

---

**2ND WORKSHOP  
“MACHINE LEARNING &  
NETWORKING” (MaLeNe)  
PROCEEDINGS**


---

**SEPTEMBER 4,  
2023**



**CO-LOCATED WITH  
THE 5TH INTERNATIONAL CONFERENCE ON  
NETWORKED SYSTEMS (NETSYS 2023)  
POTSDAM, GERMANY**

# Data distribution effects on asynchronous Parameter Server training in high delay differences networks

1<sup>st</sup> Leonard Paeleke   
Hasso Plattner Institute (HPI)  
University of Potsdam  
Potsdam, Germany  
leonard.paeleke@hpi.de

2<sup>nd</sup> Holger Karl   
Hasso Plattner Institute (HPI)  
University of Potsdam  
Potsdam, Germany  
holger.kar@hpi.del

**Abstract**—Model training with distributed Machine Learning (ML) methods, such as the well-known asynchronous Parameter Server (PS), is susceptible to network inhomogeneities, e.g. differences in link latency, computing resources, or data distribution. We evaluate how the combination of data distribution and high delay difference between workers affects asynchronous PS training. We show that a better ground-truth representation at the faster and more dominating worker increases model accuracy and reduces computational efforts by about five times.

**Index Terms**—Distributed Machine Learning, Asynchronous Parameter Server, Networks for ML

## I. INTRODUCTION

Networked systems, such as IoT networks and future cellular communication networks, are envisioned to utilize Machine Learning (ML) for complex task. Training these ML models requires data that usually is collected on networked machines. However, data transmission to, e.g., a data center can heavily load the network and violate data privacy concerns. With *distributed* ML methods, e.g., using a Parameter Server (PS) [1], [2], the training can be distributed across multiple decentralized machines (*workers*). These workers maintain an instance of the model and process their local data to compute update gradients that describe necessary changes to the model parameters (bias, weights). Typically, the data set is distributed across all workers in non-overlapping portions. After processing a predefined number of data samples, the workers exchange their update gradients with a centralized machine (*parameter server*), which applies the update gradients of all workers to the model. The updated model is then returned to the contributing workers, which then continue generating update gradients. The parameter server also stores the model parameters.

Various PS implementations exist that primarily differ in how and how many workers contribute to a model update. In synchronous implementations, the parameter server aggregates the update gradients from all workers simultaneously before broadcasting, while the workers interrupt their processing until receiving the updated model. In contrast, in asynchronous PS implementations, the parameter server updates and returns the updated model immediately after receiving an update gradient, reducing the idle time of workers [3].

In practice, PS is exposed to varying link latencies or computing capacities. Such network inhomogeneities lead to long training times for synchronous PS and motivated asynchronous PS [3]. For asynchronous PS, network inhomogeneities change the order of model updates, which impacts the model development and makes the model development susceptible to data distribution [4]. Several methods have been described to mitigate these effects, e.g., back-up workers [5], limiting worker staleness [6], or changes to the typically used stochastic gradient descent (SGD) optimizer [7], [8]. Still, it is unclear how data should be distributed when the delays between worker and parameter server differ significantly. We help understand how the combination of inhomogeneous networks and data distribution affects model development for asynchronous PS.

## II. EXPERIMENT DESIGN

In our experiment, an asynchronous PS in an inhomogeneous network trains an artificial neural network on the NSL-KDD data set [9] for different data distributions. The training infrastructure comprises three virtual machines (Ubuntu 22.04) on the same physical server with equal computing resources, setting up a network with two workers, each connected to a parameter server. We use Ray [10] to implement the asynchronous PS. To model an inhomogeneous network that causes changes in update order due to varying link latency or computation capacity, we impose delays of 1 ms and 100 ms, respectively, on the links from worker to parameter server using the `tc` package. This gives us fine-grained control over the transmission time from sending an update to receiving the updated model.

NSL-KDD is typically used for training models detecting intrusions in traffic data. Under supervised learning, a model learns to classify traffic into five classes: Denial of Service (DoS), User to Root (U2R), Remote to Local (R2L), Probing, and Normal. Here, we train the model on three quarters of the data set and evaluate the model's accuracy on the remainder. As the disproportionate impact of data distribution on model development in inhomogeneous settings is our key hypothesis, we distribute the data differently among the workers. For the experiments, the *fast* worker (with lower added link delay) accesses the DoS, R2L, and one-half of the Normal traffic

data. The *slow* worker with a higher link delay accesses the other half of the Normal traffic data. These distributions stay fixed for all experiments; the distribution of U2R and Probing data varies across experiments.

The model to train is a fully connected artificial neural network with one hidden layer. It has 93 neurons in the input layer, 21 neurons in the hidden layer, and five neurons, one for every class as a one-hot-encoding, in the output layer. The hidden layer uses ReLU activation; the output layer uses logarithmic softmax activation.

### III. DATA DISTRIBUTION EFFECTS ON MODEL QUALITY

In this section, we analyze the effects of data distribution on model development for asynchronous PS training in an inhomogeneous network. A uniform delay of 1 ms for the fast link and 100 ms for the slower link is used to form the inhomogeneous network. Throughout the experiment, we vary the proportions of U2R and Probing data accessed by each worker in increments of ten from zero to 100 percent. The data portions across both workers always add up to 100 percent. For each data distribution, we train twenty times with reshuffled data sets and with different model initializations. Model accuracy is the key metric; we show it over both number of training epochs as well as over data proportions.

Figure 1 shows that model accuracy increases the more the dominant (faster) worker knows about the ground-truth distribution. Hence, the reference independent identically distributed (IID) case achieves the highest accuracy, where data is equally distributed among the workers, such that each worker’s data portion represents the entire ground-truth distribution. This is to be expected in such a vastly heterogeneous setup. However, in real-world applications with constrained computing capacity, (quickly) reaching specific model quality levels is of higher interest than the maximal accuracy. Given, e.g., the goal of 94% accuracy, even a little information about the classes at the slower worker reduces the necessary epochs to a third (cf. 0/100, 20/80 at 94% accuracy). The number of epochs can be reduced up to five times if the faster worker has more information about these classes.

### IV. SUMMARY AND OUTLOOK

We have shown that for the well-known asynchronous PS method in inhomogeneous networks, the distribution of the training data among the workers affects the model accuracy, in particular, its evolution over number of epochs. We found that the better the training data at the dominating worker represents the ground-truth data, the higher the achieved model accuracy. Furthermore, better ground-truth representation at the dominating worker can reduce by a factor of five the computational effort needed to achieve a certain accuracy level. We conclude that data transfers between workers before training can improve the training; however, we still need to take resource trade-offs for such transfers into account.

We hence plan to extend this study, short-term, to a quantitative characterization of the efforts for training and data load arising from different data sets and models. Further,

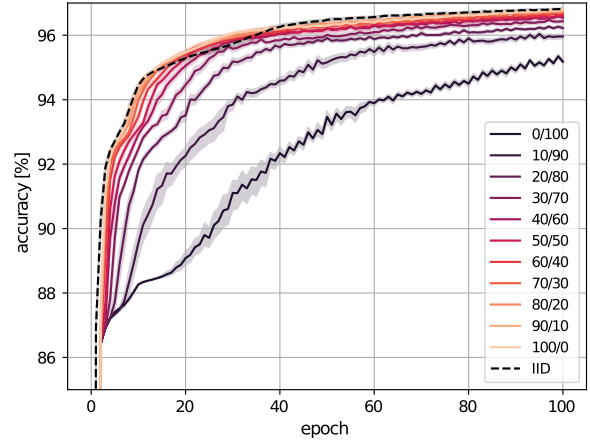


Fig. 1. Accuracy over epoch for different data distributions

we want to evaluate how susceptible different distributed ML architectures (e.g., All-Reduce or Federated Learning) are to data distribution among workers in inhomogeneous networks.

### V. ACKNOWLEDGMENTS

The authors acknowledge the financial support by the Federal Ministry of Education and Research of Germany in the project “Open6GHub” (grant number: 16KISK011).

### REFERENCES

- [1] A. Smola and S. Narayanamurthy, “An architecture for parallel topic models,” *Proceedings VLDB Endowment*, vol. 3, no. 1-2, pp. 703–710, Sep. 2010.
- [2] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su, “Scaling distributed machine learning with the parameter server,” 2014.
- [3] J. Dean, G. S. Corrado, R. Monga, K. Chen, M. Devin, Q. V. Le, M. Z. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, and A. Y. Ng, “Large scale distributed deep networks,” *Advances in neural information processing systems*, vol. 25, 2012.
- [4] L. Paeleke and H. Karl, “Wip: How latency differences between workers affect parameter server training,” 2023, access: June 8, 2023. [Online]. Available: [https://netaisys.github.io/wip\\_papers/netaisys23-final78.pdf](https://netaisys.github.io/wip_papers/netaisys23-final78.pdf)
- [5] J. Chen, X. Pan, R. Monga, S. Bengio, and R. Jozefowicz, “Revisiting distributed synchronous sgd,” *arXiv preprint arXiv:1604.00981*, 2016.
- [6] Q. Ho, J. Cipar, H. Cui, S. Lee, J. K. Kim, P. B. Gibbons, G. A. Gibson, G. Ganger, and E. P. Xing, “More effective distributed ml via a stale synchronous parallel parameter server,” *Advances in neural information processing systems*, vol. 26, 2013.
- [7] J. Jiang, B. Cui, C. Zhang, and L. Yu, “Heterogeneity-aware distributed parameter servers,” in *Proceedings of the 2017 ACM International Conference on Management of Data*, ser. SIGMOD ’17. New York, NY, USA: Association for Computing Machinery, May 2017, pp. 463–478.
- [8] X. Lian, W. Zhang, C. Zhang, and J. Liu, “Asynchronous decentralized parallel stochastic gradient descent,” in *International Conference on Machine Learning*. PMLR, 2018, pp. 3043–3052.
- [9] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, “A detailed analysis of the kdd cup 99 data set,” in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 2009, pp. 1–6.
- [10] P. Moritz, R. Nishihara, S. Wang, A. Tumanov, R. Liaw, E. Liang, M. Elibol, Z. Yang, W. Paul, M. I. Jordan, and I. Stoica, “Ray: A distributed framework for emerging AI applications,” Dec. 2017.