IBM **Research**

Efficient Protocol for Collaborative Dictionary Learning in Decentralized Networks

T. Idé (“Ide-san”), R. Raymond, and D. T. Phan
IBM Research, T.J. Watson Research Center
IBM Research - Tokyo

Agenda

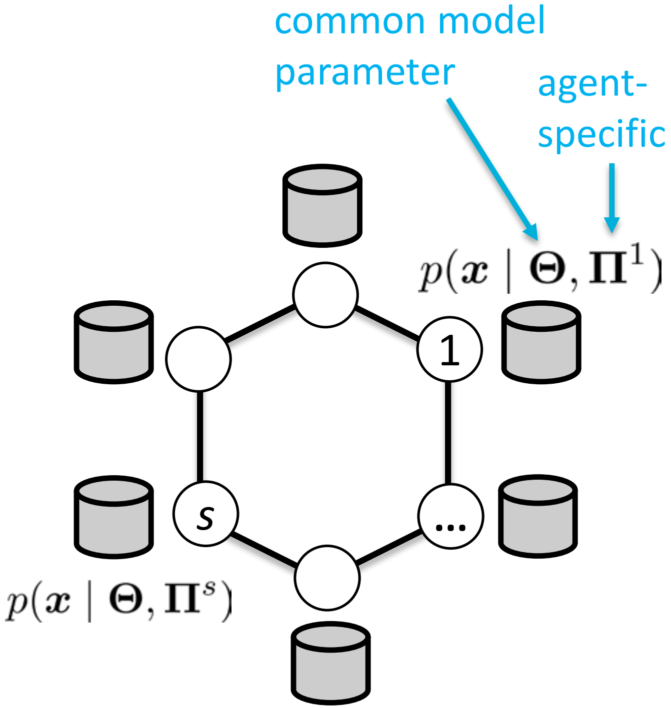
- Problem setting and overview
- Global-local separation in exponential family
- Dynamical consensus with privacy
- Numerical example

What is this in a nutshell?
Multi-task density estimation + a novel set of constraints

- What is multi-task learning?
  - Multiple agents learn their own model based on their own data
  - Leverage commonality among agents to get a better model

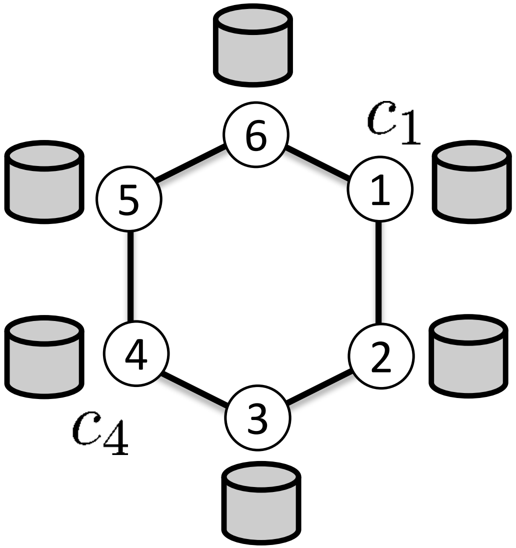
- What are the "**constraints**"?
  - Data privacy
  - Decentralized

This one is tricky



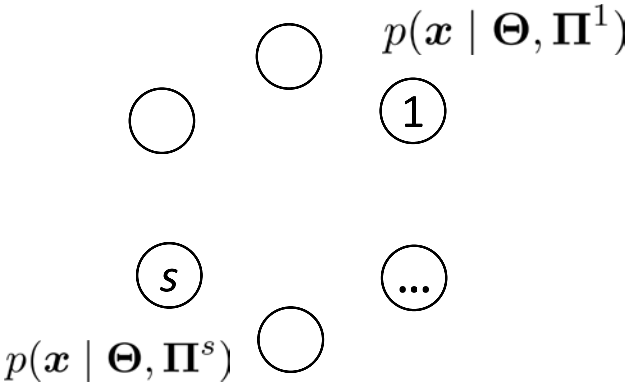
Decentralized learning may be harder than you might think…

- Example: Aggregation (summation)
  - $\bar{c} = c_1 + c_2 + \ldots + c_6$
- Easy? Not really, if we do that only through local communications
  - Broadcasting your data to all?
    - ✓ No! Total privacy breach
  - Select a leader to let her compute?
    - ✓ No! What if she is a bad guy?

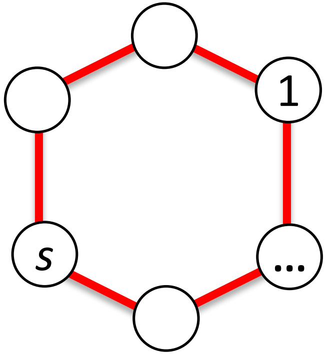


Problem setting:
Decentralized multi-task density estimation with data privacy

Each agent wishes to learn its own probability density

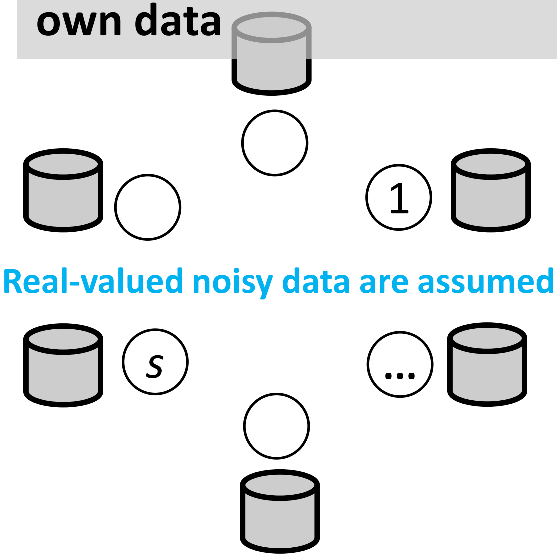


Closed, membership-based, and decentralized network



No central server!

Multiple "semi-honest" agents privately keep own data



Our solution: summary

Maximum likelihood

Mixture of exponential family

Dynamic consensus on commutation graph

Simple secret sharing scheme

For principled probabilistic multi-task learning

For decentralized learning

For data privacy

(For reference) Prior work

Multi-task learning	Decentralized	Data privacy (under distributed environment)
• Actively studied area but mostly for supervised learning • Not many of them are fully probabilistic • Little is known about how to decentralize	• Byzantine protocols assume categorical values • Multi-agent consensus methods are not in the context of multi-task learning	• Differential privacy is problematic in distributed environment • Secure multi-party computation typically needs a central server • Homomorphic encryption is too slow

Agenda

- Problem setting and overview
- Global-local separation in exponential family
- Dynamical consensus with privacy
- Numerical example

We employ a mixture of exponential family for multi-task density estimation

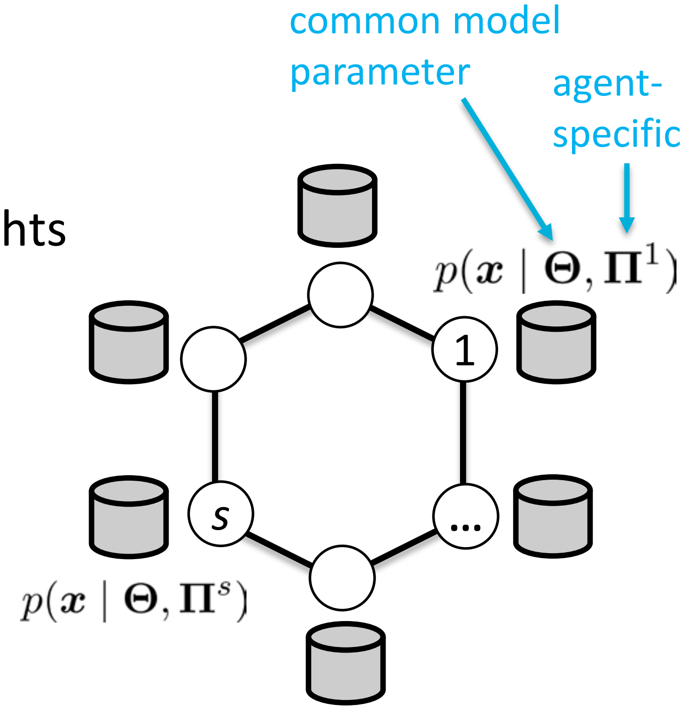
- Each agent holds its own data

$$\mathcal{D}^s = \{\boldsymbol{x}^{s(n)} \mid n = 1, \ldots, N^s;\ \boldsymbol{x}^{s(n)} \in \mathbb{R}^M\}$$

- Employ a mixture model with agent-specific weights
  - $$p(\boldsymbol{x} \mid \boldsymbol{\Theta}, \boldsymbol{\Pi}^s) = \sum_{k=1}^{K} \pi_k^s f(\boldsymbol{x} \mid \boldsymbol{\theta}_k)$$
  - The mixture coefficients $\{\boldsymbol{\pi}^1, \ldots, \boldsymbol{\pi}^S\}$ is agent-specific
  - $\{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K\}$ are shared by all the agents

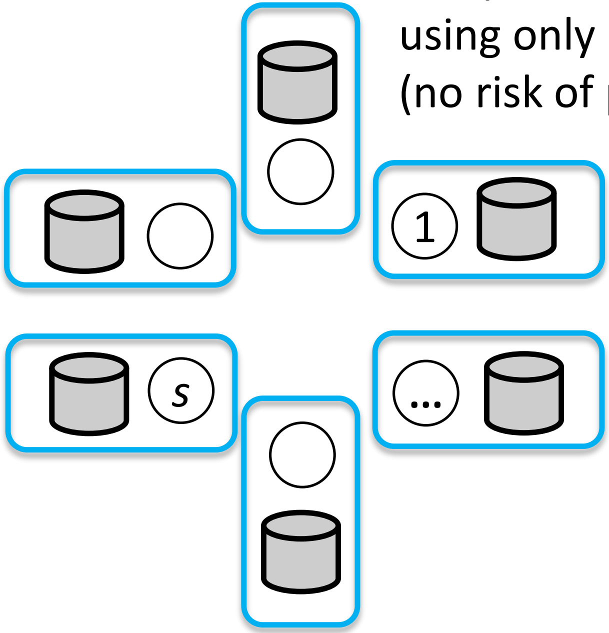
- For f, employ exponential family

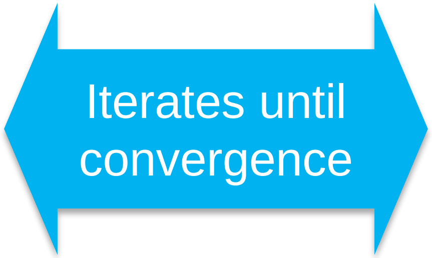
$$f(\boldsymbol{x} \mid \boldsymbol{\theta}_k) = G(\boldsymbol{\theta}_k) H(\boldsymbol{x}) \exp\left\{\boldsymbol{\eta}(\boldsymbol{\theta}_k)^\top \boldsymbol{T}(\boldsymbol{x})\right\}$$



Exponential family naturally leads to Global-Local Separation in maximum likelihood

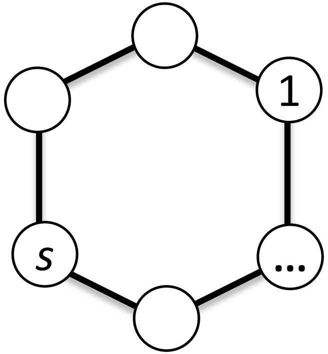
Local updates:
compute statistics locally
using only my own data
(no risk of privacy breach)





Global consensus:

- Compute aggregation
- Perform optimization to store a unique result



Agenda

- Problem setting and overview
- Global-local separation in exponential family
- Dynamical consensus with privacy
- Numerical example

Decentralized aggregation = Finding stationary state of Markovian process

- Consider an aggregation task in general:
  - $$\bar{c} = \sum_{s=1}^{S} c^s = \underbrace{\mathbf{1}}_{S\text{-dimensional vector of ones}}^\top \boldsymbol{c}$$

- Idea: consider Markovian process whose stationary state is proportional to the **1** vector

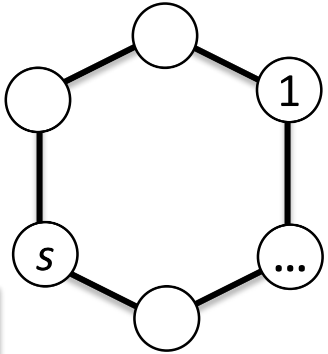
$$c^s \leftarrow c^s + \epsilon \sum_{j=1}^{S} \mathsf{A}_{s,j}(c^j - c^s) \quad \text{or} \quad \boldsymbol{c} \leftarrow \underbrace{[\mathbf{I} - \epsilon(\mathbf{D} - \mathbf{A})]}_{\text{The largest eigenvector is } \mathbf{1}}\boldsymbol{c}$$

  - **A**: Incidence matrix of the communication graph
  - **D**: Degree matrix of **A**

- This update equation converges to $\bar{c}$ in all nodes

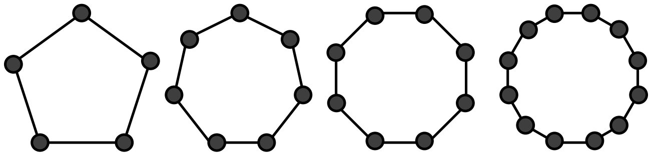
Global consensus:

- Compute aggregation
- Perform optimization to store a unique result

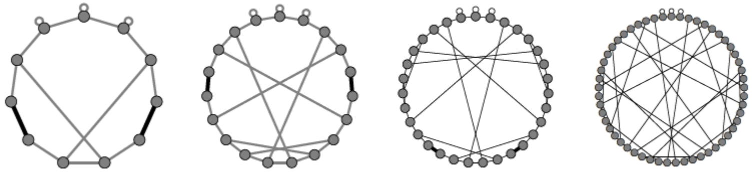


What kind of communication graph *A* should be chosen?

- Cycle graph
  - Pros
    - ✓ Sparse. Minimum number of neighbors. Good for privacy
    - ✓ Symmetric. Regular. Good for democracy.
    - ✓ Analytic expression of eigenvalues allows detailed convergence analysis
  - Cons
    - ✓ Slow convergence.

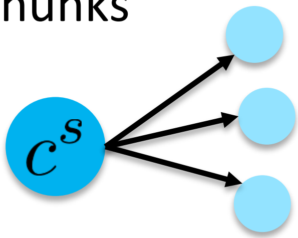


- "Cycle with inverse chord" (or modified cycle graph)
  - Pros
    - ✓ Sparse.
    - ✓ Fast convergence (~ log S)
  - Cons
    - ✓ Not regular (=not fully democratic)
    - ✓ Eigenvalue is non-smooth with *S*



How can we guarantee data privacy?
Simple secret sharing scheme by random chunking

- Randomly split datum into N_c chunks
  - $$\bar{c} = \sum_{s=1}^{S} c^s$$



$$c^s = c^{s[1]} + c^{s[3]} + c^{s[3]}$$

- Do dynamic consensus N_c times and sum
  - $$\bar{c} = \bar{c}^{[1]} + \bar{c}^{[2]} + \bar{c}^{[3]}$$
  - Note: Each chunking round has to use a different communication graph
    - ✓ Can be done simply by shuffling network address (needs network router's help)
    - ✓ Take a large Nc to make the probability of sending all the chucks to the same neighbor negligible

Agenda

- Problem setting and overview
- Global-local separation in exponential family
- Dynamical consensus with privacy
- Numerical example

Much faster than homomorphic encryption-based decentralized approach

- Proposed method is orders of magnitude faster than an existing fully-decentralized consensus algorithm using homomorphic encryption [Ruan+ 17;19]

- Number of iteration is **~ log S** when using the modified cycle graph
  - o *S*: The number of nodes

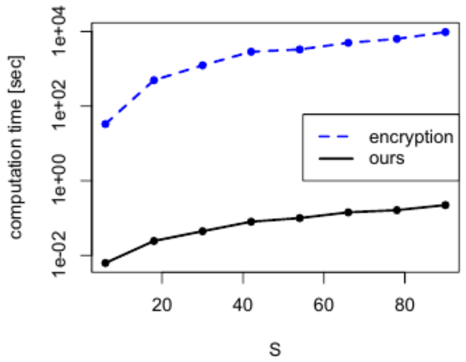


Figure 5: Computation time comparison between the proposed dynamic random chunking and the encryption-based method.

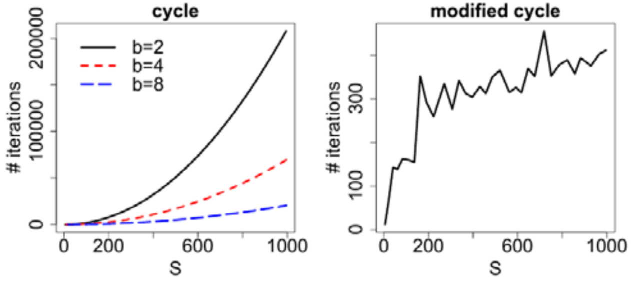


Figure 4: The number of iterations t to achieve $\left(\frac{\nu_2}{\nu_1}\right)^t = \frac{1}{\sqrt{S}}10^{-3}$.

Thank you!