FULLY DECENTRALIZED JOINT LEARNING OF PERSONALIZED MODELS AND COLLABORATION GRAPHS

Aurélien Bellet (Inria)

Includes work with:
M. Tommasi, P. Vanhaesebrouck (University of Lille & Inria)
R. Guerraoui, M. Taziki (EPFL)
V. Zantedeschi (University of Saint-Etienne)

Applied Machine Learning Days 2020 — AI & ML on the Edge
EPFL, January 27, 2020

CONNECTED DEVICES: PERVASIVE OR INVASIVE?

- Connected devices are spreading rapidly and collect increasingly personal data
  - Ex: browsing logs, health, speech, accelerometer, geolocation...
- Opportunity to provide personalized services but also a potential threat to privacy
- A first step to try and reconcile the two: keep and process data on the user device

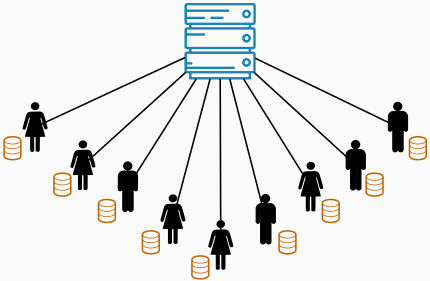
AI & ML ON THE EDGE: TWO SETTINGS OF INTEREST

- Most of previous talks: Inference on the edge
  - Pre-trained ML model pushed to user devices
  - Challenge: perform efficient and accurate on-device predictions

- This talk: Training on the edge
  - Train ML model on the data of many devices
  - Challenge: design training algorithms that scale to large number of devices

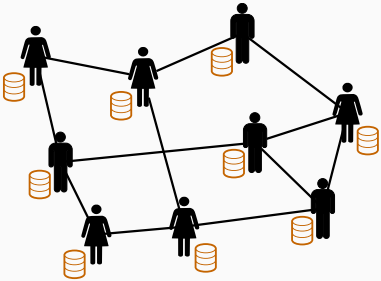
FEDERATED VS FULLY DECENTRALIZED TRAINING

Standard federated learning



- Coordination by a central server
- Single point of failure, server may become a bottleneck

Fully decentralized learning



- Device-to-device communication in a sparse network graph
- Naturally scales to many devices

See [Kairouz et al., 2019] for a detailed overview of federated/decentralized ML

GLOBAL MODEL VS PERSONALIZED MODELS

Global model

- One-size-fits-all: same model makes predictions for all devices
- Model should be trained on data from all users
- Large model may be needed to capture the specificities of each user

Personalized models

- One model per device
- Model should be trained on data from that user and from similar users
- Smaller models may be sufficient

We propose to learn personalized models in a fully decentralized setting:

- Learn “who to communicate with” by inferring a graph of similarities between users
- Collaboratively learn personalized models over this graph
- Optimize the models and the graph jointly, in an alternating fashion

PROBLEM FORMULATION

- A set of n users (devices) with common feature space $\mathcal{X}$ and label space $\mathcal{Y}$
- User i has local training dataset $\mathcal{S}_i = \{(x_i^j, y_i^j)\}_{j=1}^{m_i}$ of size $m_i \geq 0$ and wants to learn a model $\theta_i \in \mathbb{R}^p$ which generalizes well to future local data
- In isolation, user i can learn a purely local model by minimizing a local loss $\mathcal{L}_i(\theta; \mathcal{S}_i)$ (with L_i^{loc}-Lipschitz continuous gradient)
- This will generalize poorly when local data is scarce $\rightarrow$ need to collaborate

- **Asynchronous time model**: each user becomes active at random times, asynchronously and in parallel (we use global counter t to denote the t-th activation)
- **Communication model**: all users can exchange messages, but we want to restrict communication to pairs of most similar users
- We model this by a collaboration graph: a weighted graph with edge weight $w_{ij} \geq 0$ reflecting similarity between the learning tasks of users i and j

- Learn personalized models $\Theta \in \mathbb{R}^{n \times p}$ and graph weights $w \in \mathbb{R}_{\geq 0}^{n(n-1)/2}$ as solutions to

$$\min_{\substack{\Theta \in \mathbb{R}^{n \times p} \\ w \in \mathbb{R}_{\geq 0}^{n(n-1)/2}}} J(\Theta, w) = \sum_{i=1}^{n} d_i c_i \mathcal{L}_i(\theta_i; \mathcal{S}_i) + \frac{\mu}{2} \sum_{i<j} w_{ij} \|\theta_i - \theta_j\|^2 + \lambda g(w),$$

- Trade-off between accurate models on local data and smooth models over the graph
- $c_i \in (0, 1] \propto m_i$: confidence of user i, $d_i = \sum_{j \neq i} w_{ij}$: degree of i
- Term $g(w)$: avoid trivial collaboration graph, encourage sparsity
- Flexible relationships: hyperparameter $\mu \geq 0$ interpolates between learning purely local models and a shared model per connected component

OUTLINE OF THE PROPOSED ALGORITHM

We design an alternating optimization procedure over Θ and w:

1. A decentralized algorithm to learn the models given the graph
2. A decentralized algorithm to learn a graph given the models

LEARNING MODELS GIVEN THE GRAPH

- Denote neighborhood of user i by $N_i = \{j : w_{ij} > 0\}$
- Initialize models $\Theta(0) \in \mathbb{R}^{n \times p}$
- At step $t \geq 0$, a random user i becomes active:
  1. user i updates its model based on its local dataset $\mathcal{S}_i$ and the information from neighbors:

$$\theta_i(t+1) = \theta_i(t) - \frac{1}{\mu + c_i L_i^{loc}} \left(c_i \nabla \mathcal{L}_i(\theta_i(t); \mathcal{S}_i) - \mu \sum_{j \in N_i} \frac{w_{ij}}{d_i} \theta_j(t) \right)$$

  2. user i sends its updated model $\theta_i(t+1)$ to its neighborhood N_i
- The update is a combination of a local gradient step and a weighted average of neighbors' models

Proposition ([Bellet et al., 2018])

For any $T > 0$, let $(\Theta(t))_{t=1}^{T}$ be the sequence of iterates generated by the algorithm running for T iterations from an initial point $\Theta(0)$. When the local losses $\mathcal{L}_i$ are strongly convex, we have:

$$\mathbb{E}\left[f(\Theta(T)) - f^*\right] \leq \left(1 - \frac{\sigma}{nL_{max}}\right)^T \left(f(\Theta(0)) - f^*\right).$$

where $L_{max} = \max_i L_i$ and σ are smoothness and strong convexity parameters.

- Optimality gap decreases exponentially fast with T
- Constant number of per-user updates $\rightarrow$ optimality gap roughly constant in n
- Note: can prove $O(1/T)$ convergence for the standard convex case

LEARNING THE GRAPH GIVEN MODELS

$$\min_{\substack{\Theta \in \mathbb{R}^{n \times p} \\ w \in \mathbb{R}^{n(n-1)/2}_{\geq 0}}} J(\Theta, w) = \sum_{i=1}^{n} d_i c_i \mathcal{L}_i(\theta_i; \mathcal{S}_i) + \frac{\mu}{2} \sum_{i<j} w_{ij} \|\theta_i - \theta_j\|^2 + \lambda g(w)$$

- Our algorithm can deal with a large family of functions g
- Inspired by [Kalofolias, 2016], we can define

$$g(w) = \beta \|w\|^2 - 1^T \log(d + \delta) \quad \text{(with } \delta \text{ small constant)}$$

- Log barrier on the degree vector d to avoid isolated users and L_2 penalty on weights to control the graph sparsity
- The resulting objective h in w is strongly convex

- We rely on decentralized peer sampling [Jelasity et al., 2007] to allow users to communicate with a set of κ random peers
- Initialize weights $w(0)$, choose parameter $\kappa \in \{1, \dots, n-1\}$
- At each step $t \geq 0$, a random user i becomes active:
  1. Draw a set $\mathcal{K}$ of κ users and request their model, loss and degree
  2. Update the associated weights $w(t+1)_{i,\mathcal{K}}$ via a gradient update
  3. Send each updated weight $w(t+1)_{ij}$ to the associated user $j \in \mathcal{K}$

Theorem ([Zantedeschi et al., 2020])

For any $T > 0$, let $(w(t))_{t=1}^{T}$ be the sequence of iterates generated by the algorithm running for T iterations from an initial point $w(0)$. We have:

$$\mathbb{E}[h(w^{(T)}) - h^*] \leq \rho^T (h(w^{(0)}) - h^*), \quad \text{where } \rho = 1 - \frac{4}{n(n-1)} \frac{\kappa\beta\delta^2}{\kappa + 1 + 2\beta\delta^2}$$

- κ can be used to trade-off between communication cost and convergence speed

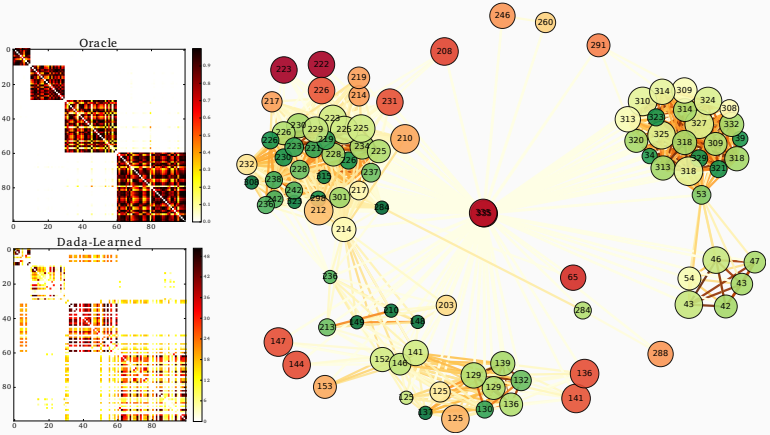
Extensions (not covered in this talk)

- Low-communication updates via greedy boosting [Zantedeschi et al., 2020]
- Algorithm with formal differential privacy guarantees [Bellet et al., 2018]

NUMERICAL EXPERIMENTS

EXPERIMENTS: SYNTHETIC DATA

- We approximately recover the ground-truth cluster structure
- Prediction accuracy is close to that of the oracle graph



EXPERIMENTS: REAL DATASETS

- Real datasets that are naturally collected at the user/device level
- Number of users n from 23 to 190, no task similarity available
- Linear models and nonlinear ensembles
- Our approach clearly outperforms both global and purely local models

Dataset	Global-lin	Local-lin	**Ours-lin**	Global-nonlin	Local-nonlin	**Ours-nonlin**
HARWS	93.64	92.69	**96.31**	94.34	93.16	95.70
VEHICLE	87.11	90.38	**91.37**	88.02	90.59	90.81
COMPUTER	62.18	60.68	69.08	69.16	66.61	**72.09**
SCHOOL	57.06	70.43	71.92	69.16	66.61	**72.22**

bold blue = best, regular blue = second best

Thank you for your attention!
Questions?

REFERENCES I

[Bellet et al., 2018] Bellet, A., Guerraoui, R., Taziki, M., and Tommasi, M. (2018).
Personalized and Private Peer-to-Peer Machine Learning.
In *AISTATS*.

[Jelasity et al., 2007] Jelasity, M., Voulgaris, S., Guerraoui, R., Kermarrec, A.-M., and van Steen, M. (2007).
Gossip-based peer sampling.
ACM Trans. Comput. Syst., 25(3).

[Kairouz et al., 2019] Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Rouayheb, S. E., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P. B., Gruteser, M., Harchaoui, Z., He, C., He, L., Huo, Z., Hutchinson, B., Hsu, J., Jaggi, M., Javidi, T., Joshi, G., Khodak, M., Konečný, J., Korolova, A., Koushanfar, F., Koyejo, S., Lepoint, T., Liu, Y., Mittal, P., Mohri, M., Nock, R., Özgür, A., Pagh, R., Raykova, M., Qi, H., Ramage, D., Raskar, R., Song, D., Song, W., Stich, S. U., Sun, Z., Suresh, A. T., Tramèr, F., Vepakomma, P., Wang, J., Xiong, L., Xu, Z., Yang, Q., Yu, F. X., Yu, H., and Zhao, S. (2019).
Advances and Open Problems in Federated Learning.
Technical report, arXiv:1912.04977.

[Kalofolias, 2016] Kalofolias, V. (2016).
How to learn a graph from smooth signals.
In *AISTATS*.

References II

[Vanhaesebrouck et al., 2017] Vanhaesebrouck, P., Bellet, A., and Tommasi, M. (2017).
Decentralized Collaborative Learning of Personalized Models over Networks.
In *AISTATS*.

[Zantedeschi et al., 2020] Zantedeschi, V., Bellet, A., and Tommasi, M. (2020).
Fully decentralized joint learning of personalized models and collaboration graphs.
In *AISTATS*.

- Here we use the oracle graph