would like to focus on the first overhead - **Communication Cost**. We would like to compare the recent techniques which have been proposed in order to tackle this problem.

**Traditional Solutions to reduce Communication Overhead :**

There are two widely accepted methods which aim to reduce the communication overhead -

* **Quantization** - In this method the gradients are computed with fewer numbers of bits. This means the workers send less data to the master per iteration which reduces the load on the network bandwidth. There have been various algorithms for quantization. For example, (Seide et al, 2014) quantized gradients to a representation of {−1, 1}, and empirically showed the communication-efficiency in training of deep neural networks. While, (Bernstein et al, 2018, Bernstein et al, 2019) considered the bi-direction communications of gradients between master and workers.
* **Sparsification** - This method requires dropping of some coordinates of gradients according to some rule. For example, (Jiang et al, 2018) proposed to transmit only a subset of model parameters in each iteration. They showed a convergence rate of O(1/ √ K) and a linear speedup as long as all of the model parameters were transmitted in limited consecutive iterations.

Naturally, these optimizations to reduce the communication overhead do not come without a price. The tradeoff here is that because of loss of precision in Quantization algorithms or dropping of some coordinates of gradients in Sparsification algorithms, the convergence rate of the model suffers. Therefore, we cannot afford too much loss of precision or dropping out of gradients.

We would like to further explore improving these techniques by targeting their disadvantages by combining Terngrad with Gossip architecture which will further reduce communication cost and speed up the training process.

**Proposed Solution:**

We would like to present a case study on efficient communication techniques implemented by researchers between multiple clusters to accelerate the training process. We believe that our survey will further help people in choosing methods to reduce communication cost between different clusters.

We attempt to analyze recent techniques which build upon the quantization and sparsification techniques of communication cost reduction without compromising too much on the convergence rate of the model.

To analyse communication cost between different architectures we have selected the following architectures - Vanilla Parameter Server architecture, TernGrad -that quantizes gradients to ternary levels {−1, 0, 1}, Edge stochastic gradient descent and Lazily aggregated Quantized Gradients first on small models such as LeNet proposed by Yann LeCun and then on large scale distributed models such as VGGNet 16 and AlexNet.

**Proposed Evaluation:**

We have selected the Tensorflow framework to benchmark our experiments on various cost reduction techniques. Since Tensorflow provides the vanilla parameter server as well as the ring all reduce technique which reduces the communication overhead by eliminating the server and aggregating the gradients at the workers site itself, it should be fairly straightforward to benchmark those two.

For the first part of our experiment, we would benchmark the vanilla parameter server and the ring all reduce algorithm. Our evaluation strategy would be first to find training time and accuracy of LeNet, AlexNet and VGGNet 16 on subset on Image net. The key parameters we would like to check are model convergence time and communication cost on the same dataset. We will calculate it by analyzing the amount of data transferred from one worker to server or vice versa, time taken to transfer , unpacking of data , loading of data for every iteration and summing up all the costs will give us the cost of the model.

Next we will attempt to implement terngrad Quantization, Edge stochastic gradient descent algorithm as well as Lazily aggregated Quantized gradient techniques using the Tensorflow framework. Again benchmarking for convergence time and cost of communication for distributed simple model LeNet and then large scale distributed models AlexNet and VGGNet16 on the same dataset as used above.

**Literature survey:**

Optimizing communication cost is a well known problem in distributed deep learning and some state of the art models include Gradient Sparsification which was proposed by Aji et al, 2017 **[5]**. In this algorithm they truncated the smallest gradients and transmitted remaining large gradients. Communication overhead was greatly reduced and 22% speed gain was achieved across 4 GPUs for neural machine translation without translation accuracy. Gradient Quantization method was proposed by DoReFa-Net **[6]** derived from AlexNet which reduced the bit length of weights, activations and gradients to 1 ,2 6 respectively. It led to a 9.8% decrease in accuracy of model prediction.

Initial distributed computing architecture where no optimization was applied were Parameter server **[7]** architectures in which there is one master node called server and many workers. Server contains the newest model and collects updated information from the worker. Here every node has connection with the server , so it is a centralized framework and widely used for distributed training. Here one main challenge is congestion at the server which can be a bottleneck in communication.

To avoid communication bottleneck at the server many researchers used All- Reduce architecture**[8]** to implement gradient aggregation without a central server. Here all workers communicate without a central node and with successful communication they acquire gradients of all workers and update their model which gives the same model for every worker. Here a successful model is attained by every worker after synchronization without a server so it is called a decentralized system.

Next we looked at Gossip **[9]** architecture which is like All Reduce architecture without a server node. But workers do not need to communicate with all workers except neighbours. Worker exchange parameters via those workers choose in a given iteration. After one communication during training, the algorithm does not guarantee the consistency of parameters across all workers but does guarantee it at the end of the algorithm.

Next we looked at a few unconventional algorithms which were very interesting to study about and implemented in our project.

1. **TernGrad:** This paper uses ternary gradients to accelerate distributed deep learning in data parallelism. Our approach requires only three numerical levels {−1, 0, 1}, which can aggressively reduce the communication time **[10]**.
2. **eSGD:** Edge Stochastic Gradient descent includes two optimization levels to improve communication cost which sends only important gradients to cloud for synchronization and second is momentum residual accumulation is designed for tracking out-of-date residual gradient coordinates to avoid low convergence rate caused by sparse updates. In paper it is shown that 91.2%, 86.7%, 81.5% accuracy on MNIST data set with gradient drop ratio 50%, 75%, 87.5% respectively **[11]**.
3. **Lazily aggregated Quantized:** This paper proposes novel aggregation of gradients which compressed the gradients communication. Here gradients are quantized first and skip less informative gradients by reusing outdated gradients **[12]**.

**References:**

**[1]** F. Seide, H. Fu, J. Droppo, G. Li, and D. Yu. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns. In Fifteenth Annual Conference of the International Speech Communication Association, 2014.

**[2]** J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar. signSGD: Compressed Optimisation for Non-Convex Problems. In International Conference on Machine Learning (ICML), 2018.

**[3]** J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar. signSGD with Majority Vote is Communication Efficient and Fault Tolerant. International Conference on Learning Representations (ICLR), 2019.

**[4]** P. Jiang and G. Agrawal. A linear speedup analysis of distributed deep learning with sparse and quantized communication. In Advances in Neural Information Processing Systems, pages 2525–2536. Curran Associates, Inc., 2018.

**[5]** Alham Fikri Aji and Kenneth Heafield. Sparse communication for distributed gradient descent. arXiv preprint:1704.05021, 2017.

**[6]** Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. arXiv preprint arXiv:1606.06160, 2016.  
**[7]** A. Smola and S. Narayanamurthy, “An architecture for parallel topic models,” Proc. VLDB Endow., vol. 3, no. 1-2, pp. 703–710, Sep. 2010.

**[8]** G. Lan, S. Lee, and Y. Zhou, “Communication-efficient algorithms for decentralized and stochastic optimization,” CoRR, vol. abs/1701.03961, 2017.

**[9]** A. Nedic and A. Ozdaglar, “Distributed subgradient methods for multiagent optimization,” IEEE Transactions on Automatic Control, vol. 54, no. 1, pp. 48–61, Jan 2009.

**[10]** Wei Wen1, Cong Xu, “TernGrad: Ternary Gradients to Reduce Communication in Distributed Deep Learning,” 31st Conference on Neural Information Processing Systems (NIPS 2017)

**[11]** Zeyi Tao and Qun Li, “Communication Efficient Distributed Deep Learning on the Edge”, Hotedge 2018

**[12]** Jun Sun, TianyiChen, “Communication-Efficient Distributed Learning via Lazily Aggregated Quantized Gradients” , 33rd Conference on Neural Information Processing Systems (NeurIPS 2019)