Towards Model-Agnostic Federated Learning over Networks

Alexander Jung*†
*Silo AI
†Dept. of Computer Science, Aalto University
alex.jung@aato.fi
orcid.org/0000-0001-7538-0990, ResearcherID: M-4407-2016

***Abstract*—We present a model-agnostic federated learning method for decentralized data with an intrinsic network structure. The network structure reflects similarities between the (statistics of) local datasets and, in turn, their associated local ("personal") models. Our method is an instance of empirical risk minimization, with the regularization term derived from the network structure of data. In particular, we require well-connected local models, forming clusters, to yield similar predictions on a common test set. The proposed method allows for a wide range of local models. The only restriction put on these local models is that they allow for efficient implementation of regularized empirical risk minimization (training). Such implementations might be available in the form of high-level programming frameworks such as `scikit-learn`, `Keras` or `PyTorch`.**

***Index Terms*—federated learning, personalization, heterogeneous, non-parametric, complex networks**

I. Introduction

Many important application domains for machine learning (ML), such as numerical weather prediction, the internet of things or healthcare, generate decentralized data [1]. Decentralized data consists of local datasets that are related by an intrinsic network structure. Such a network structure might arise from relations between the generators of local datasets or functional constrains of the computational infrastructure [2], [3]. We can represent such networked data using an undirected weighted empirical graph [4, Ch. 11].

There is a substantial body of work on machine learning (ML) and signal processing models and techniques for graph structured data [3]–[6]. Most of existing work studies parametric models for local datasets that are related by an intrinsic network structure. Arguably the most basic setting are scalar graph signal-in-noise models using different smoothness or clustering assumptions [7], [8]. The extension from scalar signal-in-noise models to vector-valued graph signals and networked exponential families has been studied in [9], [10].

Federated learning (FL) is an umbrella term for collaborative training of ML models from decentralized data. FL methods have been championed for high-dimensional parametric models such as deep nets [11]–[13]. The focus of FL research so far has been on distributed optimization methods that exchange different forms of model parameter updates such as gradients [14]–[17]. However, there is only little work on FL of non-parametric models such as decision trees. The adaption of specific decision tree algorithms to a FL setting is discussed in [12, Ch. 2].

The closest to our work is a recent study of a model agnostic FL method that uses knowledge distillation to couple the training of (arbitrary) local models [18]. Similar to this knowledge distillation approach also we use the predictions of local models on the same dataset to construct a regularization term. However, in contrast to [18] we exploit the network structure of decentralized data to construct the regularization term.

Contribution. To the best of our knowledge, we present the first model agnostic FL method for decentralized data with an intrinsic network structure. Our method copes with arbitrary collections of local models for which efficient implementations are available. Examples for such implementations can be found in Python libraries such as `scikit-learn`, `Keras` or `PyTorch` [19]–[21]. The proposed method couples the training of well-connected local models (forming a cluster) via enforcing them to deliver similar predictions for a pre-specified test set.

Outline. Section II formulates the problem of FL from decentralized data. Section III presents a model-agnostic FL method that trains heterogeneous networks of (local) ML models in a distributed fashion.

II. Problem Formulation

Section II-A introduces the empirical graph as a useful representation of collections of local datasets along with their similarities. Section II-B augments the empirical graph by assigning a separate local hypothesis space (or model) to each node. Section III presents our model agnostic FL method for coupling the training of local models by regularization. The regularization will be implemented by enforcing a small variation of local models at well-connected nodes (clusters). Section II-C introduces the generalized total variation (GTV) as quantitative measure for the variation of heterogeneous networks of ML models. large training set to train each local model.

This work has been supported by the Academy of Finland via funding granted under decision numbers 331197 and 331197.

A. The Empirical Graph

We represent decentralized data, i.e., collections of local datasets $\mathcal{D}^{(i)}$, for $i = \{1,\ldots,n\}$, using an empirical graph $\mathcal{G} := \big(\mathcal{V},\mathcal{E}\big)$ with nodes (vertices) $\mathcal{V} = \{1,\ldots,n\}$. The empirical graph of decentralized data is an undirected weighted graph $\mathcal{G} = (\mathcal{V},\mathcal{E})$ whose nodes $\mathcal{V} := \{1,\ldots,n\}$ carry the local datasets $\mathcal{D}^{(i)}$, for $i \in \mathcal{V}$. Each node $i \in \mathcal{V}$ of the empirical graph $\mathcal{G}$ carries the local dataset

$$\mathcal{D}^{(i)} := \Big\{ \big(\mathbf{x}^{(i,1)}, y^{(i,1)}\big), \ldots, \big(\mathbf{x}^{(i,m_i)}, y^{(i,m_i)}\big) \Big\}. \quad (1)$$

Here, $\mathbf{x}^{(i,r)}$ and $y^{(i,r)}$ denote, respectively, the feature vector and true label of the rth data point in the local dataset $\mathcal{D}^{(i)}$. Note that the size m_i of the local dataset might vary between different nodes $i \in \mathcal{V}$.

An undirected edge $\{i,i'\} \in \mathcal{E}$ in the empirical graph indicates that the local datasets $\mathcal{D}^{(i)}$ and $\mathcal{D}^{(i')}$ have similar statistical properties. We quantify the level of similarity by a positive edge weight $A_{i,i'} > 0$.[1] The neighbourhood of a node $i \in \mathcal{V}$ is $\mathcal{N}^{(i)} := \{i' \in \mathcal{V} : \{i,i'\} \in \mathcal{E}\}$.

Note that the undirected edges $\{i,i'\}$ of an empirical graph encode a symmetric notion of similarity between local datasets. If the local dataset $\mathcal{D}^{(i)}$ at node i is (statistically) similar to the local dataset $\mathcal{D}^{(i')}$ at node i', then also the local dataset $\mathcal{D}^{(i')}$ is (statistically) similar to the local dataset $\mathcal{D}^{(i)}$.

The empirical graph of networked data is a design choice which is guided by computational aspects and statistical aspects of the resulting ML method. For example, using an empirical graph with a relatively small number of edges ("sparse graphs") typically results in a smaller computational complexity. Indeed, the amount of computation required by the FL methods developed in Section III is proportional to the number of edges in the empirical graph. On the other hand, the empirical graph should contain sufficient number of edges between nodes that carry statistically similar local datasets. This allows GTV minimization techniques to adaptively pool local datasets into clusters of (approximately) homogeneous data.

B. Networked Models

Consider networked data with empirical graph $\mathcal{G}$ whose nodes $i \in \mathcal{V}$ carry local datasets $\mathcal{D}^{(i)}$. For each node $i \in \mathcal{V}$, we wish to learn a useful hypothesis $\widehat{h}^{(i)}$ from a local hypothesis space $\mathcal{H}^{(i)}$. The learnt hypothesis should incur a small average loss over a local dataset $\mathcal{D}^{(i)}$,

$$L_i\left(\widehat{h}^{(i)}\right) := (1/m_i) \sum_{r=1}^{m_i} L\big(\big(\mathbf{x}^{(i,r)}, y^{(i,r)}\big), \widehat{h}^{(i)}\big). \quad (2)$$

A collection of local models $\mathcal{H}^{(i)}$, for each $i \in \mathcal{V}$, defines a networked model $\mathcal{H}^{(\mathcal{G})}$ over the empirical graph $\mathcal{G}$,

$$\mathcal{H}^{(\mathcal{G})} : i \mapsto \mathcal{H}^{(i)} \text{ for each node } i \in \mathcal{V}. \quad (3)$$

[1]The notion of statistical similarity could be made precise using a probabilistic model that interprets the data points in each local dataset $\mathcal{D}^{(i)}$ as independent and identically distributed (i.i.d.) draws from an underlying probability distribution $p^{(i)}\big(\mathbf{x},y\big)$.

A networked model is constituted by networked hypothesis maps $h \in \mathcal{H}^{(\mathcal{G})}$. Each such networked hypothesis map assigns each node $i \in \mathcal{V}$ a local hypothesis,

$$h : i \mapsto h^{(i)} \in \mathcal{H}^{(i)}. \quad (4)$$

It is important to note a networked model may combine different types of local models $\mathcal{H}^{(i)}$. For example, $\mathcal{H}^{(i)}$ might be a linear model $\mathcal{H}^{(d)}$, while $\mathcal{H}^{(i')}$ might be a decision tree for some other node $i' \neq i$. The only restriction we place on the choice for local models is the availability of computational means ("a .fit() function") to train them via regularized empirical risk minimization.

C. Generalized Total Variation

In principle, we could train each local model $\mathcal{H}^{(i)}$ separately on the corresponding local dataset $\mathcal{D}^{(i)}$ for each node $i \in \mathcal{V}$. However, the local datasets might be too small to train a local model which might be a deep neural net or a linear model using a large number of features. As a remedy, we could try to pool local datasets if they have similar statistical properties to obtain a sufficiently large dataset to train the local models $\mathcal{H}^{(i)}$.

We use the network structure of the empirical graph $\mathcal{G}$ to adaptively pool local datasets with similar statistical properties. This pooling will be implemented by requiring local models at well-connected nodes (clusters) to behave similar on a common test set. To make this informal idea more precise we next introduce a quantity measure for the variation of local models across the edges in $\mathcal{G}$.

Consider two nodes $i,i' \in \mathcal{G}$ in the empirical graph that are connected by an edge eii' with weight $A_{i,i'}$. We define the variation between $h^{(i)}$ and $h^{(i')}$ via the discrepancy between their predictions

$$d\big(h^{(i)},h^{(i')}\big) := (1/m') \sum_{r=1}^{m'} \Big[L\big(\big(\mathbf{x}^{(r)}, h^{(i)}\big(\mathbf{x}^{(r)}\big)\big), h^{(i')}\big) + L\big(\big(\mathbf{x}^{(r)}, h^{(i')}\big(\mathbf{x}^{(r)}\big)\big), h^{(i)}\big) \Big] \quad (5)$$

on a common test-set

$$\mathcal{D}^{(\text{test})} = \Big\{ \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m')} \Big\}. \quad (6)$$

The test set (6) must be shared with each node $i \in \mathcal{V}$ of the empirical graph.

We then define the GTV of a networked hypothesis $h \in \mathcal{H}^{(\mathcal{G})}$, consisting of local hypothesis maps $h^{(i)} \in \mathcal{H}^{(i)}$ (for each node $i \in \mathcal{V}$) by summing the discrepancy (5) over all edges $\mathcal{E}$,

$$\text{GTV}\,\{h\} := \sum_{\{i,i'\} \in \mathcal{E}} A_{i,i'} d\big(h^{(i)},h^{(i')}\big). \quad (7)$$

Note that $\text{GTV}\,\{h\}$ is parametrized by the choice for the loss function L used to compute the discrepancy $d\big(h^{(i)},h^{(i')}\big)$ (5). The loss function might be different from the local loss function (2) used to measure the prediction error of a local hypothesis $h^{(i)}$. However, it might be beneficial to use the same loss function in (2) and (5) (see Section III-A).

III. A Model Agnostic FL Method

We now present our FL method for learning a local hypothesis map $\widehat{h}^{(i)}$ for each node i of a an empirical graph $\mathcal{G}$. This method is from an instance of regularized empirical risk minimization (RERM), using GTV (7) as regularizer,

$$\min_{\{h^{(i)} \in \mathcal{H}^{(i)}\}} \sum_{i \in \mathcal{V}} L_i \left(h^{(i)} \right) + \lambda \sum_{\{i,i'\} \in \mathcal{E}} A_{i,i'} d\big(h^{(i)}, h^{(i')}\big). \quad (8)$$

We use block-coordinate minimization [22], [23] to solve GTVMin (8). To this end, we rewrite (8) as

$$\min_{h \in \mathcal{H}^{(\mathcal{G})}} \underbrace{\sum_{i \in \mathcal{V}} L_i \left(h^{(i)} \right) + (\lambda/2) \sum_{i' \in \mathcal{N}^{(i)}} A_{i,i'} d\big(h^{(i)}, h^{(i')}\big)}_{:= f(h^{(1)}, \ldots, h^{(n)})}$$

Given some local hypothesis maps $\widehat{h}^{(i)}_{k'}$, for all nodes $i' \in \mathcal{V}$, we compute (hopefully improved) updated local hypothesis maps $\widehat{h}^{(i)}_{k'+1}$ by minimizing $f(h)$ along $h^{(i)}$, keeping the other local hypothesis maps fixed,

$$\begin{aligned} \widehat{h}^{(i)}_{k+1} &\in \operatorname*{argmin}_{h^{(i)} \in \mathcal{H}^{(i)}} f\left(\widehat{h}^{(1)}_k, \ldots, \widehat{h}^{(i-1)}_k, h^{(i)}, \widehat{h}^{(i+1)}_k, \ldots \right) \\ &\overset{(9)}{=} \operatorname*{argmin}_{h^{(i)} \in \mathcal{H}^{(i)}} L_i \left(h^{(i)} \right) + (\lambda/2) \sum_{i' \in \mathcal{N}^{(i)}} A_{i,i'} d\big(h^{(i)}, \widehat{h}^{(i')}_k\big). \quad (9) \end{aligned}$$

We obtain Algorithm 1 by iterating (9) (simultaneously at all nodes $i \in \mathcal{V}$) until a stopping criterion is met. The main computational work of Algorithm 1 is done in step (4). This step is an instance of RERM for the local model $\mathcal{H}^{(i)}$ at each node $i \in \mathcal{V}$. The regularization term for this RERM instance is a weighted sum of the discrepancies (5) between the predictions (for the labels on the test set (6)) of the local hypothesis map $h^{(i)}$ and the predictions of the current local hypothesis maps $h^{(i)}$ at neighbouring nodes $i' \in \mathcal{N}^{(i)}$.

Algorithm 1 FedRelax

Input: empirical graph $\mathcal{G}$ with edge weights $A_{i,i'}$; local loss functions $L_i(\cdot)$; test-set $\mathcal{D}' = \left\{ \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m')} \right\}$; GTV parameter λ; loss function L for computing the discrepancy $d^{(i,i')}_h$

Initialize: $k := 0$; $\widehat{h}^{(i)}_0 \equiv 0$ for all nodes $i \in \mathcal{V}$

1: **while** stopping criterion is not satisfied **do**
2: **for** all nodes $i \in \mathcal{V}$ in parallel **do**
3: share predictions $\left\{ \widehat{h}^{(i)}_k(\mathbf{x}) \right\}_{\mathbf{x} \in \mathcal{D}^{(\text{test})}}$, with neighbours $i' \in \mathcal{N}^{(i)}$
4: update hypothesis $\widehat{h}^{(i)}_k$ as follows:

$$\widehat{h}^{(i)}_{k+1} \in \operatorname*{argmin}_{h^{(i)} \in \mathcal{H}^{(i)}} L_i \left(h^{(i)} \right) + (\lambda/2) \sum_{i' \in \mathcal{N}^{(i)}} A_{i,i'} d\big(h^{(i)}, \widehat{h}^{(i')}_k\big). \quad (10)$$

5: **end for**
6: $k := k+1$
7: **end while**

A. Model Agnostic Federated Regression

Note that Algorithm 1 is parametrized by the choices for the loss function used to measure the training error (2) of a local hypothesis $\hat{h}i$ and the loss function used to measure the discrepancy (5) between the local models at connected nodes.

A popular choice for the loss function in regression problems, i.e., data points having an numeric label, is the squared error loss

$$L\big((\mathbf{x}, y), h\big) := \big(y - \underbrace{h(\mathbf{x})}_{=\hat{y}}\big)^2. \quad (11)$$

We obtain Algorithm 2 as the special case of Algorithm 1 when using the squared error loss in (2) and (5).

Algorithm 2 FedRelax Least-Squares Regression

Input: empirical graph $\mathcal{G}$ with edge weights $A_{i,i'}$; local loss functions $L_i(\cdot)$; test-set $\mathcal{D}' = \left\{ \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m')} \right\}$; GTV parameter λ

Initialize: $k := 0$; $\widehat{h}^{(i)}_0 \equiv 0$ for all nodes $i \in \mathcal{V}$

1: **while** stopping criterion is not satisfied **do**
2: **for** all nodes $i \in \mathcal{V}$ in parallel **do**
3: share test-set labels $\left\{ \widehat{h}^{(i)}_k(\mathbf{x}) \right\}_{\mathbf{x} \in \mathcal{D}^{(\text{test})}}$, with neighbours $i' \in \mathcal{N}^{(i)}$
4: update hypothesis $\widehat{h}^{(i)}_k$ as follows:

$$\widehat{h}^{(i)}_{k+1} \in \operatorname*{argmin}_{h^{(i)} \in \mathcal{H}^{(i)}} \Bigg[L_i \left(h^{(i)} \right) + (\lambda/(2m')) \sum_{i' \in \mathcal{N}^{(i)}} A_{i,i'} \sum_{r=1}^{m'} \left(h^{(i)}\big(\mathbf{x}^{(r)}\big) - \widehat{h}^{(i')}_k\big(\mathbf{x}^{(r)}\big) \right)^2 \Bigg]. \quad (12)$$

5: **end for**
6: $k := k+1$
7: **end while**

Note that the update (12) is nothing but regularized empirical risk minimization (ERM) for learning a local hypothesis $h^{(i)} \in \mathcal{H}^{(i)}$ from the local dataset $\mathcal{D}^{(i)}$. The regularization term in (12) is the average squared error loss incurred on the ("pseudo-") labeled test set (see (6))

$$\bigcup_{i' \in \mathcal{N}^{(i)}} \left\{ \big(\mathbf{x}^{(1)}, \widehat{h}^{(i')}_k\big(\mathbf{x}^{(1)}\big)\big), \ldots, \big(\mathbf{x}^{(m')}, \widehat{h}^{(i')}_k\big(\mathbf{x}^{(m')}\big)\big) \right\}. \quad (13)$$

References

[1] S. Cui, A. Hero, Z.-Q. Luo, and J.M.F. Moura, Eds., *Big Data over Networks*, Cambridge Univ. Press, 2016.

[2] M. E. J. Newman, *Networks: An Introduction*, Oxford Univ. Press, 2010.

[3] Ljubiša Stanković, Danilo Mandic, Miloš Daković, Miloš Brajović, Bruno Scalzo, Shengxi Li, and Anthony G. Constantinides, "Data analytics on graphs part iii: Machine learning on graphs, from graph topology to applications," *Foundations and Trends® in Machine Learning*, vol. 13, no. 4, pp. 332–530, 2020.

[4] O. Chapelle, B. Schölkopf, and A. Zien, Eds., *Semi-Supervised Learning*, The MIT Press, Cambridge, Massachusetts, 2006.

[5] Ljubiša Stanković, Danilo Mandic, Miloš Daković, Miloš Brajović, Bruno Scalzo, Shengxi Li, and Anthony G. Constantinides, "Data analytics on graphs part i: Graphs and spectra on graphs," *Foundations and Trends® in Machine Learning*, vol. 13, no. 1, pp. 1–157, 2020.

[6] Ljubiša Stanković, Danilo Mandic, Miloš Daković, Miloš Brajović, Bruno Scalzo, Shengxi Li, and Anthony G. Constantinides, "Data analytics on graphs part ii: Signals on graphs," *Foundations and Trends® in Machine Learning*, vol. 13, no. 2-3, pp. 158–331, 2020.

[7] A. Jung, A O. Hero, A. Mara, S. Jahromi, A. Heimowitz, and Y.C. Eldar, "Semi-supervised learning in network-structured data via total variation minimization," *IEEE Trans. Signal Processing*, vol. 67, no. 24, Dec. 2019.

[8] K. Sunil P. Frossard D.I. Shuman, S.K. Narang, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83–98, 2013.

[9] A. Jung, "Networked exponential families for big data over networks," *IEEE Access*, vol. 8, pp. 202897–202909, 2020.

[10] J. Kovačević Y. Chi R. Varma, H. Lee, "Vector-valued graph trend filtering with non-convex penalties," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 6, pp. 48–62, 2020.

[11] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Aguera y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, Aarti Singh and Jerry Zhu, Eds., Fort Lauderdale, FL, USA, 20–22 Apr 2017, vol. 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282, PMLR.

[12] H. Ludwig and N. Baracaldo, Eds., *Federated Learning: A Comprehensive Overview of Methods and Applications*, Springer, 2022.

[13] Yong Cheng Yan Kang Tianjian Chen Han Yu Qiang Yang, Yang Liu, *Federated Learning*, Springer, 1 edition, 2022.

[14] J. Liu and C. Zhang, "Distributed learning systems with first-order methods," *Foundations and Trends in Databases*, vol. 9, no. 1, pp. 100.

[15] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc' aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, Quoc Le, and Andrew Ng, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, Eds. 2012, vol. 25, Curran Associates, Inc.

[16] J. Tsitsiklis, D. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Transactions on Automatic Control*, vol. 31, no. 9, pp. 803–812, 1986.

[17] Kevin Scaman, Francis Bach, Sebastien Bubeck, Laurent Massoulié, and Yin Tat Lee, "Optimal algorithms for non-smooth distributed optimization in networks," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. 2018, vol. 31, Curran Associates, Inc.

[18] A. Afonin and S.P. Karimireddy, "Towards model-agnostic federated learning using knowledge distillation," in *International Conference on Learning Representations*, 2022.

[19] F. Pedregosa, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825–2830, 2011.

[20] Haifeng Jin, François Chollet, Qingquan Song, and Xia Hu, "Autokeras: An automl library for deep learning," *Journal of Machine Learning Research*, vol. 24, no. 6, pp. 1–6, 2023.

[21] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. 2019, vol. 32, Curran Associates, Inc.

[22] D. P. Bertsekas and J. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Athena Scientific, 2015.

[23] D. P. Bertsekas, *Convex Optimization Algorithms*, Athena Scientific, 2015.