Impact of Network Topology on the Convergence of Decentralized Federated Learning Systems

Hanna Kavalionak
ISTI-CNR
National Research Council
Pisa, Italy
hanna.kavalionak@isti.cnr.it

Emanuele Carlini
ISTI-CNR
National Research Council
Pisa, Italy
emanuele.carlini@isti.cnr.it

Patrizio Dazzi
ISTI-CNR
National Research Council
Pisa, Italy
patrizio.dazzi@isti.cnr.it

Luca Ferrucci
ISTI-CNR
National Research Council
Pisa, Italy
luca.ferrucci@isti.cnr.it

Matteo Mordacchini
IIT-CNR
National Research Council
Pisa, Italy
matteo.mordacchini@iit.cnr.it

Massimo Coppola
ISTI-CNR
National Research Council
Pisa, Italy
massimo.coppola@isti.cnr.it

***Abstract*—Federated learning is a popular framework that enables harvesting edge resources' computational power to train a machine learning model distributively. However, it is not always feasible or profitable to have a centralized server that controls and synchronizes the training process. In this paper, we consider the problem of training a machine learning model over a network of nodes in a fully decentralized fashion. In particular, we look for empirical evidence on how sensitive is the training process for various network characteristics and communication parameters. We present the outcome of several simulations conducted with different network topologies, datasets, and machine learning models.**

***Index Terms*—peer-to-peer, federated learning, network topology**

I. Introduction

The wide adoption of handheld devices and the Internet of Things (according to current statistics, the number of sensors connected to the internet will reach the impressive amount of 1 trillion in 2030 [17]) together with an incredible increase in the diffusion of mobile applications is leading to an exponential growth of the data generated at the edge of the network, in a pace that is threatening the network capacity of the Internet [4], [20]. The direct consequence of this dynamics in data is that it is becoming unfeasible or unprofitable to move all the data generated at the edge of the network to a remote central cloud. We are moving from performing data-intensive computations in public or private clouds to performing these computations locally at the edge [3], [18]. This translates to the need to enable in-place computations and then bring processing logic where the data is generated.

The Edge computing paradigm represents a concrete attempt to address this need [24]. Edge computing brings the *Utility Computing* paradigm [2] (i.e., the idea of using and renting computing capacity as any other utility) to a fully decentralized computing infrastructure, i.e., end-users and their applications can take advantage of a pervasive computing presence to which offload or run computing tasks.

Typically, multiple Edge nodes (or *Edge peers*) work together with a remote cloud to realize a "so-called" Cloud/Edge Continuum, namely a computing environment in which tasks and data move from the Cloud to the Edges and vice-versa. This kind of system organization also suits the structure of AI-enhanced applications organized according to the *Federated Learning* paradigm [12], [19], [28]. With Federated Learning (FL), each node at the edge of the network runs a learning process locally. The coefficients computed during the learning phase are periodically sent to a central aggregation entity, e.g., a remote Cloud server. Such an entity collects the model coefficients from nodes of the system and aggregates them into the working model, which is then pushed back to the Edge Peers.

The models trained in this way find wide application in different fields, including those scenarios in which the utilization of the data collected at the edge is associated with strict data protection rules [28]. Keeping sensitive data close to the source makes distributed solutions especially useful in cases when data privacy is sensitive. Moreover, these solutions are also beneficial when the network is limited in bandwidth or has restrictions in latency. Nevertheless, the centralization of the parameters aggregation brings the bottlenecks and missing possibilities to such an approach.

An alternative to centralized federated learning is the decentralized one. The decentralized learning process assumes Edge nodes to be part of an unstructured peer-to-peer network that can be used to directly communicate with a subset of Edge peers in the network (see Figure 1). Edge peers gather data from their connected end-users and train over this data to realize advanced services. In such an

978-1-6654-2744-9/21/$31.00 ©2021 IEEE

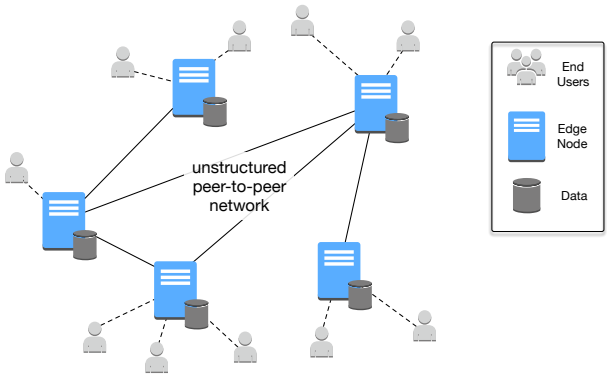


Fig. 1: Decentralized scenario. Edge nodes manages data of a set of end users and connect to each other via a peer-to-peer network.

approach, the Edge peers, after performing the training of the model locally, do not send their model coefficients to some central entity. The aggregation of the locally archived coefficients between the entities composing the network is fully decentralized. Namely, Edge peers exchange and aggregate the coefficients between themselves by using point-to-point communications.

Several recent works (such as [6], [15], [25]) have explored decentralized solution for FL. 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. In this paper, we are interested in the following research question: *how sensitive is the convergence of a decentralized FL system to network characteristics?* To this end, we made an effort to answer this question by observing the convergence speed in a set of training simulations in decentralized settings with different characteristics, including network topologies and communication parameters.

II. Related work

Federated learning is one of the most popular trends in research communities dealing with distributed machine learning [5], [9], [10], [21]. 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. Recently, research communities devoted much effort toward the decentralization of the learning process. Nevertheless, the study of the benefits of fully decentralized against centralized solutions is still a challenge and is of particular interest [27].

Hegedus et al. [6] 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. [15] and Tang et al. [25] introduce the idea of applying gossip algorithms and evaluate their effectiveness in comparison with centralized solutions. Koloskova et al. [11] improves the algorithms proposed by Lian et al. [15] with arbitrary gradient compression. Lalitha et al. [13] propose a formal description and an algorithm for a distributed federated learning problem. These works propose a solid theoretical idea for the concept of decentralized learning. Nevertheless, the actual evaluation of the impact of different distributed topologies is mostly missing.

The work of Savazzi et al. [23] 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. [7]. The focus of this work is on bandwidth optimization rather than the speed of accuracy convergence. Although not directly connected with the scope of our investigation, but still very relevant to the work of Wang et al. [26]. In their research, they propose to group the devices based on their data similarity, followed by selecting the devices with the best performance capacity. The authors evaluate the impact of data heterogeneity on device selection, model convergence, model accuracy, and fault tolerance in a federated learning setting.

A.Reisizadeh et al. propose FedPAQ [22] federated learning method. In their work, they address the communication and scalability challenges in federated learning by applying three key points: (1) periodical averaging with the server, (2) only a fraction of the devices participating in each round of model training, and (3) the edge nodes quantifies their updates before the parameters uploads. Jameel et al. [8] propose to rely on a subset of super-peers that are part of a ring topology. Each super-peer connects to a subset of the ordinary peers. Nevertheless, this work focuses mainly on the description ring topology.

While most of the existing works concentrate on the modeling of the distributed federated learning or its optimization, in our work, we focused on the impact that different distributed topologies have on the learning accuracy and convergence speed.

III. System model

A. Federated Learning

Federated Learning comes in many flavors (see [14], [16]) and it is extensively used in many real world applications [28]. 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 $\mathcal{D} = \{\mathcal{D}_1 \cup \cdots \cup \mathcal{D}_N\}$ 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 [28]: (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 [1], 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.

B. 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, Δ_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 Δ_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 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 [27].

	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 I: Datasets properties

IV. Experiments

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

Datasets. For the experiments, we have considered two datasets prepared for multi-label classification tasks. The properties of the datasets are listed in Table I. The first dataset is the Pendigits[3], 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[4], 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 Pendigts 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 obtains the best results is used for the prediction.

[1]https://scikit-learn.org/

[2]https://networkx.org/

[3]http://archive.ics.uci.edu/ml/datasets/pen-based+recognition+of+handwritten+digits

[4]https://lbd.udc.es/research/real-life-HAR-dataset/

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[5]. 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 models 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 accordion to a round-robin algorithm. When $\mathcal{N}_i < c_i$, node i communicates with all its neighbours at each iteration.

V. 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 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 2 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 3 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.

Table II 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.

Finally, in Figure 4 are reported the results of a large scale experiments conducted with the full HAR dataset over a large (N=1024) network. These results show that in large-scale setups, a scale-free network seems to yield better results in terms of convergence than the other networks.

VI. 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

[5]The probability of adding a new edge for each edge is set to 0.2

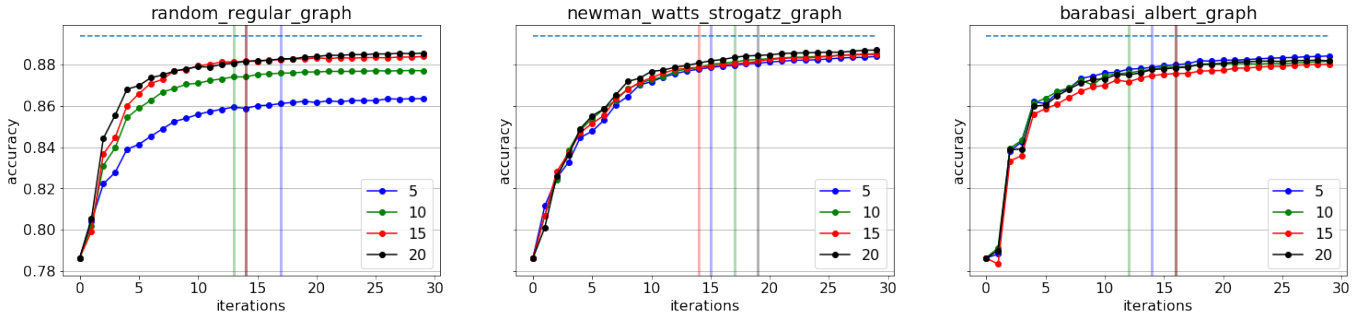


Fig. 2: Accuracy over time with different networks in relation to degree. The dotted horizontal line is the accuracy obtained with the centralized FL. The vertical lines indicate when a given series (indicated by the color) reached convergence. The plots are generated with $c = 1$, $d = 0.3$, $n = 128$, and the Pendigits dataset.

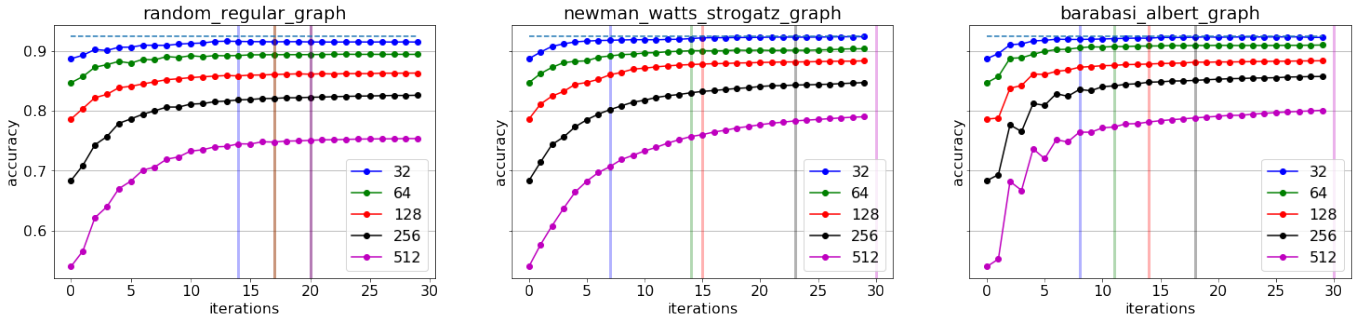


Fig. 3: Accuracy over time with different networks in relation to network size. The dotted horizontal line is the accuracy obtained with the centralized version and $n = 32$. The vertical lines indicate when a given series (indicated by the color) reached convergence. The plots are generated with $c = 1$, $d = 0.3$, $k = 5$, and the Pendigits dataset.

Dataset	Model	Network	Acc.	Conv.
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 II: Accuracy and Convergence speed for different datasets, models, and network type. Values are calculated on network of $n = 128$ by averaging the results obtained with various degree, bandwidth and churn.

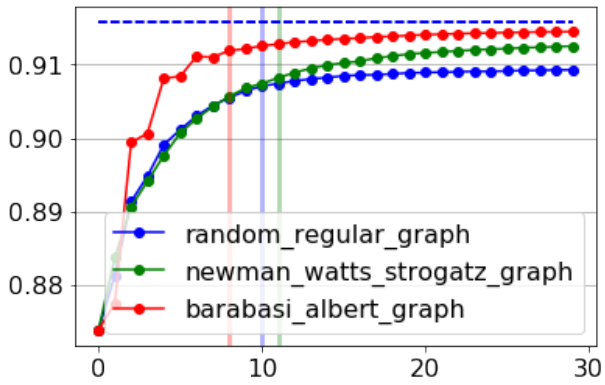


Fig. 4: Accuracy over time in relation to different networks. The dotted horizontal line is the accuracy obtained with the centralized FL. The vertical lines indicate when a given series (indicated by the color) reached convergence. The experiment was run with $N = 1024$, $c = 1$, $d = 0$, and the full-HAR dataset.

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. In particular, small-world network seems to converge faster for a small setup when the amount of training sample per node is relatively low. By comparison, scale-free network 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.

VII. 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] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys (CSUR)*, 51(4):1–35, 2018.

[2] J. Broberg, S. Venugopal, and R. Buyya. Market-oriented grids and utility computing: The state-of-the-art and future directions. *Journal of Grid Computing*, 6(3):255–276, 2008.

[3] H. Cao, M. Wachowicz, C. Renso, and E. Carlini. Analytics everywhere: generating insights from the internet of things. *IEEE Access*, 7:71749–71769, 2019.

[4] M. Chiang and T. Zhang. Fog and iot: An overview of research opportunities. *IEEE Internet of Things Journal*, 3(6):854–864, 2016.

[5] L. Ferrucci, M. Mordacchini, M. Coppola, E. Carlini, H. Kavalionak, and P. Dazzi. Latency preserving self-optimizing placement at the edge. FRAME '21, page 3–8, New York, NY, USA, 2020. Association for Computing Machinery.

[6] I. Hegedűs, G. Danner, and M. Jelasity. Decentralized learning works: An empirical comparison of gossip learning and federated learning. *Journal of Parallel and Distributed Computing*, 148:109–124, 2021.

[7] C. Hu, J. Jiang, and Z. Wang. Decentralized federated learning: A segmented gossip approach, 2019.

[8] M. Jameel, J. Grabocka, M. ul Islam Arif, and L. Schmidt-Thieme. Ring-star: A sparse topology for faster model averaging in decentralized parallel SGD. In P. Cellier and K. Driessens, editors, *Machine Learning and Knowledge Discovery in Databases - International Workshops of ECML PKDD 2019, Würzburg, Germany, September 16-20, 2019, Proceedings, Part I*, volume 1167 of *Communications in Computer and Information Science*, pages 333–341. Springer, 2019.

[9] P. Kairouz et al. Advances and open problems in federated learning. *CoRR*, abs/1912.04977, 2019.

[10] H. Kavalionak, C. Gennaro, G. Amato, C. Vairo, C. Perciante, C. Meghini, F. Falchi, and F. Rabitti. Edge-based video surveillance with embedded devices. In *Proceedings of the 28th Italian Symposium on Advanced Database Systems, Villasimius, Sud Sardegna, Italy (virtual due to Covid-19 pandemic), June 21-24, 2020*, volume 2646 of *CEUR Workshop Proceedings*, pages 278–285. CEUR-WS.org, 2020.

[11] A. Koloskova, S. Stich, and M. Jaggi. Decentralized stochastic optimization and gossip algorithms with compressed communication. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3478–3487. PMLR, 09–15 Jun 2019.

[12] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon. Federated learning: Strategies for improving communication efficiency, 2017.

[14] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[13] A. Lalitha, S. Shekhar, T. Javidi, and F. Koushanfar. Fully decentralized federated learning. In *Third workshop on Bayesian Deep Learning (NeurIPS)*, 2018.

[15] X. Lian, C. Zhang, H. Zhang, C. Hsieh, W. Zhang, and J. Liu. Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5330–5340, 2017.

[16] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 22(3):2031–2063, 2020.

[17] M. Marjani, F. Nasaruddin, A. Gani, A. Karim, I. A. T. Hashem, A. Siddiqa, and I. Yaqoob. Big iot data analytics: Architecture, opportunities, and open research challenges. *IEEE Access*, 5:5247–5261, 2017.

[18] M. Marzolla, M. Mordacchini, and S. Orlando. A p2p resource discovery system based on a forest of trees. In *17th International Workshop on Database and Expert Systems Applications (DEXA'06)*, pages 261–265. IEEE, 2006.

[19] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In A. Singh and J. Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282, Fort Lauderdale, FL, USA, 20–22 Apr 2017. PMLR.

[20] M. Mordacchini, P. Dazzi, G. Tolomei, R. Baraglia, F. Silvestri, and S. Orlando. Challenges in designing an interest-based distributed aggregation of users in p2p systems. In *2009 International Conference on Ultra Modern Telecommunications & Workshops*, pages 1–8. IEEE, 2009.

[21] K. Niwa, N. Harada, G. Zhang, and W. B. Kleijn. *Edge-Consensus Learning: Deep Learning on P2P Networks with Nonhomogeneous Data*, page 668–678. Association for Computing Machinery, New York, NY, USA, 2020.

[22] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie, and R. Pedarsani. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In S. Chiappa and R. Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2021–2031. PMLR, 26–28 Aug 2020.

[23] S. Savazzi, M. Nicoli, and V. Rampa. Federated learning with cooperating devices: A consensus approach for massive iot networks. *IEEE Internet Things J.*, 7(5):4641–4654, 2020.

[24] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646, 2016.

[25] H. Tang, X. Lian, M. Yan, C. Zhang, and J. Liu. d^2: Decentralized training over decentralized data. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4848–4856. PMLR, 10–15 Jul 2018.

[26] Y. Wang, J. Wolfrath, N. Sreekumar, D. Kumar, and A. Chandra. *Accelerated Training via Device Similarity in Federated Learning*, page 31–36. Association for Computing Machinery, New York, NY, USA, 2021.

[27] Q. Xia, W. Ye, Z. Tao, J. Wu, and Q. Li. A survey of federated learning for edge computing: Research problems and solutions. *High-Confidence Computing*, page 100008, 2021.

[28] Q. Yang, Y. Liu, T. Chen, and Y. Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.