Subspace Parsimonious Dictionary Learning and its use in Federated Learning

Dimitris Ampeliotis
Department of Digital Media and Communication,
Ionian University, Greece
Email: ampeliotis@ionio.gr

Alexandros Gkillas
Industrial Systems Institute, Athena Research Center,
Patras Science Park, Greece
Email: gkillas@isi.gr

***Abstract*—A federated dictionary learning problem, in which the edge devices have data with different statistical properties, is considered. A dictionary learning cost function that can lead to dictionaries that do not exhaust their full representation capabilities is proposed. The edge devices utilize this algorithm to design the local dictionaries. Likewise, a new aggregation rule is proposed for the central server, that takes into account the particular form of the local dictionaries. Simulation results were conducted that demonstrated that the proposed approach leads to solutions that achieve smaller root mean square error, are more sparse, but most importantly, require significantly smaller numbers of federated learning rounds to achieve some required error.**

I. Introduction

Over the last years, we have witnessed a tremendous interest for data-driven learning methods [1]. In particular, the remarkable performance of these methods in solving difficult real-world problems has made them the option of choice for various modern scientific and technological applications [2], [3]. Two of the most prominent categories of data-driven methods are deep-learning approaches [4] and dictionary-learning [5]. However, the need to train increasingly complex data-driven models requires substantial computational resources and massive training datasets [6], making centralized computing architectures unsuitable. Decentralized approaches are necessary to address these requirements. Nevertheless, distributed learning is a challenging task and many researchers are developing efficient algorithms to accomplish this goal.

Distributed data-driven learning methods are generally classified into three primary categories. The first category involves distributed nodes (devices) that communicate with some central node, known as fusion server, to gather data for centralized processing, as noted in [7]. This approach is impractical in cases where the local nodes have large datasets. On the other side, fully decentralized strategies that are based on local information exchange among neighboring nodes have been developed in recent years, as discussed in [8]. More recently, a third category has emerged: Federated Learning (FL), as described in [9], takes an intermediate position between the first two categories, proposing a paradigm where data is collected locally at the edge users, and some processing is also performed locally. Still, global information is shared between a central server and the dispersed devices (or users), as demonstrated in Fig. 1.

Federated learning methods try to addresses the main limitations of the methods that fall in the other two categories, namely the limited storage and computational capabilities of fusion center-based methods, and the increased communication requirements of fully distributed architectures. In more detail, the FL paradigm proposes a protocol in which local nodes compute local data-driven learning models, and these models are transmitted to the central server. Thus, data exchange is avoided, an approach that requires significantly less communication and is more privacy-aware. In the sequel, the central node that receives models from a set of clients, employs a proper aggregation rule to derive a single, global, data-driven model that, hopefully, inherits characteristics from all the local models employed in the aggregation process. This approach is repeated for a number of so-called *federated learning rounds*. FL approaches can be classified into three primary categories [10]: horizontal FL, vertical FL, and federated transfer learning. These categories arise when the feature and/or index spaces of the distributed datasets are identical or not identical.

This study focuses on FL techniques for addressing the dictionary learning problem which has numerous applications [11], [12], [13]. While centralized algorithms for this problem are efficient [14], the research on decentralized methods has mainly focused on fully distributed strategies, such as [15], [16] and [17]. Recently, FL-based approaches have been proposed for dictionary learning [18], [19]. Unlike most previous works that focused on i.i.d. scenarios for dictionary learning, we suggest a suitable non-i.i.d. dictionary learning problem at the edge devices to enable dictionary aggregation at the central server. Additionally, we propose a proper aggregation rule to combine the dictionaries at the central server.

II. Problem Formulation

We consider a set of N nodes (in the following, we also use the terms users or edge devices to refer to these entities), where each node $n \in \mathcal{N} = \{1, 2, \ldots, N\}$ possesses a local dataset, that we represent by the matrix

$$\boldsymbol{Y}_n \in \mathbb{R}^{d \times m_n}, \qquad n \in \mathcal{N} \ , \tag{1}$$

where d denotes the dimension of the local data samples and m_n is the number of column vectors in the dataset of user n, while typically $m_n >> d$.

This work has received funding from the European Union's research and innovation programme TRUSTEE under grant agreement No 101070214.

979-8-3503-3959-8/23/$31.00 ©2023 IEEE

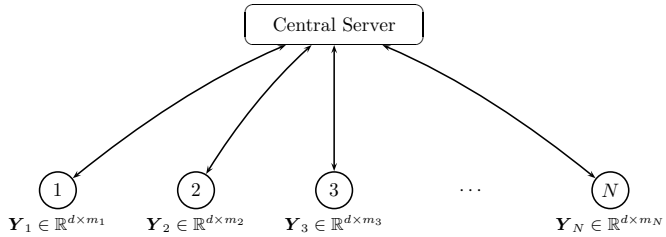

Fig. 1. The Federated Learning model considered, where a number of edge devices collaborate via a central server in order to solve some common data-driven learning task. Each node n has a local dataset $\boldsymbol{Y}_n$ that, in general, has different size and statistics.

The nodes are interested in computing a *common dictionary* matrix $\boldsymbol{D} \in \mathbb{R}^{d\times K}$ that is suitable for the sparse representation of the entire dataset, i.e. the concatenation of the datasets of all the users. Under the Federated Learning approach, the cooperation of the users for computing such a common dictionary is accomplished with the help of a central server, that is able to communicate with all the edge devices and follows an iterative procedure. In every federated learning round, each node estimates a local dictionary matrix, sends this estimate to the central server that applies some aggregation rule, and the aggregate dictionary is sent back to all the users, that utilize it to initialize the next local estimation procedure.

III. The Proposed Approach

A. Dictionaries and representation capability

Before proceeding to the details of the proposed approach, it is useful to mention some preliminaries regarding the representation of vectors using dictionaries. In essence, the goal of a dictionary $\boldsymbol{D} \in \mathbb{R}^{d\times K}$ is to provide sparse representations of vectors $\boldsymbol{y} \in \mathbb{R}^d$, as given by

$$\boldsymbol{y} \approx \boldsymbol{D}\boldsymbol{g} \ ,$$

where $\boldsymbol{g} \in \mathbb{R}^K$ is a sparse vector, in the sense that it contains only a few, say s, non-zero entries. Although, in a first look, this may seem as a significant restriction regarding the variety of vectors $\boldsymbol{y}$ that can be represented in this form, the model is quite general. This can be demonstrated if one considers that there are

$$S_{\boldsymbol{D},s} = \binom{K}{s} = \frac{K!}{s!(K-s)!}$$

possible choices for selecting the non-zero elements in vector $\boldsymbol{g}$, and thus, the Dictionary can represent all the vectors that belong to any of the $S_{\boldsymbol{D},s}$ linear subspaces described by selecting s atoms from $\boldsymbol{D}$. Clearly, this number of subspaces can be quite large. In the sequel, we use the term *capacity* to refer to this number of subspaces.

Among several questions that may arise in such a setting, one that is relevant to this work is the following: When performing dictionary learning using some data $\boldsymbol{Y}$, should we "consume" all such linear subspaces or is there any benefit from utilizing only a minimum number of such subspaces, provided that the error remains within some prescribed bound? Clearly, if all such subspaces are used, then the respective dictionary cannot easily accommodate new data that does not reside in any of such subspaces, without "forgetting" some of the data on which it has already been trained. This reasoning is quite relevant in distributed and federated dictionary learning: Edge users should employ dictionary learning methods that accurately describe their local data without utilizing all the capacity of their local dictionaries, having in mind that this facilitates the derivation of an aggregate dictionary at the central server that can represent the data of all the edge users.

B. Subspace parsimonious dictionary learning

Our scope in this sub-section is to derive a (centralized) dictionary learning method that tries to minimize the number of subspaces utilized. This is in agreement with the usual case where the edge devices do not have big and/or rich datasets. Our approach into deriving a subspace parsimonious dictionary learning method is to penalize the use of new atoms from the dictionary. To this end, focusing on any particular node n, we propose the following cost function

$$C_n(\boldsymbol{D}_n, \boldsymbol{G}_n) = \frac{1}{2}\left\|\boldsymbol{Y}_n - \boldsymbol{D}_n\boldsymbol{G}_n\right\|_{\mathrm{F}}^2 + \left\|\boldsymbol{\Lambda}_n\boldsymbol{G}_n\right\|_1 \ , \quad (2)$$

where $\boldsymbol{D}_n$ and $\boldsymbol{G}_n$ denote, respectively, the local dictionary and sparse approximation matrices at node n, while $\boldsymbol{\Lambda}_n \in \mathbb{R}^{K\times K}$ is a diagonal matrix defined as

$$\boldsymbol{\Lambda}_n = \begin{bmatrix} \lambda_{n,1} & 0 & \cdots & 0 \\ 0 & \lambda_{n,2} & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \lambda_{n,K} \end{bmatrix} . \quad (3)$$

The reasoning for selecting this cost function is that it enables a different penalty term for controlling the use of each atom. In particular, if the parameter $\lambda_{n,k}$ is large, then this implies a large penalty for the use of the k-th atom. Thus, it is possible to design the matrix $\boldsymbol{\Lambda}_n$, so that the dictionary learning process will try to utilize a small number of atoms. For example, one can select

$$\lambda_{n,k} = \lambda_{n,0} + (k-1)\cdot\delta_n \quad (4)$$

for some parameters $\lambda_{n,0}, \delta_n > 0$, so that the atoms with smaller index will yield smaller cost, and thus, will be preferred over atoms with greater index. Thus, some atoms with large indexes may not be used at all, thus arriving to a *subspace parsimonious* dictionary.

C. Federated dictionary learning

In the context of federated dictionary learning, where the edge devices may have non-i.i.d. data, it is reasonable to apply a different atom penalization strategy at each node. This approach will create groups of atoms that are mainly used at one edge device, leaving other atoms for the other nodes to utilize. It should be noted that this approach is different from using different dictionaries at each node, each with a limited number of atoms. Rather, the proposed approach can be seen as a *soft* version of a method that utilizes separate dictionaries at each node.

To achieve such an atom personalization approach, we let node n prefer atoms that have index close to some predefined atom index that we denote as $k_n \in \{1, 2, \ldots, K\}$. We compute the atom-index-difference between the k-th index and the

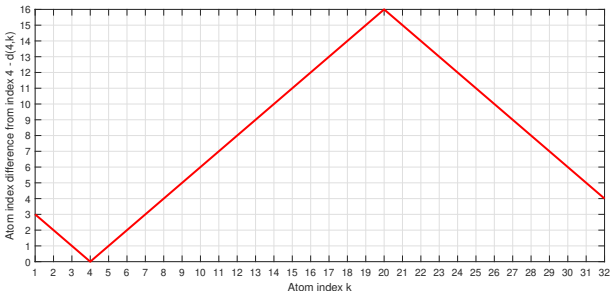

Fig. 2. An illustration of the atom-index-difference function. The function $d(4,k)$ is demonstrated, as a function of the atom index k. This function measures the atom index difference from the atom at $k_n = 4$, in a cyclical fashion. We have assumed that the number of atoms is $K = 32$.

"central" index k_n in a cyclical manner, as given by the expression

$$d(k_n, k) = min\left\{|k_n - k|, |k_n + K - k|, |k_n - K - k|\right\} \ .$$

To demonstrate this atom-index-difference function, Fig. 2 shows the form of an example function $d(4,k)$ as a function of the atom index k. The scope of this function is to provide an integer that measures the distance between k and k_n, considering a cyclical distance (e.g., atoms 1 and K have $d(1,K) = 1$).

The parameters $\lambda_{n,k}$ in a federated dictionary learning setting can be given by the expression

$$\lambda_{n,k} = \lambda_{n,0} + d(k_n, k) \cdot \delta_n \ , \tag{5}$$

in a fashion similar to Equation (4). Finally, the parameters k_n can be equally spaced in the interval $[1, K]$, for example

$$k_n = 1 + (n-1)\left\lfloor \frac{K}{N} \right\rfloor \ , \ n = 1, 2, \ldots, N \ . \tag{6}$$

D. Atom-specific aggregation rule

The proposed approach can be combined with a simple averaging rule, that gives the aggregated dictionary as the average of the dictionaries provided to the server by the clients. Also, other aggregation rules may be employed, for example, one that first aligns the atoms of the dictionaries so that the sum of Euclidean distances is minimized [20]. However, in this work, we propose an *atom-specific* combination rule that combines each column of the dictionary using independent weights. More specifically, we exploit the fact that the atoms have been (softly) assigned to the clients, in the sense that the atoms closer to the index k_n are mostly used by the n-th edge device. More generally, the smaller the value of $\lambda_{n,k}$ the more likely is that atom k is used by node n. Thus, we define the following weights

$$w_{n,k} = \frac{\max_{k'}(\lambda_{n,k'}) + \lambda_{n,0} - \lambda_{n,k}}{\sum_{n'=1}^{N} \max_{k'}(\lambda_{n',k'}) + \lambda_{n',0} - \lambda_{n',k}} \tag{7}$$

Using these weights, we define

$$\boldsymbol{W}_n = \begin{bmatrix} w_{n,1} & 0 & \cdots & 0 \\ 0 & w_{n,2} & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & w_{n,K} \end{bmatrix} \ , \tag{8}$$

and we arrive at the following atom-specific aggregation rule

$$\boldsymbol{D}_{agg} = \sum_{n=1}^{N} \boldsymbol{D}_n \cdot \boldsymbol{W}_n \ , \tag{9}$$

where $\boldsymbol{D}_{agg}$ denotes the aggregate dictionary computed at the server. It is easy to verify that the previous aggregation rule reduces to a simple averaging rule, in the case where $\delta_n = 0$.

IV. Numerical results

In order to validate the derivations presented in the previous, some numerical results were conducted. In particular, we simulated a federated dictionary learning scenario with $N = 4$ edge devices. The data of each edge device was created synthetically, where for each device 8 randomly selected atoms were employed to generate $m_n = 2000$ vectors of dimension $d = 32$. Each data vector was generated after randomly selecting 4 of the 8 atoms of each node, and generating 4 random Gaussian weights to construct a weighted average of the selected atoms. Thus, the data on each client has different statistical characteristics.

Each edge device tries to minimize its local cost, given in Equation (2), using an alternating optimization approach, where a proximal gradient descend algorithm is used to perform optimization regarding the matrix $\boldsymbol{G}_n$, while a projected (normalizing the atoms to have unit length is a projection operation) gradient descend algorithm is used to optimize the cost function regarding the local dictionary $\boldsymbol{D}_n$. Each edge device performs $T = 200$ such local alternating optimization steps before sending the dictionary obtained to the central server. It should be noted that the case in which the edge devices perform such a large number of iterations constitutes a more challenging problem for aggregation, since the dictionaries sent to the central server become significantly different [20]. The edge devices use a common value for the parameters $\lambda_{n,0}$ and δ_n. We simulate 50 federated learning rounds using the above settings, where the aggregation rule given by Equation (9) is employed at the central server. In order to demonstrate the effectiveness of the proposed approach, we compare its performance against an approach in which the matrix $\boldsymbol{\Lambda}_n$ is a diagonal matrix with elements equal to $\lambda_{n,0}$, i.e., the subspace parsimonious feature proposed is not in effect. It should be noted that, for $\delta_n = 0$ the proposed approach reduces to a scenario in which all nodes solve the simple LASSO problem, while the aggregation rule reduces to a simple averaging rule.

The global root mean square error as a function of the federated learning round is demonstrated in Fig. 3. We compare the performance of the simple scheme, corresponding to the case where $\delta_n = 0$, with the proposed subspace parsimonious approach. We conducted a large number of simulations for various values for the parameters $\lambda_{n,0}$ and δ_n and Fig. 3 shows the best results for each of the considered schemes. It is evident from Fig. 3 that the proposed approach, in the scenario considered here, is able to provide solutions with smaller root mean square error which are simultaneously more sparse, as compared to the simple approach. More importantly, the proposed approach converges significantly faster that the simple approach: For achieving a root mean square error equal to 0.01, the proposed approach requires 25 federated

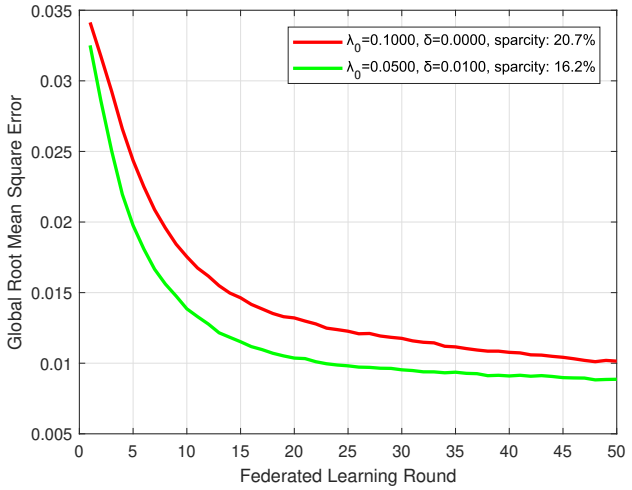

Fig. 3. Global root mean square error as a function of the federated learning round performed.

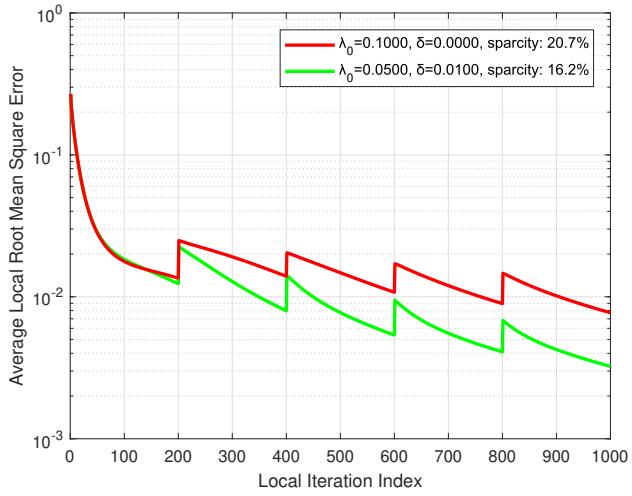

Fig. 4. Average local root mean square error as a function of the local iterations performed, focusing on the first 5 federated learning rounds

learning rounds while the simple approach requires more than 50 federated learning rounds.

In order to get a better insight of the internals of the considered federated dictionary learning approaches, Fig. 4 demonstrates the average (over the edge devices) local root mean square error, as a function of the local iteration index, where only the 5 first federated learning rounds are shown. From this figure, it is evident that the averaging operation performed at the central server has a negative effect to the local root mean square errors. This is well justified, since the data at the various nodes has different statistical characteristics, and the averaging operation tries to reduce such dissimilarities. Furthermore, we note from Fig. 4 that the proposed subspace parsimonious approach is able to better adapt to the particularities of the local data, as demonstrated by the faster reduction of the average local root mean square error after each dictionary aggregation round.

V. Conclusions

In this work, a federated dictionary learning problem with non-i.i.d. data at the edge devices was considered. A dictionary learning method that does not exhaust the number of linear subspaces used by the local dictionaries was proposed. The use of the proposed method in a federated dictionary learning scenario was studied and a proper aggregation rule was proposed. Simulation results were conducted on syntherically generated data, which demonstrated that the proposed approach yields solutions with smaller root mean square error, that are also more sparse, and converge significantly faster, as compared to a simpler LASSO-based federated dictionary learning approach.

References

[1] G. Strang, *Linear Algebra and Learning from Data*. Wellesley-Cambridge Press, 2019.

[2] F. Provost and T. Fawcett, "Data science and its relationship to big data and data-driven decision making," *Big data*, vol. 1, no. 1, pp. 51–59, 2013.

[3] S. J. Qin, "Survey on data-driven industrial process monitoring and diagnosis," *Annual reviews in control*, vol. 36, no. 2, pp. 220–234, 2012.

[4] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[5] I. Tošić and P. Frossard, "Dictionary learning," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 27–38, 2011.

[6] X.-W. Chen and X. Lin, "Big data deep learning: challenges and perspectives," *IEEE access*, vol. 2, pp. 514–525, 2014.

[7] A. Agarwal and J. C. Duchi, "Distributed delayed stochastic optimization," *Advances in neural information processing systems*, vol. 24, 2011.

[8] A. H. Sayed, *Adaptation, Learning, and Optimization over Networks*. Now Publishers, 2014.

[9] H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Federated learning of deep networks using model averaging," *arXiv preprint arXiv:1602.05629*, 2016.

[10] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[11] B. Dumitrescu and P. Irofti, *Dictionary learning algorithms and applications*. Springer, 2018.

[12] A. Gkillas, D. Ampeliotis, and K. Berberidis, "Efficient coupled dictionary learning and sparse coding for noisy piecewise-smooth signals: Application to hyperspectral imaging," in *IEEE International Conference on Image Processing (ICIP 2020)*, Abu Dhabi, United Arab Emirates, 25-28 October 2020.

[13] E.-V. Pikoulis, C. Mavrokefalidis, and A. S. Lalos, "A data-aware dictionary-learning based technique for the acceleration of deep convolutional networks," in *2021 IEEE 23rd International Workshop on Multimedia Signal Processing (MMSP)*, 2021, pp. 1–5.

[14] M. Aharon, M. Elad, and A. Bruckstein, "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on signal processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[15] J. Chen, Z. J. Towfic, and A. H. Sayed, "Dictionary learning over distributed models," *IEEE Transactions on Signal Processing*, vol. 63, no. 4, pp. 1001–1016, 2014.

[16] A. Daneshmand, G. Scutari, and F. Facchinei, "Distributed dictionary learning," in *2016 50th Asilomar Conference on Signals, Systems and Computers*. IEEE, 2016, pp. 1001–1005.

[17] D. Ampeliotis, C. Mavrokefalidis, and K. Berberidis, "Distributed dictionary learning via projections onto convex sets," in *European Signal Processing Conference (EUSIPCO 2017)*, Kos Island, Greece, August 28 - September 2 2017.

[18] K. Huang, X. Liu, F. Li, C. Yang, O. Kaynak, and T. Huang, "A federated dictionary learning method for process monitoring with industrial applications," *IEEE Transactions on Artificial Intelligence*, 2022.

[19] A. Gkillas, D. Ampeliotis, and K. Berberidis, "Federated dictionary

learning from non-iid data," in *2022 IEEE 14th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)*, 2022, pp. 1–5.

[20] D. Ampeliotis, C. Mavrokefalidis, K. Berberidis, and S. Theodoridis, "Adapt-align-combine for diffusion-based distributed dictionary learning," in *24th European Signal Processing Conference (EUSIPCO 2016)*, Budapest, Hungary, August 29 - September 2 2016.