Online Tensor Method for Community Detection

Niranjan U N

UCI

Joint work with Anandkumar, Hakeem, Huang, Verma.

2/7/13
Outline

1. Introduction

2. Community Model
   - Tensor Forms for Stochastic Block Model
   - Extension to Mixed Membership Model

3. Learning Algorithm
   - Pre-processing
   - Stochastic Tensor Gradient Descent
   - Post-processing

4. GPU Implementation

5. Results
   - Validation Methods
   - Real-world Datasets

6. Conclusion and Future Work
Outline

1. Introduction

2. Community Model
   - Tensor Forms for Stochastic Block Model
   - Extension to Mixed Membership Model

3. Learning Algorithm
   - Pre-processing
   - Stochastic Tensor Gradient Descent
   - Post-processing

4. GPU Implementation

5. Results
   - Validation Methods
   - Real-world Datasets

6. Conclusion and Future Work
Community Models in Social Networks

Motivation: Overall Picture

- What is the context of our problem?
- Why are we interested?
- How do we approach this problem?
- When are the methods we develop useful?
- Where do we begin?
- Other important points?
Community Membership Model

- $k$: communities and network size $n$
- $\pi_u$: community membership vector of node $u$
- Single community membership: $\pi_u = e_i$ if node $u$ is in community $i$
- $e_i$ is the basis vector in $i^{th}$ coordinate
- Independent draws for community vectors $\{\pi_u\}$

Edge Generation Model

- Edges are conditionally independent given community memberships
- Probability of an edge from $u$ to $v$ is $\pi_u^\top P \pi_v$, $P \in [0, 1]^{k \times k}$
- Notice that $\pi_u^\top P \pi_v = P_{i,j}$ if $\pi_u = e_i$ and $\pi_v = e_j$
Mixed Membership Model

- \( k \) communities and network size \( n \)
- \( \pi_u \in \mathbb{R}_+^k \): community membership of node \( u \)
- Entries of \( \pi_u \): fractional memberships

Dirichlet Community Membership Model

- \( \{\pi_u\} \) are independent draws from Dirichlet distribution

\[
\mathbb{P}[\pi_u] \propto \prod_{j=1}^{k} \pi_u(j)^{\alpha_j - 1}, \quad \sum_{j=1}^{k} \pi_u(j) = 1
\]

Edge generation: same as stochastic block model

- Edges are conditionally independent given community memberships
- Probability of an edge from \( u \) to \( v \) is \( \pi_u^\top P \pi_v \), \( P \in [0, 1]^{k \times k} \)

Observations regarding Dirichlet distribution

\[
\mathbb{P}[\pi] \propto \prod_{j=1}^{k} \pi(j)^{\alpha_j - 1}, \quad \sum_{j=1}^{k} \pi(j) = 1
\]

Geometric View

Dirichlet distribution supported over the simplex \( \Delta^{k-1} \)

\[
\Delta^{k-1} := \{ \pi \in \mathbb{R}^k, \pi(i) \in [0, 1], \sum_i \pi(i) = 1 \}
\]

- Concentration parameter \( \alpha_0 := \sum_j \alpha_j \)
- \( \alpha_0 \to 0 \): stochastic block model
- \( \alpha_0 = 1 \): mixed membership model (our focus for detecting overlapping communities)
- \( \alpha_0 \to \infty \): fully dense vector
- Roughly, level of sparsity in \( \pi \) is \( O(\alpha_0) \)

Observations regarding Dirichlet distribution

\[ P[\pi] \propto \prod_{j=1}^{k} \pi(j)^{\alpha_j - 1}, \quad \sum_{j=1}^{k} \pi(j) = 1 \]

Geometric View

Dirichlet distribution supported over the simplex \( \Delta^{k-1} \)

\[ \Delta^{k-1} := \{ \pi \in \mathbb{R}^k, \pi(i) \in [0, 1], \sum_i \pi(i) = 1 \} \]

- Concentration parameter
  \[ \alpha_0 := \sum_j \alpha_j \]

- \( \alpha_0 \to 0 \): stochastic block model
- \( \alpha_0 = 1 \): mixed membership model (our focus for detecting overlapping communities)
- \( \alpha_0 \to \infty \): fully dense vector
- Roughly, level of sparsity in \( \pi \) is \( O(\alpha_0) \)

Observations regarding Dirichlet distribution

\[ \mathbb{P}[\pi] \propto \prod_{j=1}^{k} \pi(j)^{\alpha_j - 1}, \quad \sum_{j=1}^{k} \pi(j) = 1 \]

Geometric View

Dirichlet distribution supported over the simplex \( \Delta^{k-1} \)

\[ \Delta^{k-1} := \{ \pi \in \mathbb{R}^k, \pi(i) \in [0,1], \sum_i \pi(i) = 1 \} \]

- Concentration parameter \( \alpha_0 := \sum_j \alpha_j \)
- \( \alpha_0 \to 0 \): stochastic block model
- \( \alpha_0 = 1 \): mixed membership model (our focus for detecting overlapping communities)
- \( \alpha_0 \to \infty \): fully dense vector

Roughly, level of sparsity in \( \pi \) is \( O(\alpha_0) \)

Observations regarding Dirichlet distribution

\[ \mathbb{P}[\pi] \propto \prod_{j=1}^{k} \pi(j)^{\alpha_j - 1}, \quad \sum_{j=1}^{k} \pi(j) = 1 \]

Geometric View

Dirichlet distribution supported over the simplex \( \Delta^{k-1} \)

\[ \Delta^{k-1} := \{ \pi \in \mathbb{R}^k, \pi(i) \in [0, 1], \sum_{i} \pi(i) = 1 \} \]

- Concentration parameter \( \alpha_0 := \sum_{j} \alpha_j \)
- \( \alpha_0 \to 0 \): stochastic block model
- \( \alpha_0 = 1 \): mixed membership model (our focus for detecting overlapping communities)
- \( \alpha_0 \to \infty \): fully dense vector
- Roughly, level of sparsity in \( \pi \) is \( O(\alpha_0) \)

Observations regarding Dirichlet distribution

\[ \mathbb{P}[\pi] \propto \prod_{j=1}^{k} \pi(j)^{\alpha_j - 1}, \quad \sum_{j=1}^{k} \pi(j) = 1 \]

Geometric View

Dirichlet distribution supported over the simplex \( \Delta^{k-1} \)

\[ \Delta^{k-1} := \{ \pi \in \mathbb{R}^k, \pi(i) \in [0, 1], \sum_i \pi(i) = 1 \} \]

- Concentration parameter \( \alpha_0 := \sum_j \alpha_j \)
- \( \alpha_0 \to 0 \): stochastic block model
- \( \alpha_0 = 1 \): mixed membership model (our focus for detecting overlapping communities)
- \( \alpha_0 \to \infty \): fully dense vector
- Roughly, level of sparsity in \( \pi \) is \( O(\alpha_0) \)

Employing Mixed Membership Models

Advantages

- Mixed membership models incorporate overlapping communities
- Stochastic block model is a special case
- Same edge generation process: retains conditional independence
- Model sparse community membership: natural in most networks

Challenges in Learning Mixed Membership Models

- Identifiability: when can parameters be estimated?
- Guaranteed learning: what input required?
- Potentially large sample and computational complexities

Summary of Previous Results

Results

- **Guaranteed** learning of overlapping community models.
- **Correctness** under exact moments: edges and 3-star counts.
- **Efficient** sample and computational complexity.

Scaling Requirements

$k$ communities, $n$ nodes, concentration parameter: $\alpha_0$, intra/inter-community edge probabilities: $p, q$

\[
n = \tilde{\Omega}(k^2(\alpha_0 + 1)^2), \quad \frac{p - q}{\sqrt{p}} = \tilde{\Omega}\left(\frac{(\alpha_0 + 1)k}{n^{1/2}}\right).
\]

- For stochastic block model ($\alpha_0 = 0$), tight results
- Performance degradation as $\alpha_0$ increases
- Efficient method for sparse community overlaps

---

Overview of Previous Techniques

Method of Moments and Spectral Approach

- **Inverse moment method**: solve equations relating parameters to observed moments
- **Spectral approach**: reduce equation solving to computing the spectrum of the observed moments
- **Non-convex** but computationally tractable approaches

Spectral Approach to Learning Mixed Membership Models

- **Edge and Subgraph Counts**: moments of the observed network
- **Tensor Spectral Approach**: low rank tensor form and efficient decomposition via power method

Tensor Preliminaries

- For a tensor $T$, define (for matrices $V_i$ of appropriate dimensions)

$$[T(W_1, W_2, W_3)]_{i_1,i_2,i_3} := \sum_{j_1,j_2,j_3} (T)_{j_1,j_2,j_3} \prod_{m\in[3]} W_m(j_m,i_m)$$

- For a matrix $M$,

$$M(W_1, W_2) := W_1^\top MW_2$$

- For a symmetric tensor $T$ of the form

$$T = \sum_{r=1}^{k} \lambda_r \phi_r \otimes 3$$

$$T(v,v,v) = \sum_{r\in[k]} \lambda_r (v^\top \phi_r) \otimes 3$$

$$T(I,v,v) = \sum_{r\in[k]} \lambda_r \langle v, \phi_r \rangle^2 \phi_r$$

- $k$ terms in the sum $\implies$ rank-$k$ tensor.
Orthogonal Tensor Eigen Analysis

Consider orthogonal symmetric tensor $T = \sum_i w_i \mu_i \otimes^3$

$$T = \sum_{i=1}^k w_i \mu_i \otimes^3. \quad T(I, \mu_i, \mu_i) = w_i \mu_i$$

Obtaining eigenvectors through power iterations

$$u \mapsto \frac{T(I, u, u)}{\|T(I, u, u)\|}$$

Challenges and Solution

- Challenge: Other eigenvectors present
  Solution: Only stable vectors are basis vectors $\{\mu_i\}$

- Challenge: empirical moments
  Solution: robust tensor decomposition methods
Some Related Work

Stochastic Methods


GPU Implementation

- Soman, Narang. Fast Community Detection Algorithm with GPUs and Multicore Architectures. IPDPS, 2011.

Community Detection

Outline

1. Introduction

2. Community Model
   - Tensor Forms for Stochastic Block Model
   - Extension to Mixed Membership Model

3. Learning Algorithm
   - Pre-processing
   - Stochastic Tensor Gradient Descent
   - Post-processing

4. GPU Implementation

5. Results
   - Validation Methods
   - Real-world Datasets

6. Conclusion and Future Work
Outline

1. Introduction
2. Community Model
   - Tensor Forms for Stochastic Block Model
   - Extension to Mixed Membership Model
3. Learning Algorithm
   - Pre-processing
   - Stochastic Tensor Gradient Descent
   - Post-processing
4. GPU Implementation
5. Results
   - Validation Methods
   - Real-world Datasets
6. Conclusion and Future Work
Graph Moments for our Method

- Subgraph counts as moments of a random graph distribution

Edge Counts
- Consider partition $X, A, B, C$.
- Adjacency Submatrices $G_{X,A}, G_{X,B}, G_{X,C}$

3-Star Count Tensor
- # of 3-star subgraphs from $X$ to $A, B, C$.
- $M_3(a, b, c) := \frac{1}{|X|} \# \text{ of 3-stars with leaves } a,b,c$
- $M_3 \in \mathbb{R}^{|A| \times |B| \times |C|}$ is a tensor

Learning via 3-Star Count Tensor
Stochastic Block Model: Classical Approach

Community Membership Model
- $k$ communities and network size $n$
- $\pi_u = e_i$ if node $u$ is in community $i$.

Edge Generation Model
- Edges are conditionally independent given community memberships
- Probability of an edge from $u$ to $v$ is $\pi_u^T P \pi_v$, $P \in [0, 1]^{k \times k}$. 
Recall $\Pi$ is the membership matrix of the nodes, $P$ is the community connectivity matrix.

- Denote $\Pi_A = [\pi_a]_{a \in A}$, $\lambda_i = \mathbb{P} [\pi = e_i]$ and $F_A := \Pi_A^T P^T$.

**Neighborhood vector**

- Neighborhood vectors: $G_{x,A}$, $G_{x,B}$, $G_{x,C}$

\[
\mathbb{E}[G_{x,A}^\top | \Pi_A, x] = \pi_x^\top P \Pi_A = \Pi_A^\top P^\top \pi_x = F_A \pi_x
\]

**Linear Model for Edge Generation**
3-star Tensor Moment Form

- #3-star subgraphs from $X$ to $A, B, C$.

Consider an element in tensor $M_3$

$$M_3(a, b, c) = |X|^{-1} \sum_{x \in X} G(x, a) G(x, b) G(x, c).$$

$$M_3 := |X|^{-1} \sum_{x \in X} [G^\top_{x,A} \otimes G^\top_{x,B} \otimes G^\top_{x,C}]$$

Conditional Independence of Edges

$$\mathbb{E}[M_3|\Pi_{A,B,C,X}] = \sum_{x \in X} |X|^{-1} [(F_A\pi_x) \otimes (F_B\pi_x) \otimes (F_C\pi_x)]$$

Taking expectation over all $x \in X$

$$\mathbb{E}[M_3|\Pi_{A,B,C}] = \sum_i \lambda_i [(F_A)_i \otimes (F_B)_i \otimes (F_C)_i]$$

Goal: Recover $F_A, F_B, F_C, \tilde{\lambda}$
1. Introduction

2. Community Model
   - Tensor Forms for Stochastic Block Model
   - Extension to Mixed Membership Model

3. Learning Algorithm
   - Pre-processing
   - Stochastic Tensor Gradient Descent
   - Post-processing

4. GPU Implementation

5. Results
   - Validation Methods
   - Real-world Datasets

6. Conclusion and Future Work
Moment Forms for the Mixed Membership Model

Recall Mixed Membership Model

The membership vectors are $\pi_u \sim \text{Dir}(\alpha)$

Conditional independence of the edges

Still a linear multi-view model

$$\mathbb{E}[M_3|\Pi_{A,B,C,X}] = \sum_{x \in X} |X|^{-1}[(F_A\pi_x) \otimes (F_B\pi_x) \otimes (F_C\pi_x)]$$

But expectation over $x \in X$ does not preserve tensor form

$$\mathbb{E}[M_3|\Pi_{A,B,C}] \neq \sum_i \lambda_i [(F_A)_i \otimes (F_B)_i \otimes (F_C)_i] \pi_x \otimes^3 \text{ no longer a diagonal tensor}$$

so adjust the moments by subtracting extraneous terms

This leads to a unified learning framework via tensor CP decomposition
Outline

1. Introduction

2. Community Model
   - Tensor Forms for Stochastic Block Model
   - Extension to Mixed Membership Model

3. Learning Algorithm
   - Pre-processing
   - Stochastic Tensor Gradient Descent
   - Post-processing

4. GPU Implementation

5. Results
   - Validation Methods
   - Real-world Datasets

6. Conclusion and Future Work
Overview of Our Current Techniques

Pre-processing

- **Partitioning**: random partitioning and $k$-means heuristic on both weighted and unweighted adjacency matrices
- **Whitening**: obtaining symmetric and orthogonal tensor for tractable decomposition

Stochastic Tensor Gradient Descent

- **Iterative Approach**: low rank decomposition via stochastic gradient descent on a loss function
- **Device Implementation**: efficient implementation of linear algebraic operations on the GPU

Post-processing

- **Thresholding**: to recover the spectrum
- **$p$-values, error function and match ratio**: for validation against ground truth
Outline

1 Introduction

2 Community Model
   - Tensor Forms for Stochastic Block Model
   - Extension to Mixed Membership Model

3 Learning Algorithm
   - Pre-processing
     - Stochastic Tensor Gradient Descent
   - Post-processing

4 GPU Implementation

5 Results
   - Validation Methods
   - Real-world Datasets

6 Conclusion and Future Work
Partitioning

- **Theoretically**, any random partition of size $\Theta(n)$ has sufficient number of 3-stars to faithfully recover the mixed membership community model.

- In **practice**, we use $k$-means with $L_1$-norm (Hamming distance)
  
  - Partition the head nodes and the leaf nodes
  - Cluster the leaf nodes with $k = 3$
  - Split each cluster into three subparts randomly
  - Combine a subpart from each cluster to get a new cluster of leaves
  - Similarly, get two more partitions.

- **Why this heuristic?** because it intuitively ensures **well-distributed partitioning**, and is computationally fast.
Whitening: Convert to Orthogonal Symmetric Tensor

- Define $Pairs(Y_1, Y_2) := \mathbb{E}_x [G_{x,Y_1}^\top \otimes G_{x,Y_2}^\top]$
- Transform $B$: $Z_B := Pairs(A, C)(Pairs(B, C))^\dagger$ and $\tilde{G}_{x,B}^\top = Z_B G_{x,B}^\top$
- Recall $F = \Pi^\top P^\top \implies \tilde{G}_{x,B}^\top = F_A F_B^\dagger G_{x,B}^\top$
- Compute the second moment as
  $$M_2 = \mathbb{E}_x \left[ \tilde{G}_{x,C}^\top \otimes \tilde{G}_{x,B}^\top \mid \Pi_A, \Pi_B, \Pi_C \right] = \frac{1}{n_X} F_A \left( \sum_{x \in X} \pi_x \pi_x^\top \right) F_A^\top$$
- Substitute the Dirichlet moments.
- To get the symmetric form, adjust the off-diagonal terms by subtracting extraneous terms.
- Similarly, set $C'$. 
Dimensionality Reduction of the 3-star Count Tensor

- **Idea**
  - Symmetrize third moment
  - Compute whitening matrix
  - Orthogonalize using multilinear transformation

- **Note:** Multilinear transformations consist of inner products suitable for GPU computing.

- Recall $k \ll n$ (number of communities), suitable for device memory and fast computation.
Outline

1. Introduction
2. Community Model
   - Tensor Forms for Stochastic Block Model
   - Extension to Mixed Membership Model
3. Learning Algorithm
   - Pre-processing
   - Stochastic Tensor Gradient Descent
   - Post-processing
4. GPU Implementation
5. Results
   - Validation Methods
   - Real-world Datasets
6. Conclusion and Future Work
Optimization Problem

Samples from Whitening Step

- Denote a 3-star, after the whitening step, with head \( t \in X \) as \( \mathcal{R}^t \).
- This is just one of the \( t \) sample terms in the summation to compute whitened tensor.
- Use matrix-matrix products: let \( \mathcal{R}^t = G_{t,A}^\top W \otimes \tilde{G}_{t,B}^\top W \otimes \tilde{G}_{t,C}^\top W \).

Objective Function

- \( \arg \min_v \left\{ \left\| \sum_{i \in [k]} v_i^\otimes 3 - \frac{1}{n_x} \sum_{t \in X} \mathcal{R}^t \right\|_F^2 \right\} \) where \( v_i \) are the eigenvectors of the data tensor to be computed.
- This is the low rank formulation of the tensor eigenvalue problem \( T(I, v, v) = \lambda v \).
- Recap: why are we interested in this? we use these to recover the membership estimates.

Never form the tensor explicitly
Stochastic Update

Update Equations

\[ v_{i}^{t+1} \leftarrow v_{i}^{t} - \eta \frac{\partial L^{t}(v)}{\partial v_{i}} \]  where \( \eta \) is the learning rate.

Derivative Evaluated at a Single Datapoint

- Expand the square; drop constant terms.

\[ \frac{\partial L^{t}(v)}{\partial v_{i}} = 3 \sum_{j=1}^{k} \langle v_{j}, v_{i} \rangle^{2} v_{j} - \sum_{\{A,B,C\}} \langle v_{i}, w_{A}^{t} \rangle \langle v_{i}, w_{B}^{t} \rangle w_{C}^{t} \]

where \( w_{A}^{t}, w_{B}^{t} \) and \( w_{C}^{t} \) are the whitened samples.

- Trade-off: minimize the orthogonality cost while maximizing the correlation reward.

- Note that the derivative is written entirely in terms of inner products.
Outline

1. Introduction
2. Community Model
   - Tensor Forms for Stochastic Block Model
   - Extension to Mixed Membership Model
3. Learning Algorithm
   - Pre-processing
   - Stochastic Tensor Gradient Descent
   - Post-processing
4. GPU Implementation
5. Results
   - Validation Methods
   - Real-world Datasets
6. Conclusion and Future Work
Membership Estimates

Recovering the Spectrum
- Compute the tensor eigenvalues as $\lambda_i = ||v_i||^3$.

Thresholding
- Tune a threshold parameter $\tau$ to get the membership of all the nodes in the network.

---

Outline

1. Introduction

2. Community Model
   - Tensor Forms for Stochastic Block Model
   - Extension to Mixed Membership Model

3. Learning Algorithm
   - Pre-processing
   - Stochastic Tensor Gradient Descent
   - Post-processing

4. GPU Implementation

5. Results
   - Validation Methods
   - Real-world Datasets

6. Conclusion and Future Work
Overview

Why do we choose GPUs?
- Inner products and matrix operations
- Efficient BLAS libraries for GPUs exist permitting relatively fast code development

Benefit: Hundreds of cores; massive parallelism.

Challenge: Limited memory for large scale data, although the algorithm is scalable; requires careful buffer reuse and data transfers via PCI.

Key insight: Never form the tensor explicitly.
GPU Compute Architecture

Kepler Architecture (Quadro K5000)

- Each GPU has many **SMX units** made of CUDA cores, L1 cache and shared memory.
- All the SMX units have a common L2 cache and **global memory**.
- **CUDA kernels**: multiple threads
GPU Programming

CULA-Dense Library

- BLAS routines provided.
- **Standard interface:** CPU-GPU transfers and synchronization are automatically handled.
- **Device interface:** direct access to GPU memory; programmer must explicitly handle.
- Buffer transfers via PCI are costly and scale poorly; hence, **device interface** more suited iterative algorithms.

![Graph showing CPU-GPU buffer round-trip transaction time vs. logarithm of buffer size divided by 8]
How do we obtain speed-up?

- **Precision matters:** double precision means roughly double the space and double the time.
- For **STGD**, identify common inner product terms while collapsing the derivative terms and construct quadratic forms.
- **Note:** whitened tensor not stored, transform adjacency submatrices with whitening matrix and sample for **STGD**.
- Reuse GPU buffers.
- Parallelize across eigenvectors; use **BLAS III** as much as possible.
Outline

1. Introduction
2. Community Model
   - Tensor Forms for Stochastic Block Model
   - Extension to Mixed Membership Model
3. Learning Algorithm
   - Pre-processing
   - Stochastic Tensor Gradient Descent
   - Post-processing
4. GPU Implementation
5. Results
   - Validation Methods
   - Real-world Datasets
6. Conclusion and Future Work
Outline

1 Introduction

2 Community Model
   • Tensor Forms for Stochastic Block Model
   • Extension to Mixed Membership Model

3 Learning Algorithm
   • Pre-processing
   • Stochastic Tensor Gradient Descent
   • Post-processing

4 GPU Implementation

5 Results
   • Validation Methods
   • Real-world Datasets

6 Conclusion and Future Work
Measures and Criteria

Note: ground truth known to us.

*p*-value

- **Why?** learning algorithm is unsupervised, so results in rows of $\hat{\Pi}$ being permuted as compared to $\Pi$; also $\hat{k} \neq k$.
- **Null:** $H_{ij} = \rho(\hat{\Pi}_i, \Pi_j) = 0$; choose significance level be $\alpha$ (eg, 0.05).
- **$t$-test:** test statistic is $T_{ij} := \frac{\rho(\hat{\Pi}_i, \Pi_j) \sqrt{n-2}}{\sqrt{1-\rho(\hat{\Pi}_i, \Pi_j)^2}}$ and $T_{ij} \sim t_{n-2}$.
- **$p$-value:** probability of not rejecting the null; form a $\hat{k} \times k$ matrix.

Error Function

- $E := \frac{1}{nk} \sum_{\{p-val[\hat{\Pi}_i, \Pi_j] \leq 0.05\}} \left\{ \sum_{x \in X} |\hat{\Pi}_i(x) - \Pi_j(x)| \right\}$

How Well do Estimated Memberships Match?

- $M := \frac{1}{k} \sum_{j \in [k]} \left\{ \sum_{i \in [\hat{k}]} \left\{ p - val \left[ \hat{\Pi}_i, \Pi_j \right] \leq 0.05 \right\} \right\} \geq 1$
Outline

1. Introduction

2. Community Model
   - Tensor Forms for Stochastic Block Model
   - Extension to Mixed Membership Model

3. Learning Algorithm
   - Pre-processing
   - Stochastic Tensor Gradient Descent
   - Post-processing

4. GPU Implementation

5. Results
   - Validation Methods
   - Real-world Datasets

6. Conclusion and Future Work
Facebook: Symmetric Graph

- UNC FB network: 18163 users, mean deg = 84, median deg = 62
- Ground truth: 360 significant communities with $\geq 20$ users each.
- Goal: learn user features, given connectivity.
- Application: recommender systems.

Results by Tuning $\tau$

<table>
<thead>
<tr>
<th>$k$</th>
<th>$\hat{k}$</th>
<th>Error</th>
<th>Recovery Ratio</th>
<th>Time (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>360</td>
<td>100</td>
<td>0.0124</td>
<td>39%</td>
<td>189.9</td>
</tr>
<tr>
<td>360</td>
<td>100</td>
<td>0.0185</td>
<td>100%</td>
<td>189.9</td>
</tr>
<tr>
<td>360</td>
<td>500</td>
<td>0.0142</td>
<td>71%</td>
<td>468.4</td>
</tr>
<tr>
<td>360</td>
<td>500</td>
<td>0.0175</td>
<td>100%</td>
<td>468.4</td>
</tr>
</tbody>
</table>

Note on quality vs quantity: error vs recovery ratio trade-off, i.e., recover top communities with higher accuracy or recover more number of communities with lower accuracy.
Yelp: Asymmetric Graph

- **Yelp challenge**: bipartite graph with 45,981 users, 11,537 businesses.
- Review stars are the edge weights (approximated by Poisson model).
- **Data cleaning**: remove small and closed businesses; focus on AZ.
- **Business attributes**: categories and location (ground truth).
- **User attributes**: male, female and review counts.

### Results by Tuning $\tau$

<table>
<thead>
<tr>
<th>$k$</th>
<th>$\hat{k}$</th>
<th>Error</th>
<th>Recovery Ratio</th>
<th>Time (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>159</td>
<td>100</td>
<td>0.0233</td>
<td>43.4%</td>
<td>1126.6</td>
</tr>
<tr>
<td>159</td>
<td>100</td>
<td>0.0608</td>
<td>79.87%</td>
<td>1126.6</td>
</tr>
<tr>
<td>159</td>
<td>500</td>
<td>0.0638</td>
<td>72.33%</td>
<td>1706.2</td>
</tr>
<tr>
<td>159</td>
<td>500</td>
<td>0.3361</td>
<td>100%</td>
<td>1706.2</td>
</tr>
</tbody>
</table>

Note on quality vs quantity: error vs recovery ratio trade-off, i.e., recover top communities with higher accuracy or recover more number of communities with lower accuracy.
<table>
<thead>
<tr>
<th>Rank</th>
<th>Category</th>
<th>Business</th>
<th>Stars</th>
<th>Avg. Stars</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Latin American</td>
<td>Salvadoreno Restaurant</td>
<td>4.0</td>
<td>3.70</td>
</tr>
<tr>
<td>2</td>
<td>Gluten Free</td>
<td>P.F. Chang's China Bistro</td>
<td>3.5</td>
<td>3.68</td>
</tr>
<tr>
<td>3</td>
<td>Hobby Shops</td>
<td>Make Meaning</td>
<td>4.5</td>
<td>3.68</td>
</tr>
<tr>
<td>4</td>
<td>Mass Media</td>
<td>KJZZ 91.5FM</td>
<td>4.0</td>
<td>3.70</td>
</tr>
<tr>
<td>5</td>
<td>Yoga</td>
<td>Sutra Midtown</td>
<td>4.5</td>
<td>3.69</td>
</tr>
<tr>
<td>6</td>
<td>Churches</td>
<td>St Andrew the Apostle Catholic Church</td>
<td>4.5</td>
<td>3.71</td>
</tr>
<tr>
<td>7</td>
<td>Art Galleries</td>
<td>Sette Lisa Gallery</td>
<td>4.5</td>
<td>3.69</td>
</tr>
<tr>
<td>8</td>
<td>Libraries</td>
<td>Cholla Branch Library</td>
<td>4.0</td>
<td>3.70</td>
</tr>
<tr>
<td>9</td>
<td>Religious Organizations</td>
<td>St Andrew the Apostle Catholic Church</td>
<td>4.5</td>
<td>3.70</td>
</tr>
<tr>
<td>10</td>
<td>Wickenburg(city)</td>
<td>A Taste of the Caribbean</td>
<td>4.0</td>
<td>3.68</td>
</tr>
</tbody>
</table>

Top recovered business categories with lowest error and businesses with highest membership weights in $\hat{\Pi}$. Average stars denote the average review stars of estimated members in that category.
Yelp Results and Visualization

<table>
<thead>
<tr>
<th>Rank</th>
<th>Category</th>
<th>Business</th>
<th>Stars</th>
<th>Avg. Stars</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Latin American</td>
<td>Salvadoreno Restaurant</td>
<td>4.0</td>
<td>3.70</td>
</tr>
<tr>
<td>2</td>
<td>Gluten Free</td>
<td>P.F. Chang's China Bistro</td>
<td>3.5</td>
<td>3.68</td>
</tr>
<tr>
<td>3</td>
<td>Hobby Shops</td>
<td>Make Meaning</td>
<td>4.5</td>
<td>3.68</td>
</tr>
<tr>
<td>4</td>
<td>Mass Media</td>
<td>KJZZ 91.5FM</td>
<td>4.0</td>
<td>3.70</td>
</tr>
<tr>
<td>5</td>
<td>Yoga</td>
<td>Sutra Midtown</td>
<td>4.5</td>
<td>3.69</td>
</tr>
<tr>
<td>6</td>
<td>Churches</td>
<td>St Andrew the Apostle Catholic Church</td>
<td>4.5</td>
<td>3.71</td>
</tr>
<tr>
<td>7</td>
<td>Art Galleries</td>
<td>Sette Lisa Gallery</td>
<td>4.5</td>
<td>3.69</td>
</tr>
<tr>
<td>8</td>
<td>Libraries</td>
<td>Cholla Branch Library</td>
<td>4.0</td>
<td>3.70</td>
</tr>
<tr>
<td>9</td>
<td>Religious Organizations</td>
<td>St Andrew the Apostle Catholic Church</td>
<td>4.5</td>
<td>3.70</td>
</tr>
<tr>
<td>10</td>
<td>Wickenburg(city)</td>
<td>A Taste of the Caribbean</td>
<td>4.0</td>
<td>3.68</td>
</tr>
</tbody>
</table>

Top recovered business categories with lowest error and businesses with highest membership weights in $\hat{\Pi}$. Average stars denote the average review stars of estimated members in that category.

Ground truth graph and estimated graph
Outline

1. Introduction
2. Community Model
   - Tensor Forms for Stochastic Block Model
   - Extension to Mixed Membership Model
3. Learning Algorithm
   - Pre-processing
   - Stochastic Tensor Gradient Descent
   - Post-processing
4. GPU Implementation
5. Results
   - Validation Methods
   - Real-world Datasets
6. Conclusion and Future Work
Conclusion

- Efficient tensor-based learning algorithm for detecting overlapping communities.
- Stochastic method.
- Fast GPU implementation.
- Hypothesis testing for validation against ground truth.
- Good results on real-world datasets.
Tensor-based Learning Algorithms: Theory & Practice

- **Proof of convergence**: extension of the ODE method to prove convergence of stochastic approximation algorithms for tensors.
- **Over-complete tensor case**: tensor decomposition when $k \gg n$.
- **Out-of-core algorithms**: extension to millions of nodes using blocked BLAS operations, sparse matrix algorithms and using data movement involving hard disk, main memory and GPU memory.
- **Image datasets**: learn hidden features; comparison with deep learning.