DECENTRALIZED COLLABORATIVE LEARNING OF PERSONALIZED MODELS OVER NETWORKS

Aurélien Bellet (Inria MAGNET)
Joint work with Paul Vanhaesebrouck and Marc Tommasi

1. Broad context
2. Problem setting
3. Model propagation
4. Collaborative learning
5. Experiments
6. Future work
BROAD CONTEXT
We love taking photos with our phone. Can you tag them automatically?
LEARNING FROM PERSONAL DATA

- sandwich 100%
- snack food 100%
- meal 85%
- lunch 80%

Other examples of applications:
- Recommend content based on user activity logs
- Predict health risks based on medical history

4
• Other examples of applications
  • Recommend content based on user activity logs
  • Predict health risks based on medical history
• Centralized data can be processed efficiently in a data center
• Lack of user control over its personal data
  • What is collected? Who can access it? How is it used and what for?
• Vulnerability to attacks / subpoenas
  • Yahoo data breach (500M users), Twitter / Wikileaks court orders
• Costly infrastructure for service provider
• Personal data stays on user’s device → better control

• Peer-to-peer communications without a central server → harder to collect data systematically (no single point of entry)
Some scientific challenges

1. How to efficiently learn in a decentralized way under these communication constraints?
2. How to prevent malicious users from inferring sensitive data or manipulating the outcome to their advantage?
Some scientific challenges

1. How to efficiently learn in a decentralized way under these communication constraints?

2. How to prevent malicious users from inferring sensitive data or manipulating the outcome to their advantage?
• Users wake up independently and asynchronously, select a random neighbor and exchange information
  • Equivalent view: at each step, activate a random network edge
  • Simple and asynchronous → well suited to large networks
• Users wake up independently and asynchronously, select a random neighbor and exchange information
  • Equivalent view: at each step, activate a random network edge
• Simple and asynchronous → well suited to large networks
• Users wake up independently and asynchronously, select a random neighbor and exchange information
  • Equivalent view: at each step, activate a random network edge
• Simple and asynchronous → well suited to large networks
KEY PRINCIPLE: GOSSIP ALGORITHM

- Users wake up independently and asynchronously, select a random neighbor and exchange information
  - Equivalent view: at each step, activate a random network edge
- Simple and asynchronous → well suited to large networks
• Users wake up independently and asynchronously, select a random neighbor and exchange information
  • Equivalent view: at each step, activate a random network edge
• Simple and asynchronous $\rightarrow$ well suited to large networks
• Users wake up independently and asynchronously, select a random neighbor and exchange information
  • Equivalent view: at each step, activate a random network edge
• Simple and asynchronous → well suited to large networks
• Users wake up independently and asynchronously, select a random neighbor and exchange information
  • Equivalent view: at each step, activate a random network edge

• Simple and asynchronous $\rightarrow$ well suited to large networks
• Users wake up independently and asynchronously, select a random neighbor and exchange information
  • Equivalent view: at each step, activate a random network edge
• Simple and asynchronous $\rightarrow$ well suited to large networks
• Users wake up independently and asynchronously, select a random neighbor and exchange information
  • Equivalent view: at each step, activate a random network edge
• Simple and asynchronous → well suited to large networks
KEY PRINCIPLE: GOSSIP ALGORITHM

- Users wake up independently and asynchronously, select a random neighbor and exchange information
  - Equivalent view: at each step, activate a random network edge
- Simple and asynchronous $\rightarrow$ well suited to large networks
**EXISTING WORK: CONSENSUS LEARNING**

- Gossip algorithms to optimize a loss function over the union of personal datasets [Nedic and Ozdaglar, 2009, Duchi et al., 2012, Wei and Ozdaglar, 2012, Colin et al., 2016]

- General idea: at each step
  1. perform a local model update based on personal data
  2. average with neighbor
• Gossip algorithms to optimize a loss function over the union of personal datasets [Nedic and Ozdaglar, 2009, Duchi et al., 2012, Wei and Ozdaglar, 2012, Colin et al., 2016]

• General idea: at each step
  1. perform a local model update based on personal data
  2. average with neighbor
EXISTING WORK: CONSENSUS LEARNING

- Gossip algorithms to optimize a loss function over the union of personal datasets [Nedic and Ozdaglar, 2009, Duchi et al., 2012, Wei and Ozdaglar, 2012, Colin et al., 2016]

- General idea: at each step
  1. perform a local model update based on personal data
  2. average with neighbor
• Gossip algorithms to optimize a loss function over the union of personal datasets [Nedic and Ozdaglar, 2009, Duchi et al., 2012, Wei and Ozdaglar, 2012, Colin et al., 2016]

• General idea: at each step
  1. perform a local model update based on personal data
  2. average with neighbor
• Gossip algorithms to optimize a loss function over the union of personal datasets [Nedic and Ozdaglar, 2009, Duchi et al., 2012, Wei and Ozdaglar, 2012, Colin et al., 2016]

• General idea: at each step
  1. perform a local model update based on personal data
  2. average with neighbor
EXISTING WORK: CONSENSUS LEARNING

- Gossip algorithms to optimize a loss function over the union of personal datasets [Nedic and Ozdaglar, 2009, Duchi et al., 2012, Wei and Ozdaglar, 2012, Colin et al., 2016]

- General idea: at each step
  1. perform a local model update based on personal data
  2. average with neighbor
EXISTING WORK: CONSENSUS LEARNING

• Gossip algorithms to optimize a loss function over the union of personal datasets [Nedic and Ozdaglar, 2009, Duchi et al., 2012, Wei and Ozdaglar, 2012, Colin et al., 2016]

• General idea: at each step
  1. perform a local model update based on personal data
  2. average with neighbor
• Gossip algorithms to optimize a loss function over the union of personal datasets [Nedic and Ozdaglar, 2009, Duchi et al., 2012, Wei and Ozdaglar, 2012, Colin et al., 2016]

• General idea: at each step
  1. perform a local model update based on personal data
  2. average with neighbor
**EXISTING WORK: CONSENSUS LEARNING**

- Gossip algorithms to optimize a loss function over the union of personal datasets [Nedic and Ozdaglar, 2009, Duchi et al., 2012, Wei and Ozdaglar, 2012, Colin et al., 2016]

- General idea: at each step
  1. perform a local model update based on personal data
  2. average with neighbor
The image contains a diagram and text discussing existing work in consensus learning. The key points are:

- **Gossip algorithms** to optimize a loss function over the union of personal datasets [Nedic and Ozdaglar, 2009, Duchi et al., 2012, Wei and Ozdaglar, 2012, Colin et al., 2016]

- **General idea**: at each step
  1. perform a local model update based on personal data
  2. average with a neighbor
• Gossip algorithms to optimize a loss function over the union of personal datasets [Nedic and Ozdaglar, 2009, Duchi et al., 2012, Wei and Ozdaglar, 2012, Colin et al., 2016]

• General idea: at each step
  1. perform a local model update based on personal data
  2. average with neighbor
• Gossip algorithms to optimize a loss function over the union of personal datasets [Nedic and Ozdaglar, 2009, Duchi et al., 2012, Wei and Ozdaglar, 2012, Colin et al., 2016]

• General idea: at each step
  1. perform a local model update based on personal data
  2. average with neighbor
• Gossip algorithms to optimize a loss function over the union of personal datasets [Nedic and Ozdaglar, 2009, Duchi et al., 2012, Wei and Ozdaglar, 2012, Colin et al., 2016]

• General idea: at each step
  1. perform a local model update based on personal data
  2. average with neighbor
EXISTING WORK: CONSENSUS LEARNING

• Gossip algorithms to optimize a loss function over the union of personal datasets [Nedic and Ozdaglar, 2009, Duchi et al., 2012, Wei and Ozdaglar, 2012, Colin et al., 2016]

• General idea: at each step
  1. perform a local model update based on personal data
  2. average with neighbor
• Gossip algorithms to optimize a loss function over the union of personal datasets [Nedic and Ozdaglar, 2009, Duchi et al., 2012, Wei and Ozdaglar, 2012, Colin et al., 2016]

• General idea: at each step
  1. perform a local model update based on personal data
  2. average with neighbor
• Gossip algorithms to learn a personalized model for each user according to its own learning objective

• 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 \( \ell : \mathbb{R}^p \times \mathcal{X} \times \mathcal{Y} \)

• Agent \( i \) has dataset \( S_i = \{(x_i^j, y_i^j)\}_{j=1}^{m_i} \) of size \( m_i \geq 0 \) drawn i.i.d. from its own distribution \( \mu_i \) over \( \mathcal{X} \times \mathcal{Y} \)

• Goal of agent \( i \): learn a model \( \theta_i \in \mathbb{R}^p \) with small expected loss

\[
\mathbb{E}_{(x_i, y_i) \sim \mu_i} \ell(\theta_i; x_i, y_i)
\]

• In isolation, agent \( i \) can learn a “solitary” model

\[
\theta_i^{\text{sol}} \in \arg \min_{\theta \in \mathbb{R}^p} \mathcal{L}_i(\theta) = \sum_{j=1}^{m_i} \ell(\theta; x_i^j, y_i^j)
\]

• How to improve upon \( \theta_i^{\text{sol}} \) with the help of other users?
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 $i = j$

• **Simplifying assumption**: network weights are given and represent the underlying similarity between agents
  
  • Ex: movie recommendation task where the network is set up when users go to the same movie

• In a more general setup one would need to
  
  • estimate weights based on auxiliary observation or local data
  • construct a $k$-NN overlay on top of the physical communication network [Jelasity et al., 2009]
• Agents have only a **local view** of the network

• They only know their neighborhood $\mathcal{N}_i = \{j \neq i : W_{ij} > 0\}$ and the associated weights
MODEL PROPAGATION
Main idea: smooth the solitary models over the network

- $c_i \in (0, 1]$: confidence in initial model $\theta_i^{sol}$
  - Proportional to the number of training points $m_i$
- Find new set of models $\Theta \in \mathbb{R}^{n \times p}$ by solving

$$
\min_{\Theta \in \mathbb{R}^{n \times p}} Q_{MP}(\Theta) = \frac{1}{2} \left( \sum_{i<j} W_{ij} \| \theta_i - \theta_j \|^2 + \mu \sum_{i=1}^n D_{ii} c_i \| \theta_i - \theta_i^{sol} \|^2 \right)
$$

- Trade-off between smoothing models within neighborhoods and not diverging too much from confident models
- Term $D_{ii} = \sum_j W_{ij}$ is just for normalization
- Strict generalization of Label Propagation (LP) [Zhou et al., 2004]
  - Constant $c_i$’s $\rightarrow$ recover LP
  - Variable $c_i$’s $\rightarrow$ cannot be expressed as LP
  - Our gossip algorithm will readily apply to LP!
SYNCHRONOUS DECENTRALIZED ALGORITHM

- Cannot use closed-form solution (requires global knowledge)
- The following iteration converges to the same quantity

\[
\Theta(t + 1) = (\alpha I + \bar{\alpha} C)^{-1} \left( \alpha P \Theta(t) + \bar{\alpha} C \Theta^{sol} \right)
\]

- \( P = D^{-1}W \) (stochastic similarity matrix)
- \( \alpha \in (0, 1) \) such that \( \mu = (1 - \alpha)/\alpha \), \( \bar{\alpha} = 1 - \alpha \)
- Decomposes into

\[
\theta_i(t + 1) = \frac{1}{\alpha + \bar{\alpha} c_i} \left( \alpha \sum_{j \in \mathcal{N}_i} \frac{W_{ij}}{D_{ii}} \theta_j(t) + \bar{\alpha} c_i \theta_i^{sol} \right)
\]

- This is a decentralized but synchronous process
  - Assumes availability of global clock
  - Synchronization incurs delays (must wait for everyone to finish)
  - All neighbors must be contacted at each step
• Each agent has a **local Poisson clock** and wakes up when it ticks → equivalent to activating a random node at each step $t$

• **Idea of our algorithm**: each agent $i$ maintains a (possibly outdated) knowledge $\tilde{\Theta}_i(t) \in \mathbb{R}^{n \times p}$ of its neighbors’ models
  
  • $\tilde{\Theta}_i(t) \in \mathbb{R}^p$: agent $i$’s model at time $t$
  
  • for $j \neq i$, $\tilde{\Theta}_i(t) \in \mathbb{R}^p$: agent $i$’s last knowledge of the model of $j$
  
  • For $j \notin \mathcal{N}_i \cup \{i\}$ and any $t > 0$, we maintain $\tilde{\Theta}_i(t) = 0$
ASYNCHRONOUS GOSSIP ALGORITHM

· At step $t$, some agent $i$ wakes up and two actions are performed
  1. **Communication step:** agent $i$ selects a random neighbor $j \in \mathcal{N}_i$ w.p. $\pi_i^j$ and both agents update their knowledge of each other:

     $$\tilde{\Theta}_i^j(t + 1) = \tilde{\Theta}_j^i(t)$$
     $$\tilde{\Theta}_j^i(t + 1) = \tilde{\Theta}_i^j(t),$$

  2. **Update step:** agents $i$ and $j$ update their own models based on current knowledge. For $l \in \{i, j\}$:

     $$\tilde{\Theta}_l^i(t + 1) = (\alpha + \bar{\alpha} c_l)^{-1} \left(\alpha \sum_{k \in \mathcal{N}_l} \frac{W_{lk}}{D_{ll}} \tilde{\Theta}_l^k(t + 1) + \bar{\alpha} c_l \theta_l^{sol} \right).$$

· All other variables in the network remain unchanged

· For any $i \in [n]$, $\pi_i \in [0, 1]^n$ must satisfy $\sum_{j=1}^{n} \pi_i^j = 1$ and $\pi_i^j > 0$ if and only if $j \in \mathcal{N}_i$
Theorem ([Vanhaesebrouck et al., 2016])

Let $\tilde{\Theta}(0) \in \mathbb{R}^{n^2 \times p}$ be some initial value and $(\tilde{\Theta}(t))_{t \in \mathbb{N}}$ be the sequence generated by our algorithm. Let $\Theta^* = \arg\min_{\Theta \in \mathbb{R}^{n \times p}} Q_{MP}(\Theta)$ be the optimal solution to model propagation. For any $i \in [n]$,

$$\lim_{t \to \infty} \mathbb{E} \left[ \tilde{\Theta}_i^j(t) \right] = \Theta^*_j \text{ for } j \in \mathcal{N}_i \cup \{i\}.$$ 

Sketch of proof

- Rewrite algorithm as a random iterative process over $\tilde{\Theta} \in \mathbb{R}^{n^2 \times p}$:

$$\tilde{\Theta}(t + 1) = A(t)\tilde{\Theta}(t) + b(t)$$

- Show that spectral radius of $\mathbb{E}[A(t)]$ is smaller than 1
- Exhibit convergence to desired quantity
COLLABORATIVE LEARNING
• Model propagation is very simple but forgets data

• Alternative: learn / propagate models simultaneously by solving

\[ \min_{\Theta \in \mathbb{R}^{n \times p}} Q_{CL}(\Theta) = \sum_{i < j}^{n} W_{ij} \| \theta_i - \theta_j \|^2 + \mu \sum_{i=1}^{n} D_{ii} L_i(\theta_i) \]

• Trade-off between smoothing models within neighborhoods and good accuracy on local datasets

• Note: confidence is built in second term

• More flexibility in settings where different parameter values may lead to similar predictions
We will rely on ADMM [Boyd et al., 2011, Wei and Ozdaglar, 2012], which is popular for solving decentralized consensus problems

\[
\min_{\theta \in \mathbb{R}^p} \sum_{i=1}^{n} \mathcal{L}_i(\theta)
\]

**Main idea**: reformulate our problem as a partial consensus and decouple the objectives

Let \( \Theta_i \) be the set of \(|\mathcal{N}_i| + 1 \) variables \( \theta_j \in \mathbb{R}^p \) for \( j \in \mathcal{N}_i \cup \{i\} \), denote \( \theta_j \) by \( \Theta^j_i \) and define

\[
\mathcal{Q}_{CL}^i(\Theta_i) = \frac{1}{2} \sum_{j \in \mathcal{N}_i} W_{ij} \|\Theta^j_i - \Theta^i_i\|^2 + \mu D_{ii} \mathcal{L}_i(\Theta^i_i),
\]

We can rewrite our problem as

\[
\min_{\Theta \in \mathbb{R}^{n \times p}} \sum_{i=1}^{n} \mathcal{Q}_{CL}^i(\Theta_i)
\]
• **Decoupling**: introduce a local copy $\tilde{\Theta}_i \in \mathbb{R}(|\mathcal{N}_i|+1) \times p$ of the decision variables $\Theta_i$ for each agent $i$

• **Partial consensus**: impose equality constraints on the variables $\tilde{\Theta}_i = \tilde{\Theta}_j$ for all $i \in [n], j \in \mathcal{N}_i$
  - two neighboring agents agree on each other’s personalized model

• Further introduce 4 secondary variables $Z_{ei}^i, Z_{ej}^i, Z_{ei}^j$ and $Z_{ej}^j$ for each edge $e = (i, j)$
  - can be viewed as estimates of the models $\tilde{\Theta}_i$ and $\tilde{\Theta}_j$ known by each end of $e$
  - will allow efficient decomposition of ADMM updates
• Final reformulation: denoting $\tilde{\Theta} = [\tilde{\Theta}_1^T, \ldots, \tilde{\Theta}_n^T]^T \in \mathbb{R}^{(2|E|+n) \times p}$ and $Z \in \mathbb{R}^{4|E| \times p}$

$$\min_{\tilde{\Theta} \in \mathbb{R}^{(2|E|+n) \times p}} \sum_{i=1}^{n} Q^{i}_{CL}(\tilde{\Theta}_i)$$

s.t. $\forall e = (i, j) \in E$, \quad \left\{ \begin{align*}
Z_{ei}^i &= \tilde{\Theta}_i^i, \quad Z_{ei}^j = \tilde{\Theta}_j^i \\
Z_{ej}^i &= \tilde{\Theta}_j^i, \quad Z_{ej}^j = \tilde{\Theta}_j^j
\end{align*} \right.$

where $C_E = \{Z \in \mathbb{R}^{4|E| \times p} | Z_{ei}^i = Z_{ej}^i, Z_{ej}^j = Z_{ei}^j \text{ for all } e = (i, j) \in E\}$

• Constraints involving $\tilde{\Theta}$ can be written $D\tilde{\Theta} + HZ = 0$ where
  - $H = -I$ of dimension $4|E| \times 4|E|$ is diagonal invertible
  - $D \in \mathbb{R}^{4|E| \times (2|E|+n)}$ contains exactly one entry of 1 in each row

• Assumptions of [Wei and Ozdaglar, 2013] satisfied
• The augmented Lagrangian of the problem is

\[ L_\rho(\tilde{\Theta}, Z, \Lambda) = \sum_{i=1}^{n} L^i_\rho(\tilde{\Theta}_i, Z_i, \Lambda_i), \]

where \(\rho > 0\) is a penalty parameter, \(Z \in \mathcal{C}_E\) and

\[ L^i_\rho(\tilde{\Theta}_i, Z_i, \Lambda_i) = q^i_{\text{CL}}(\tilde{\Theta}_i) + \sum_{j:e=(i,j)\in E} \left[ \Lambda^i_{ei}(\tilde{\Theta}^i_i - Z^i_{ei}) + \Lambda^j_{ei}(\tilde{\Theta}^j_i - Z^j_{ei}) + \frac{\rho}{2} \left( \|\tilde{\Theta}^i_i - Z^i_{ei}\|^2 + \|\tilde{\Theta}^j_i - Z^j_{ei}\|^2 \right) \right]. \]

• ADMM iteratively minimize the augmented Lagrangian by alternating
  1. minimization w.r.t. primal variable \(\tilde{\Theta}\)
  2. minimization w.r.t. secondary variable \(Z\)
  3. update of the dual variable \(\Lambda\)
• Assume that agent $i$ wakes up at step $t$ and selects $j \in \mathcal{N}_i$. Denoting $e = (i, j)$

1. Agent $i$ updates its primal variables:

$$\tilde{\Theta}_i(t + 1) = \arg \min_{\Theta \in \mathbb{R}^{(|\mathcal{N}_i|+1) \times p}} L^i_p(\Theta, Z_i(t), \Lambda_i(t)),$$

and sends $\tilde{\Theta}_i(t + 1), \tilde{\Theta}_j(t + 1), \Lambda^i_{ei}(t), \Lambda^j_{ei}(t)$ to agent $j$. Agent $j$ executes the same steps w.r.t. $i$.

2. Using $\tilde{\Theta}_j(t + 1), \tilde{\Theta}_i(t + 1), \Lambda^j_{ei}(t), \Lambda^i_{ei}(t)$ received from $j$, agent $i$ updates its secondary variables $Z^i_{ei}(t + 1)$ and $Z^j_{ei}(t + 1)$ (closed form). Agent $j$ updates its secondary variables symmetrically.

3. Agent $i$ updates its dual variables $\Lambda^i_{ei}(t + 1)$ and $\Lambda^j_{ei}(t + 1)$ (closed-form). Agent $j$ updates its dual variables symmetrically.

• Convergence in $O(1/t)$ [Wei and Ozdaglar, 2013]
EXPERIMENTS
• We consider $n = 300$ agents and a 1D mean estimation task
  (loss $f(\theta; x_i) = \|\theta - x_i\|^2$)
  • Network topology derived from the two moons dataset
  • Each agent $i$ has a true 1D Gaussian distribution $\mu_i$ centered at -1
    or +1 depending on the moon it belongs to
  • Each agent $i$ receives a random number $m_i$ of samples from $\mu_i$
• Confidence values help a lot for imbalanced datasets
• Our MP algorithm has fast convergence without synchronization
• We consider a set of $n = 100$ agents and a linear classification task in $\mathbb{R}^p$ (with hinge loss)
  • Target models lie in a 2D subspace, network weights based on the angle between true models
  • Each agent $i$ receives a random number $m_i$ of samples with label given by the prediction of target model (plus noise)
• Both CL and MP provide great improvements over local models
• CL consistently outperforms MP by significant margin
• Effectively compensating for training size imbalance
COLLABORATIVE LINEAR CLASSIFICATION

- CL algorithm converges as fast as a synchronous approach
- MP much faster to converge and can be used as warm-start to speed up CL
- Number of iterations to converge to near-optimal accuracy scales linearly with network size
FUTURE WORK
FUTURE WORK (AND ADS)

• Study link between similarity graph and generalization performance
• Generic methods to estimate/learn graph weights
• Decentralized discovery of similar peers
• Privacy-preserving mechanisms

Quick ads

• Make sure you attend the NIPS 2016 workshop on Private Multi-Party Machine Learning!
• We have many open positions in our Inria team (tenured, postdocs, PhDs, Master internships) with exciting projects!
Thank you for your attention! Questions?


Distributed Alternating Direction Method of Multipliers.
In Proceedings of the 51th IEEE Conference on Decision and Control (CDC), pages 5445–5450.

On the O(1/k) Convergence of Asynchronous Distributed Alternating Direction Method of Multipliers.
In IEEE Global Conference on Signal and Information Processing (GlobalSIP).

Learning with local and global consistency.
In Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS),
volume 16, pages 321–328.