

# Decentralized Federated Learning and Network Topologies: an Empirical Study on Convergence

Hanna Kavalionak<sup>1</sup>, Emanuele Carlini<sup>1</sup>, Patrizio Dazzi<sup>3</sup>, Luca Ferrucci<sup>1</sup>,  
Matteo Mordacchini<sup>2</sup> and Massimo Coppola<sup>1</sup>

<sup>1</sup>*Institute of Information Science and Technologies (ISTI), National Research Council (CNR), Pisa, Italy*

<sup>2</sup>*Institute of Informatics and Telematics (IIT), National Research Council (CNR), Pisa, Italy*

<sup>3</sup>*Department of Computer Science, University of Pisa, Italy*

## Abstract

Federated Learning is a well-known learning paradigm that allows the distributed training of machine learning models. Federated Learning keeps data in the source devices and communicates only the model's coefficients to a centralized server. This paper studies the decentralized flavor of Federated Learning. A peer-to-peer network replaces the centralized server, and nodes exchange model's coefficients directly. In particular, we look for empirical evidence on the effect of different network topologies and communication parameters on the convergence in the training of distributed models. Our observations suggest that small-world networks converge faster for small amounts of nodes, while xx are more suitable for larger setups.

## Keywords

Federated Learning, Peer-to-Peer, Distributed Systems,

## 1. Introduction

Federated Learning (FL) [1] is one of the most popular trends in research communities dealing with distributed machine learning [2, 3] and decentralized intelligence [4, 5]. At its core, FL is a distributed framework in which machine learning models are trained over a set of devices, with each device having a small subset of the whole training data. Each node then runs a learning process locally. The coefficients computed during the learning phase are periodically sent to a central aggregation entity, e.g., a remote server. The server collects the model coefficients from all nodes of the system and aggregates them into the working model, which is then pushed back to the nodes. FL can find application in many fields [1], including mobile applications [6], healthcare [7], and autonomous vehicles [8].

FL offers a series of out-of-the-box benefits that it makes it attractive to academia and industry, including: (i) Models constantly improve using nodes data with no need to aggregate data in a

---


*SEBD 2022: The 30th Italian Symposium on Advanced Database Systems, June 19-22, 2022, Tirrenia (PI), Italy*

✉ hanna.kavalionak@isti.cnr.it (H. Kavalionak); emanuele.carlini@isti.cnr.it (E. Carlini); patrizio.dazzi@unipi.it (P. Dazzi); luca.ferrucci@isti.cnr.it (L. Ferrucci); matteo.mordacchini@iit.cnr.it (M. Mordacchini); massimo.coppola@isti.cnr.it (M. Coppola)

🆔 0000-0002-8852-3062 (H. Kavalionak); 0000-0003-3643-5404 (E. Carlini); 0000-0001-8504-1503 (P. Dazzi); 0000-0003-4159-0644 (L. Ferrucci); 0000-0002-1406-828X (M. Mordacchini); 0000-0002-7937-4157 (M. Coppola)



© 2022 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

centralized server; (ii) Source data are kept close to the source enabling privacy-aware learning; (iii) FL also fits situations when the network is limited in bandwidth and latency.

Recently, research communities devoted much effort to decentralizing the learning process. Nevertheless, the study of the benefits of fully decentralized against centralized solutions is still a challenge and is of particular interest [9]. The decentralized learning process assumes nodes (i.e., *peers*) to be part of an unstructured peer-to-peer network (i.e., *overlay*) that can be used to directly communicate with a subset of the other nodes in the network. The training is an iterative process: after performing the training of the model locally, the nodes send their model to the neighbours using point-to-point communications and the aggregate the coefficients. Due to the nature of decentralized communication, the model can converge to stable values only after a certain number of iterations. Decentralized FL retains all the benefit of traditional FL, but also allows a distributed learning process without the need to setup and operate a centralized entity to aggregate model coefficients. In this paper, we are interested in empirically measuring how sensitive is the convergence of a decentralized FL system to network topologies and communication parameters.

The content of this discussion paper is based on another manuscript [10] already published by the same authors, which contains a more detailed related work and the results of a large scale experiments.

## 2. Related work

Hegedus et al.[11] propose an empirical comparison of gossip-based learning and federated learning. The authors propose the evaluation of the effectiveness of both these technologies based on the three available datasets. In the paper, the authors consider a network with a fixed number of neighbors. Lian et al.[12] and Tang et al.[13] introduce the idea of applying gossip algorithms and evaluate their effectiveness in comparison with centralized solutions. Koloskova et al.[14] improves the algorithms proposed by Lian et al.[12] with arbitrary gradient compression. Lalitha et al.[15] propose a formal description and an algorithm for a distributed federated learning problem. The work of Savazzi et al.[16] studies gossip-based distributed solutions for the learning models. The focus of this work is on the way data is distributed between nodes in the system. Another work that relies on the data segmentation between the nodes for the learning is the work of Hu et al.[17]. The focus of this work is on bandwidth optimization rather than the speed of accuracy convergence. These works propose a solid theoretical idea for the concept of decentralized learning. Most of them emphasized how decentralization impacts the quality of training and, consequently, on the quality of the prediction. However, just a few explore the decentralized system's behavior in terms of the characteristics of the overlay and its convergence speed.

### 3. System model

#### 3.1. Federated Learning

Federated Learning comes in many flavors (see [18, 6]) and it is extensively used in many real world applications [1]. In this paper, we focus on the so-called *horizontal* FL, in which data on each node (or device) shares the same feature space but is composed of different, unique samples. A typical horizontal federated-learning system considers  $N$  nodes who collaboratively train a machine learning model.

More formally, each node  $i$  has a local dataset  $\mathcal{D}_i$ , such that the whole dataset  $\mathcal{D} = \{\mathcal{D}_1 \cup \dots \cup \mathcal{D}_N\}$  and that  $\mathcal{D}_i \cap \mathcal{D}_{i'} = \emptyset$  for  $i \neq i'$ . The training process of horizontal FL is agnostic with respect to the specific learning model used. Commonly used models in FL exploit a gradient-descent strategy to minimize the value of a loss function  $l(\cdot)$  (a function that gives a measure of the error), defined on the parameters vector  $w$ , i.e.  $l(w)$ .

Procedurally, the training phase of the model is usually composed of the following steps [1]: (i) the nodes train a local instance of the machine learning model by using their data. Afterward, The coefficients of the model are sent to a centralized server; (ii) the server performs an aggregation of the coefficients received by the nodes to create an "aggregated" model; (iii) the server sends back the aggregated model's coefficients to the nodes; (iv) nodes update their model with the aggregated one received. These four steps continue until the training step is completed (typically until the loss function of the aggregated model reaches convergence). Therefore, the ultimate objective of a FL system is to maximize the quality of the model at the centralized server, which is then sent to all the nodes in the system. FL comes with many security implications, especially in terms of data privacy and integrity. In this paper, we do not consider these security aspects. In reality, nodes would employ cryptography schemes, such as homomorphic encryption [19], to protect their privacy and avoid data leakage. While privacy is essential in the actual implementation of FL systems, our analysis is not affected by the (non) presence of privacy-preserving mechanisms.

#### 3.2. Decentralized Federated Learning

By comparison with the centralized FL, the general objective of decentralised learning is to minimize the average of the loss functions for all the nodes in the system. Indeed, the core difference between centralized and decentralized federated learning is, as the name implies, that the latter does not require a centralized server to work. Instead, the training phase and model distribution are performed only with peer-to-peer communications between nodes.

We model the communication network with a undirected graph of  $N$  nodes. We define the neighborhood of a user  $i$ , denoted as  $\mathcal{N}_i$ , as the set of nodes  $j$  that have an edge in the network going from  $i$  to  $j$ . We also consider the edge to be bi-directional, i.e. if  $j \in \mathcal{N}_i$  then  $i \in \mathcal{N}_j$ . Nodes communicate in rounds, with each rounds having the same length for each node,  $\Delta_g$ . In our model  $\mathcal{N}_i$  does not change over rounds and remains static.

The training procedure is the following: (i) Each node  $i$  initially trains its model on the local  $\mathcal{D}_i$  and obtain  $w_i$ ; (ii) for each  $\Delta_g$ , the  $w_i$  is communicated to a subset of  $\mathcal{N}_i$ , according to the available bandwidth. Here, we assume that each neighbour  $j$  also communicates back its own  $w_j$ . (iii) upon the reception of a  $w$  nodes update their model by means of an aggregation

	Pendigits	HAR	full-HAR
Size	10992	10000	499276
Number of features	16	93	93
Number of labels	10	4	4
Label distribution	balanced	4:3:2:1	4:3:2:1

**Table 1**  
Datasets properties

function  $\mathcal{A}(\cdot)$ . The aggregation function is a crucial element when considering the quality of a model. Since our focus is not on the quality, here we consider a simple aggregation function that performs an average of the coefficients [9].

## 4. Experimental Setup

All the experiments have been run in Python 3.8 on a single workstation machine. The Scikit-learn<sup>1</sup> library was used for the classifiers. The networks were generated with the NetworkX<sup>2</sup> library.

**Datasets.** For the experiments, we have considered two datasets prepared for multi-label classification tasks. The properties of the datasets are listed in Table 1. The first dataset is the Pendigits<sup>3</sup>, in which each sample contains features taken from handwritten digits from 250 different subjects. The task is to classify each image with the correct digits. In the second dataset, HAR<sup>4</sup>, four human activities (inactive, active, driving, walking) were recorded with a mobile app and put in relation with several sensors measurements in the phone. The task is to classify the activities using data from the accelerometer, gyroscope, magnetometer, and GPS. For most of the experiments, we randomly selected 10K samples from the original dataset (which consists of around 500K samples) to be similar in size to the Pendigits dataset.

The 10% part of each dataset is reserved for testing, and the remainder 90% is distributed to nodes for training. The training data has been divided among clients proportionally, i.e., each client roughly receives the same amount of samples. This means that when testing larger networks, the number of samples is lower per node: this is the cause of lower general quality in the prediction power of larger networks. Data does not change during the simulation.

**Models.** We used two machine learning classifiers in our experiments. The first one is a *stochastic gradient descent (SGD)* model that trains linear Support Vector Machine using the hinge loss function. The second one is a *Logistic Regressor* classifier. Since logistic regressors are used for binary classification problems, we use the one-vs-rest (OvR) strategy for the multi-class classification. The OvR subdivides the original multi-classification problem into multiple binary sub-problems (one for each class) and trains a model for each sub-problem. The model that

<sup>1</sup><https://scikit-learn.org/>

<sup>2</sup><https://networkx.org/>

<sup>3</sup><http://archive.ics.uci.edu/ml/datasets/pen-based+recognition+of+handwritten+digits>

<sup>4</sup><https://lbd.udc.es/research/real-life-HAR-dataset/>

obtains the best results is used for the prediction.

**Networks.** We use three different types of networks derived from graph theory to build the *overlay* between nodes. For each overlay, we assume a bidirectional communication channel that corresponds to an undirected graph. Nodes have no self-edges or double edges. All networks are also fully connected.

- *Regular random graph.* Every node has the same degree. Neighbors are chosen randomly.
- *Small-world graph.* It is built with the Watts–Strogatz technique<sup>5</sup>. It has short average path lengths, i.e., the distance between two randomly chosen nodes is proportional to the logarithm of the graph’s size. Some social networks and biological networks can be modeled with a small-world graph.
- *Scale-free graph.* It is built with the Barabási–Albert preferential attachment technique. These graphs have nodes, called hubs, with a disproportionately large degree compared with other nodes. Many networks can be modeled as a scale-free graph, including the Internet and many social networks.

These networks are characterized by two parameters. The first one is the size of the network  $N = \{32, 64, 128, 256, 512\}$ . The second parameter  $K = \{5, 10, 15, 20\}$  controls how nodes connect to each other. The semantic of this parameter is different with respect to the network considered. In the random graph, it defines the degree of the nodes; in the small-world graph, it defines the number of nearest neighbors that each node connects to in the ring topology; in the scale-free graph, it defines the number of edges to attach from a new node to existing nodes.

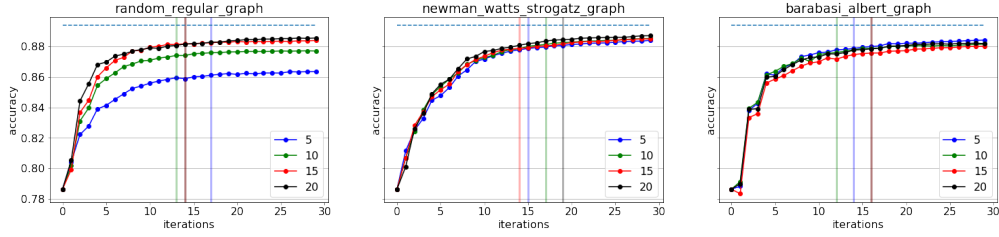
**Communication.** The communication among nodes is divided into synchronous rounds, or iterations, in which each node has a chance to communicate and exchange the model coefficients with another node. Each node can communicate only with its neighbors on the network. An iteration terminates when all nodes have had the chance to communicate. Communication between nodes is affected in two ways. First, we simulate connections between nodes not working properly by dropping some communication with various percentages,  $d = \{0, 0.1, 0.2, 0.3\}$ . When a communication is dropped the corresponding model information is lost. Second, we define the maximum number of nodes communications per rounds,  $c = \{1, 2, 4, 8\}$ . For example, when  $c = 1$ , each node can communicate with only one neighbor at each iteration. The selection of neighbors is made according to a round-robin algorithm. When  $\mathcal{N}_i < c_i$ , node  $i$  communicates with all its neighbours at each iteration.

## 5. Evaluation

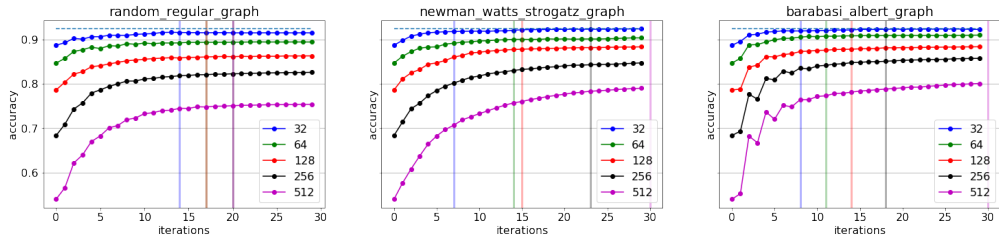
The system is evaluated by considering how fast each experiment converges to a value for the model *accuracy*, i.e., the fraction of correctly classified items. We have also measured other quality metrics such as precision, zero-one-loss, and f1 score, and we observed that the results are comparable with the accuracy. The *system accuracy* is the average of the accuracy values

---

<sup>5</sup>The probability of adding a new edge for each edge is set to 0.2



**Figure 1:** Accuracy over time with different networks in relation to degree. The dotted horizontal line is the accuracy obtained with the centralized FL. The vertical lines indicate when a given series (indicated by the color) reached convergence. The plots are generated with  $c = 1$ ,  $d = 0.3$ ,  $n = 128$ , and the Pendigits dataset.



**Figure 2:** Accuracy over time with different networks in relation to network size. The dotted horizontal line is the accuracy obtained with the centralized version and  $n = 32$ . The vertical lines indicate when a given series (indicated by the color) reached convergence. The plots are generated with  $c = 1$ ,  $d = 0.3$ ,  $k = 5$ , and the Pendigits dataset.

of all nodes in the network. It is computed at the end of every iteration. The convergence is measured by counting the difference between consecutive measurements of the system accuracy. We consider the system to have converged if, for three consecutive times, the accuracy is not lower than the previous value and the difference is less than 0.001.

Figure 1 shows the influence of the degree parameter  $K$  on the convergence and the overall accuracy of the decentralized learning. On the random graph, the degree has a high impact in terms of overall classification accuracy. In particular, a 5 degrees network converges to an accuracy slightly above 86%, whereas the 20 degrees network converges to 88%. The speed convergence is also affected, with the 10 degrees network converging earliest at the 13th iteration. The small-world network obtains similar values for the final accuracy. There is a marginal difference between the various degree values, having a similar trend in all cases. Similar to the small-world network, the scale-free network shows marginal differences in all cases. We can notice significant "steps" in the increment of accuracy due to those iterations in which high-degree nodes are updated with better models.

Figure 2 shows how the accuracy varies over time with the size of the network  $N$ . It is worth noticing that the lower maximum accuracy obtained by larger networks is because the amount of data per node is much lower, skewing the global accuracy drastically. However, we can notice that smaller networks reach convergence faster than larger ones, but no significant differences can be seen for different kind of networks.

HAR	Log. Reg	Scale-free	0.87	10.03
		Random Graph	0.86	7.62
		Small-world	0.87	6.33
	SGD	Scale-free	0.88	11.33
		Random Graph	0.87	10.11
		Small-world	0.88	8.22
Pendigits	Log. Reg	Scale-free	0.86	12.59
		Random Graph	0.86	9.11
		Small-world	0.86	7.12
	SGD	Scale-free	0.89	14.50
		Random Graph	0.89	11.17
		Small-world	0.89	9.73

**Table 2**

Accuracy and Convergence speed for different datasets, models, and network type.

Table 2 reports aggregated results of comparing all experiments varying multiple parameters with a network of fixed size ( $N = 128$ ). The global system accuracy is only slightly influenced by the various parameter configurations, with better results for the SDG model. In terms of convergence, it is clear how topology has a relevant impact, as the small-world networks converge more rapidly than the other networks on average.

## 6. Conclusion

Although decentralized federated learning is gaining momentum, only a few works analyze its behavior related to the network topology. This paper evaluated the impact of different network characteristics on the convergence speed of a decentralized federated learning system. In particular, we have empirically evaluated how sensitive is the training process for the various network characteristics and parameters.

Our results suggest that clustered networks such as scale-free and small-world look more suitable to support decentralized training than other kind of networks. In particular, small-world networks seem to converge faster for a small setup when the amount of training sample per node is relatively low. By comparison, scale-free networks obtained better results in the large-scale test. That could indicate that having a hierarchical organization of the network in which hubs (or super peers) can collect the aggregated models and redistribute them could increase the convergence speed. Naturally, this is a trade-off, as hierarchical systems also increase the load on specific nodes and are sensitive to the single point of failure (if a super-peer becomes unavailable, much valuable information is lost). We reserve to study these trade-offs in future work, also considering dynamic networks, i.e., that changes during the training phase.

## Acknowledgments

This work has been partially supported by the European Union’s Horizon 2020 Research and Innovation program, under the project TEACHING (Grant agreement ID: 871385).

## References

- [1] Q. Yang, Y. Liu, T. Chen, Y. Tong, Federated machine learning: Concept and applications, *ACM Transactions on Intelligent Systems and Technology (TIST)* 10 (2019) 1–19.
- [2] P. Kairouz, et al., Advances and open problems in federated learning., *CoRR abs/1912.04977* (2019).
- [3] K. Niwa, N. Harada, G. Zhang, W. B. Kleijn, Edge-Consensus Learning: Deep Learning on P2P Networks with Nonhomogeneous Data, Association for Computing Machinery, New York, NY, USA, 2020, p. 668–678.
- [4] H. Kavalionak, et al., Edge-based video surveillance with embedded devices, in: Proc. of the 28th Italian Symp. on Advanced Database Systems, Villasimius, Italy, June 21-24, volume 2646 of *CEUR Workshop Proc.*, CEUR-WS.org, 2020.
- [5] L. Ferrucci, M. Mordacchini, M. Coppola, E. Carlini, H. Kavalionak, P. Dazzi, Latency preserving self-optimizing placement at the edge, *FRAME '21*, Association for Computing Machinery, New York, NY, USA, 2020, p. 3–8.
- [6] W. Y. B. Lim, et al., Federated learning in mobile edge networks: A comprehensive survey, *IEEE Comm. Surveys & Tutorials* 22 (2020).
- [7] N. Rieke, et al., The future of digital health with federated learning, *NPJ digital medicine* 3 (2020) 1–7.
- [8] S. R. Pokhrel, J. Choi, A decentralized federated learning approach for connected autonomous vehicles, in: 2020 IEEE Wireless Comm. and Networking Conf. Workshops (WCNCW), IEEE, 2020, pp. 1–6.
- [9] Q. Xia, W. Ye, Z. Tao, J. Wu, Q. Li, A survey of federated learning for edge computing: Research problems and solutions, *High-Confidence Computing* (2021).
- [10] H. Kavalionak, et al., Impact of network topology on the convergence of decentralized federated learning systems, in: 2021 IEEE Symp. on Comp. and Comm. (ISCC), IEEE, 2021.
- [11] I. Hegedűs, G. Danner, M. Jelasity, Decentralized learning works: An empirical comparison of gossip learning and federated learning, *Journal of Parallel and Dist. Comp.* 148 (2021).
- [12] X. Lian, C. Zhang, H. Zhang, C. Hsieh, W. Zhang, J. Liu, Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent, in: *Advances in Neural Inf. Processing Systems 30: Annual Conf. on Neural Inf. Processing Systems 2017*, December 4-9, 2017, Long Beach, CA, USA, 2017.
- [13] H. Tang, X. Lian, M. Yan, C. Zhang, J. Liu,  $d^2$ : Decentralized training over decentralized data, in: Proc. of the 35th Int. Conf. on Machine Learning, volume 80 of *Proc. of Machine Learning Research*, PMLR, 2018, pp. 4848–4856.
- [14] A. Koloskova, S. Stich, M. Jaggi, Decentralized stochastic optimization and gossip algorithms with compressed communication, in: Proc. of the 36th Int. Conf. on Machine Learning, volume 97 of *Proc. of Machine Learning Research*, PMLR, 2019, pp. 3478–3487.
- [15] A. Lalitha, S. Shekhar, T. Javidi, F. Koushanfar, Fully decentralized federated learning, in: Third workshop on Bayesian Deep Learning (NeurIPS), 2018.
- [16] S. Savazzi, M. Nicoli, V. Rampa, Federated learning with cooperating devices: A consensus approach for massive iot networks, *IEEE Internet Things J.* 7 (2020) 4641–4654.
- [17] C. Hu, J. Jiang, Z. Wang, Decentralized federated learning: A segmented gossip approach, 2019. [arXiv:1908.07782](https://arxiv.org/abs/1908.07782).
- [18] T. Li, A. K. Sahu, A. Talwalkar, V. Smith, Federated learning: Challenges, methods, and future directions, *IEEE Signal Processing Magazine* 37 (2020) 50–60.
- [19] A. Acar, H. Aksu, A. S. Uluagac, M. Conti, A survey on homomorphic encryption schemes: Theory and implementation, *ACM Comp. Surveys (CSUR)* 51 (2018) 1–35.