Personalized and Private Peer-to-Peer Machine Learning

(novembre 2018)

Pierre-Louis Biojout, Leonid Kniazev

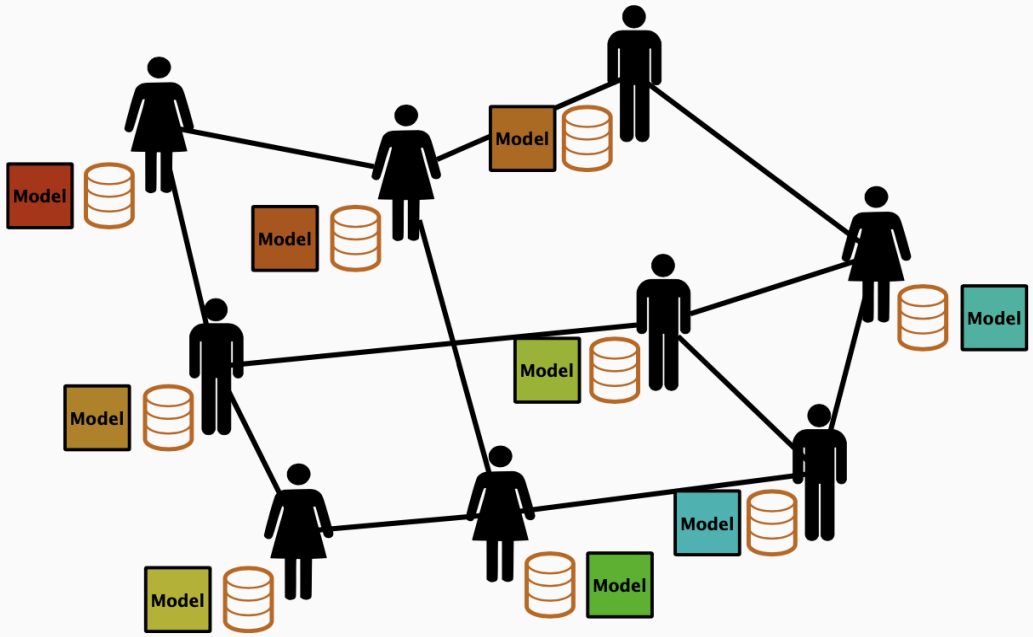
EA MAP 578

Learning from connected devices data

- Connected devices are widespread and collect increasingly large and sensitive user data
- Ex: browsing logs, health data, accelerometer, geolocation...
- Great opportunity for providing personalized services but raises
- serious privacy concerns
- Centralize data from all devices: best for utility, bad for privacy
- Learn on each device separately: best for privacy, bad for utility

Proposed solution: fully decentralized network

- Personal data stays on user's device
- Peer-to-peer and asynchronous communication
- No single point of failure/entry as in server-client architecture
- Scalability-by-design to many devices through local updates (see e.g. NIPS 2017 paper [Lian et al., 2017])
- Learn a personalized model for each user (multi-task learning)
- General idea: trade-off between model accuracy on local data and smoothness with respect to similar users



Problem setting

Problem setting

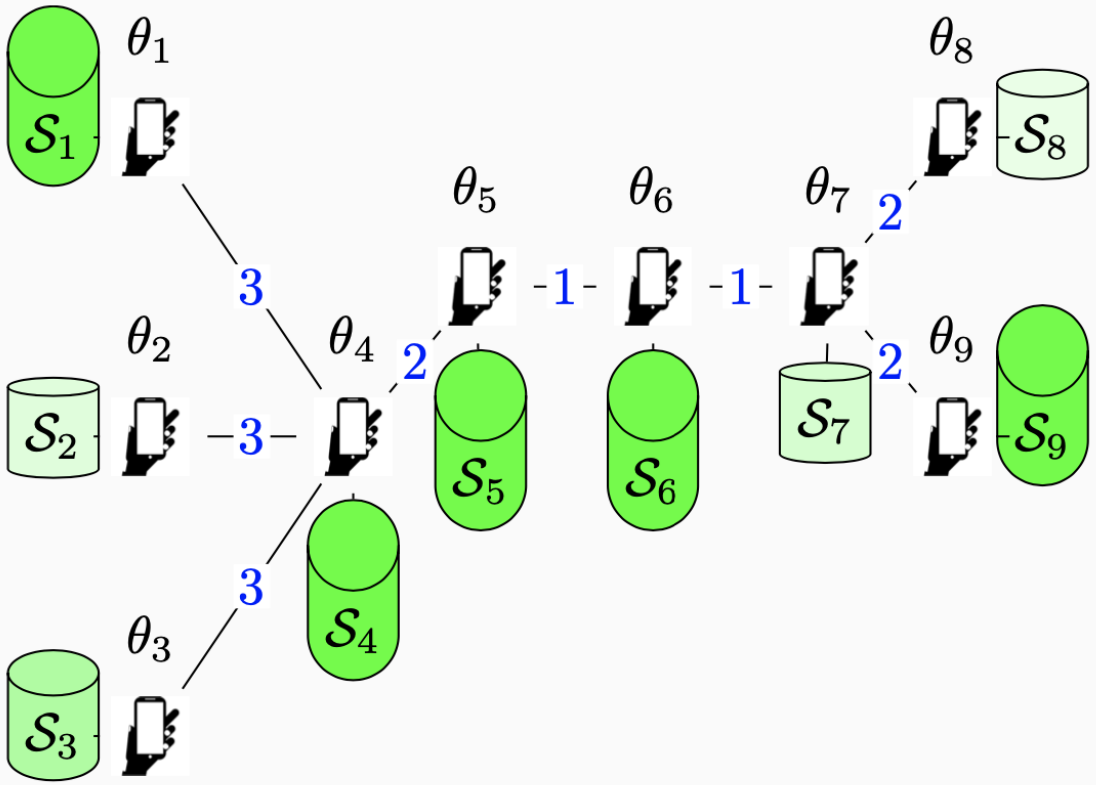
- A set $V = [\![n]\!] = \{1, \ldots, n\}$ of n learning agents
- A convex loss function $\ell : \mathbb{R}^p \times \mathcal{X} \times \mathcal{Y}$
- Personalized and imbalanced data: agent i has dataset $\mathcal{S}_i = \{(x_i^j, y_i^j)\}_{j=1}^{m_i}$ of size $m_i \geq 0$ drawn from μ_i
- Purely local model: agent i can learn a model θ_i on its own by minimizing the loss on its local data

$$\mathcal{L}_i(\theta) = \frac{1}{m_i} \sum_{j=1}^{m_i} \ell(\theta; x_i^j, y_i^j) + \lambda_i \|\theta\|^2, \text{ with } \lambda_i \geq 0$$

Problem setting

- Network: weighted connected graph $G = (V, E)$
- $E \subseteq V \times V$ set of undirected edges
- Weight matrix $W \in \mathbb{R}^{n \times n}$: symmetric, nonnegative, with $W_{ij} = 0$ if $(i, j) \notin E$ or $i = j$
- **Assumption**: network weights are given and represent the underlying similarity between agents

Problem setting



- Agents have only a local view of the network: they only know their neighborhood $\mathcal{N}_i = \{j \neq i : W_{ij} > 0\}$ and associated weights

Problem formulation

- Denoting $\Theta = [\Theta_1; \ldots; \Theta_n] \in \mathbb{R}^{np}$, we use a graph regularization formulation [Evgeniou and Pontil, 2004, Vanhaesebrouck et al., 2017]:

$$\min_{\Theta \in \mathbb{R}^{np}} \mathcal{Q}_{\mathcal{L}}(\Theta) = \frac{1}{2} \sum_{i<j}^{n} W_{ij} \|\Theta_i - \Theta_j\|^2 + \mu \sum_{i=1}^{n} D_{ii} c_i \mathcal{L}_i(\Theta_i; \mathcal{S}_i)$$

  - $\mu > 0$ trade-off parameter, $D_{ii} = \sum_j W_{ij}$ normalization factor
  - $c_i \in (0, 1] \propto m_i$ is the "confidence" of agent i

- Implements a trade-off between having similar models for strongly connected agents and models that are accurate on their respective local datasets

Non-private decentralized algorithm

Non-private decentralized algorithm

- **Time and communication models**:
  - Asynchronous time: each agent has a random local clock and wakes up when it ticks
  - Broadcast communication: agents send messages to all their neighbors at once (without expecting a reply)

- **Algorithm**: assume agent i wakes up at step t
  1. Agent i updates its model based on information from neighbors:

$$\Theta_i(t+1) = (1-\alpha)\Theta_i(t) + \alpha\Big(\sum_{j\in\mathcal{N}_i} \frac{W_{ij}}{D_{ii}}\Theta_j(t) - \mu c_i \nabla\mathcal{L}_i(\Theta_i(t); \mathcal{S}_i)\Big)$$

  2. Agent i sends its updated model $\Theta_i(t+1)$ to its neighborhood $\mathcal{N}_i$

Private algorithm

Private algorithm

- In some applications, data may be sensitive and agents may not want to reveal it to anyone else
- In our algorithms, the agents never communicate their local data but exchange sequences of models computed from data
- Consider an adversary observing all the information sent over the network (but not the internal memory of agents)
- **Goal:** how can we guarantee that no/little information about the local dataset is leaked by the algorithm?

Differential privacy

(ϵ, δ)-Differential Privacy

Let $\mathcal{M}$ be a randomized mechanism taking a dataset as input, and let $\epsilon > 0, \delta \geq 0$. We say that $\mathcal{M}$ is (ϵ, δ)-differentially private if for all datasets $\mathcal{S}, \mathcal{S}'$ differing in a single data point and for all sets of possible outputs $\mathcal{O} \subseteq \text{range}(\mathcal{M})$, we have:

$$Pr(\mathcal{M}(\mathcal{S}) \in \mathcal{O}) \leq e^{\epsilon} Pr(\mathcal{M}(\mathcal{S}') \in \mathcal{O}) + \delta.$$

- Guarantees that $\mathcal{M}$ does not leak much information about any individual data point
- Information-theoretic (no computational assumptions)

Differentially private algorithm

- **Differentially-private algorithm**:
  1. Replace the update of the algorithm by

$$\widetilde{\Theta}_i(t+1) = (1-\alpha)\widetilde{\Theta}_i(t) + \alpha\left(\sum_{j\in\mathcal{N}_i} \frac{W_{ij}}{D_{ii}}\widetilde{\Theta}_j(t) - \mu c_i(\nabla\mathcal{L}_i(\widetilde{\Theta}_i(t);\mathcal{S}_i) + \eta_i(t))\right),$$

     where $\eta_i(t) \sim Laplace(0, s_i(t))^p \in \mathbb{R}^p$
  2. Agent i then broadcasts noisy iterate $\widetilde{\Theta}_i(t+1)$ to its neighbors

Privacy guarantee

Theorem ([Bellet et al., 2017])

Let $i \in [\![n]\!]$ and assume

- *$\ell(\cdot; x, y)$ L_0-Lipschitz w.r.t. the L_1-norm for all (x, y)*
- *Agent i wakes up on iterations $t_i^1, \dots, t_i^{T_i}$*
- *For some $\epsilon_i(t_i^k) > 0$, the noise scale is $s_i(t_i^k) = \frac{2L_0}{\epsilon_i(t_i^k) m_i}$*

Then for any initial point $\widetilde{\Theta}(0)$ independent of $\mathcal{S}_i$, the mechanism $\mathcal{M}_i(\mathcal{S}_i)$ is $(\bar{\epsilon}_i, 0)$-DP with $\bar{\epsilon}_i = \sum_{k=1}^{T_i} \epsilon_i(t_i^k)$.

- Sweet spot: the less data, the more noise added by the agent, but the least influence in the network

Privacy/utility trade-off

Theorem ([Bellet et al., 2017])

For any $T > 0$, let $(\widetilde{\Theta}(t))_{t=1}^{T}$ be the sequence of iterates generated by T iterations. Under appropriate assumptions, we have:

$$\mathbb{E}\left[\mathcal{Q}_{CL}(\Theta(T)) - \mathcal{Q}_{CL}^{\star}\right] \leq (1-\rho)^{T}\left(\mathcal{Q}_{CL}(\Theta(0)) - \mathcal{Q}_{CL}^{*}\right) + \textit{additive error}.$$

- Second term gives additive error due to noise
- More results in the paper

Experiments

Experiments

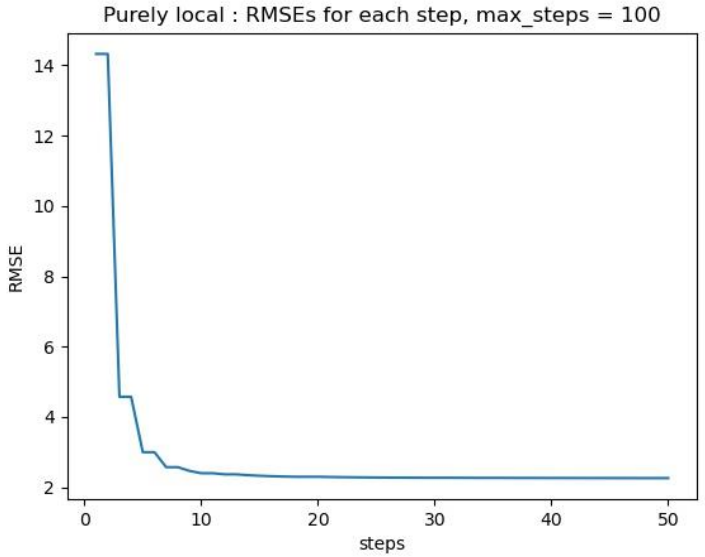
- Implementation in Python
- Simulation of the network on a single computer
- Use of ml-100k dataset
- The network is obtained by setting $W_{ij} = 1$ if agent i is within the 10-nearest neighbors of agent j

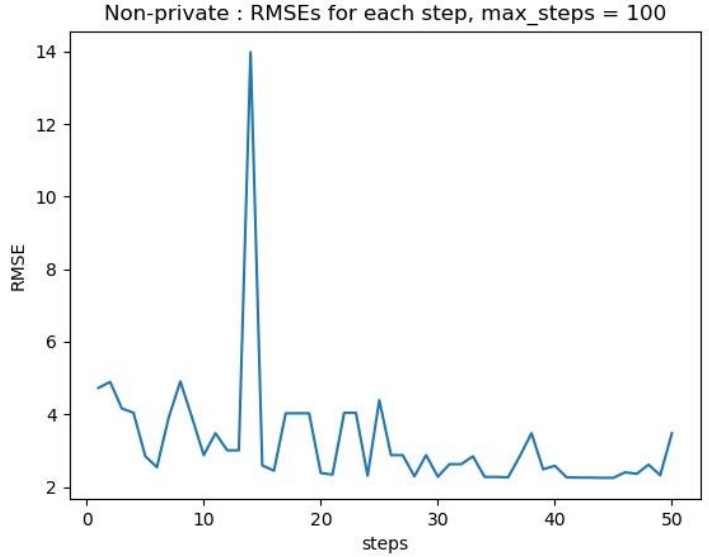
Results

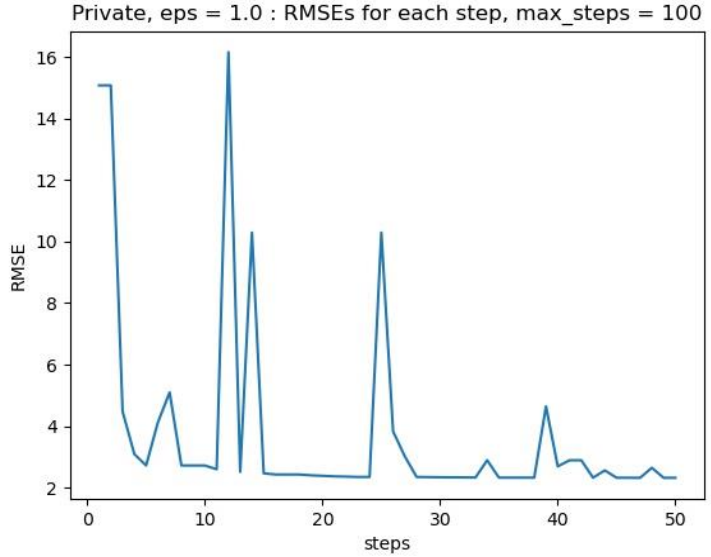
	Purely local models	Non-priv. CD	Priv. $\bar{\varepsilon} = 1$	Priv. $\bar{\varepsilon} = 0.5$	Priv. $\bar{\varepsilon} = 0.1$
Published results	1.2834	0.9502	0.9527	0.9545	0.9855
Our results	2.5821	2.5676	2.6524	2.4766	2.9688

Per-user test RMSE (averaged over users and 5 random runs) on MovieLens-100K.

Graphs


Purely local : RMSEs for each step, max_steps = 100
RMSE
steps
14
12
10
8
6
4
2
0
10
20
30
40
50


Non-private : RMSEs for each step, max_steps = 100
RMSE
steps
14
12
10
8
6
4
2
0
10
20
30
40
50


Private, eps = 1.0 : RMSEs for each step, max_steps = 100
RMSE
steps
16
14
12
10
8
6
4
2
0
10
20
30
40
50

Conclusion

Acknowledgement

- Based on the paper *Personalized and Private Peer-to-Peer Machine Learning* (Aurélien Bellet and al.)
- Part of the illustrations and formulas in our slides are from the slides *personalized and private peer-to-peer machine learning* (NIPS 2017 workshop on “Machine Learning on the Phone and other Consumer Devices” Long Beach, December 9, 2017)