Collective Classification

ACMS 80770: Deep Learning with Graphs

Instructor: Navid Shervani-Tabar

Department of Applied and Comp Math and Stats

UNIVERSITY OF NOTRE DAME

Relational Data

- When dealing with structured data represented on graphs, we often are interested in **labeling nodes** given a predefined set of classes.
  - Predicting node **attributes**:
    - political affiliation based on online activity and interactions
  - Modeling **influence** of nodes on each other:
    - Spread of infectious disease.

Relational Data

- When dealing with structured data represented on graphs, we often are interested in **labeling nodes** given a predefined set of classes.
  - Predicting node **attributes**:
    - political affiliation based on online activity and interactions
  - Modeling **influence** of nodes on each other:
    - Spread of infectious disease.
- The available information in such problems is often the **attributes** of the nodes in the graph and the **relation** between nodes provided by the graph structure.

Relational Data

- In **classical** machine learning, the attributes of each data point are used as the input to perform classification.
- These methods are designed for **flat data**, where all data points are identically-structured.

Relational Data

- In **classical** machine learning, the attributes of each data point are used as the input to perform classification.
- These methods are designed for **flat data**, where all data points are identically-structured.
- However, some data are inherently **relational**.
- Classical classification methods **look over** the relations between data points.

Relational Data

- In **classical** machine learning, the attributes of each data point are used as the input to perform classification.
- These methods are designed for **flat data**, where all data points are identically-structured.
- However, some data are inherently **relational**.
- Classical classification methods **look over** the relations between data points.
- There are two general trends seen in graph structured data
  - **Homophily**: Nodes tend to associate with similar others.
  - **Influence**: Nodes that are connected tend to influence each other.

Collective Classification

- In general, there are three types of information in relational data that we can **exploit** to perform node classification

Collective Classification

- In general, there are three types of information in relational data that we can **exploit** to perform node classification
  - Correlation between the label y_i of node Y_i and it's attributes $\boldsymbol{a}_i$.

Collective Classification

- In general, there are three types of information in relational data that we can **exploit** to perform node classification
  - Correlation between the label y_i of node Y_i and it's attributes $\boldsymbol{a}_i$.
  - Correlation between the label y_i of node Y_i and the observed labels x_j and attributes $\boldsymbol{a}_j$ of its neighboring nodes $X_j \in N(Y_i)$.

Collective Classification

- In general, there are three types of information in relational data that we can **exploit** to perform node classification
  - Correlation between the label y_i of node Y_i and it's attributes $\boldsymbol{a}_i$.
  - Correlation between the label y_i of node Y_i and the observed labels x_j and attributes $\boldsymbol{a}_j$ of its neighboring nodes $X_j \in N(Y_i)$.
  - Correlation between the label y_i of node Y_i and the unobserved labels y_j of its neighboring nodes $Y_j \in N(Y_i)$.

Collective Classification

- In general, there are three types of information in relational data that we can **exploit** to perform node classification
  - Correlation between the label y_i of node Y_i and it's attributes $\boldsymbol{a}_i$.
  - Correlation between the label y_i of node Y_i and the observed labels x_j and attributes $\boldsymbol{a}_j$ of its neighboring nodes $X_j \in N(Y_i)$.
  - Correlation between the label y_i of node Y_i and the unobserved labels y_j of its neighboring nodes $Y_j \in N(Y_i)$.
- **Collective classification** refers to the classification methods that leverage the relational information between data points for the task of label assignment to these data.

Collective Classification

- Let $G = (V, E)$ be a partially labeled data.
- V comprises two disjoint subsets, $\mathcal{X}$ and $\mathcal{Y}$, denoting labeled and unlabeled data, respectively, such that

$$\mathcal{X} \cup \mathcal{Y} = V$$

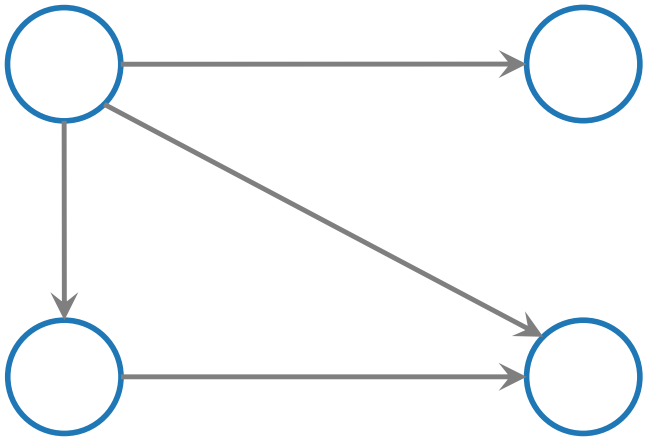
and

$$\mathcal{X} \cap \mathcal{Y} = \emptyset$$

- Given a set of labels $L = \{l_1, \ldots, l_K\}$, we aim to assign each unlabeled node $Y_i \in \mathcal{Y}$ with a label $y_i \in L$.

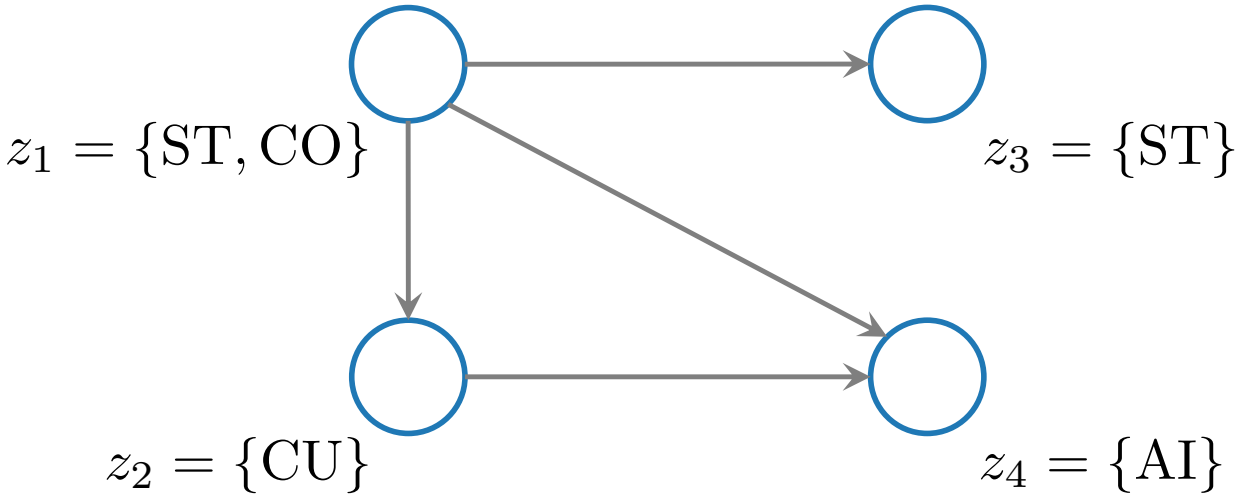
Collective Classification

- Consider the graph representing the connection between webpages:



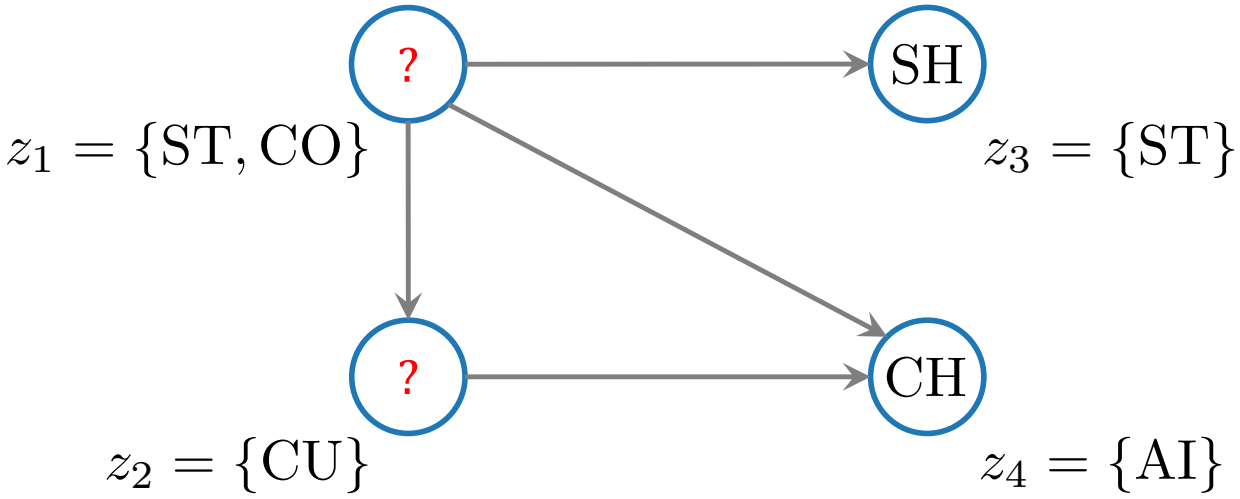
Collective Classification

- Consider the graph representing the connection between webpages:
  - We have a set of nodes with known attributes.



Collective Classification

- Consider the graph representing the connection between webpages.
  - We have a set of nodes with known attributes
  - We have labels for some of the nodes.
- The goal is to predict labels for unlabeled nodes



Collective Classification

- Collective classification algorithms can generally be divided into two categories
  - **Local** conditional classifiers
    - Iterative classification
    - Gibbs sampling
  - **Global** optimizers
    - Loopy belief propagation

Iterative Classification

- Iterative classification methods uses a **local classifier** to iteratively update the labels of unlabeled nodes Y_i in the graph G.
- The classifier is **local** in the sense that it uses the information from **neighborhood** $N(Y_i)$ to predict label y_i of node Y_i.
- Mathematically put

$$y_i = f(\boldsymbol{a}_i)$$

where $\boldsymbol{a}_i$ is the neighborhood **information** for node Y_i.

Iterative Classification

- Iterative classification methods uses a **local classifier** to iteratively update the labels of unlabeled nodes Y_i in the graph G.
- The classifier is **local** in the sense that it uses the information from **neighborhood** $N(Y_i)$ to predict label y_i of node Y_i.
- Mathematically put

$$y_i = f(\boldsymbol{a}_i)$$

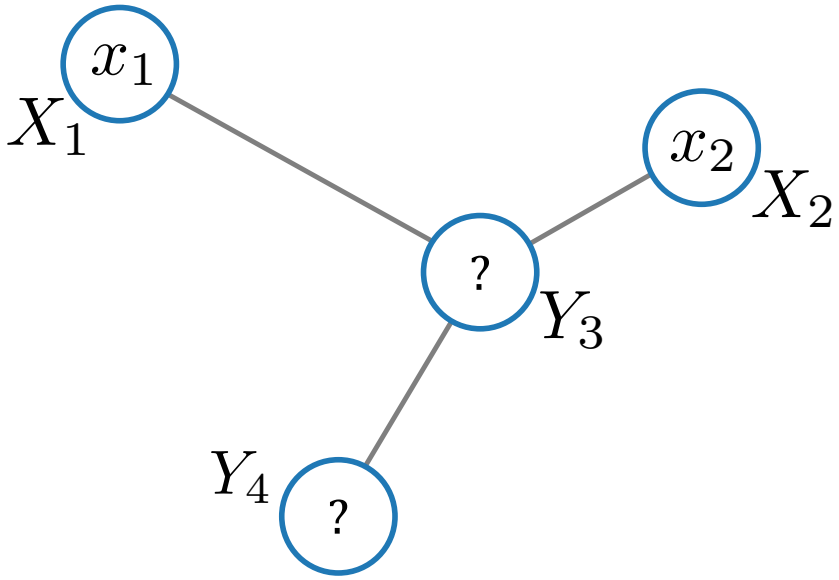
  where $\boldsymbol{a}_i$ is the neighborhood **information** for node Y_i.
- The local neighborhood information are represented as a **summary vector** using an aggregation operator.
- *E.g.,* Count, Proportion, Exist, minimum, maximum, and mode.

Iterative Classification

- We use the relational information vector $\boldsymbol{a}_i$ constructed by the known neighbors of Y_i

$$\boldsymbol{a}_i = \texttt{aggregate}\left(\{z_i, x_j \mid X_j \in N\left(Y_i\right) \cap \mathcal{X}\}\right)$$

to **initialize label** y_i using a local classifier f.



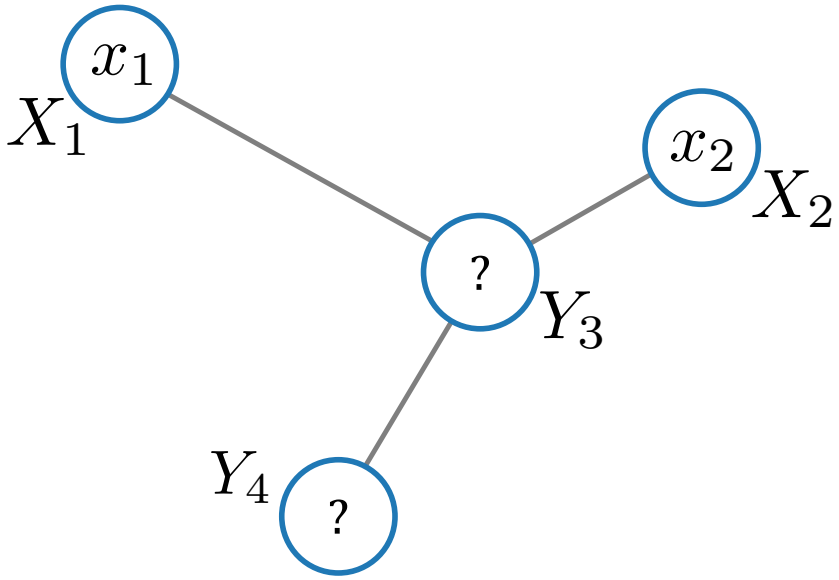
Iterative Classification

- We use the relational information vector $\boldsymbol{a}_i$ constructed by the known neighbors of Y_i

$$\boldsymbol{a}_i = \texttt{aggregate}\left(\{z_i, x_j \mid X_j \in N\left(Y_i\right) \cap \mathcal{X}\}\right)$$

  to **initialize label** y_i using a local classifier f.

- This step is referred to as **bootstrapping**.



Iterative Classification

- We use the relational information vector $\boldsymbol{a}_i$ constructed by the known neighbors of Y_i

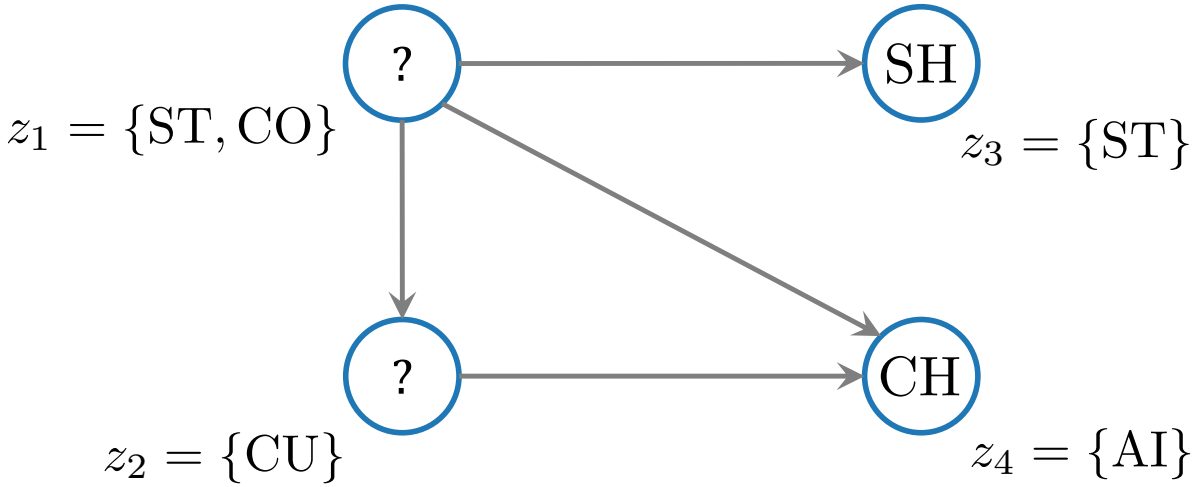
$$\boldsymbol{a}_i = \texttt{aggregate}\left(\{z_i, x_j \mid X_j \in N\left(Y_i\right) \cap \mathcal{X}\}\right)$$

  to **initialize label** y_i using a local classifier f.

- This step is referred to as **bootstrapping**.
- Next, we define an **ordering** $\mathcal{O}$ over unlabeled nodes.
- We visit unlabeled nodes Y_i in the order given by $\mathcal{O}$ to update their labels y_i using the local classifier f based on the information vector $\boldsymbol{a}_i$ constructed from the **latest label update**.
- We repeat this until the labels are **stable** or the **maximum** number of iterations is reached.

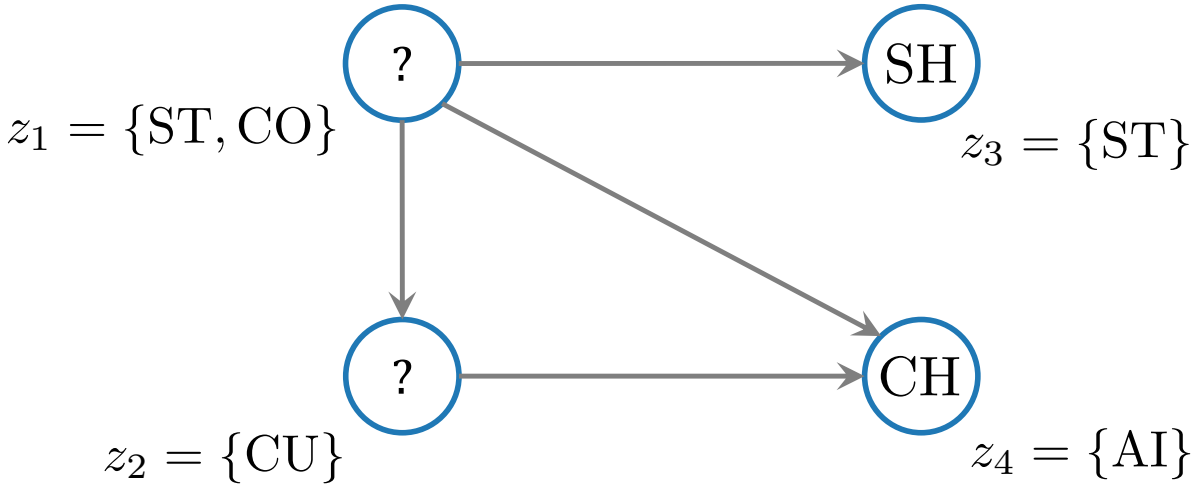
Iterative Classification

- Bootstrapping



Iterative Classification

- Bootstrapping

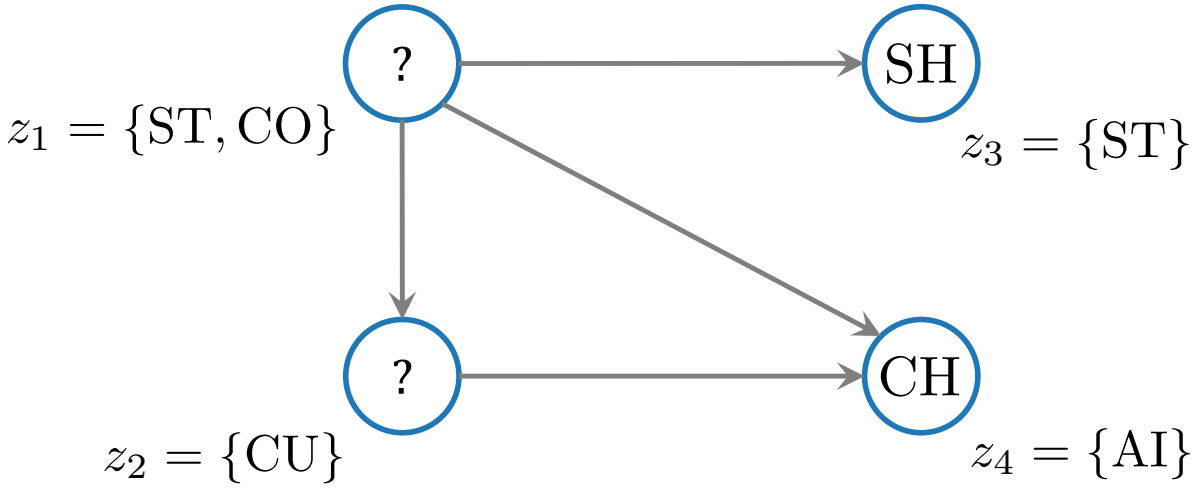


$$\boldsymbol{a}_1^{(0)} = (\quad\quad\quad\quad) \oplus \texttt{aggregate}(\quad\quad\quad\quad)$$

ST CO CU AI

Iterative Classification

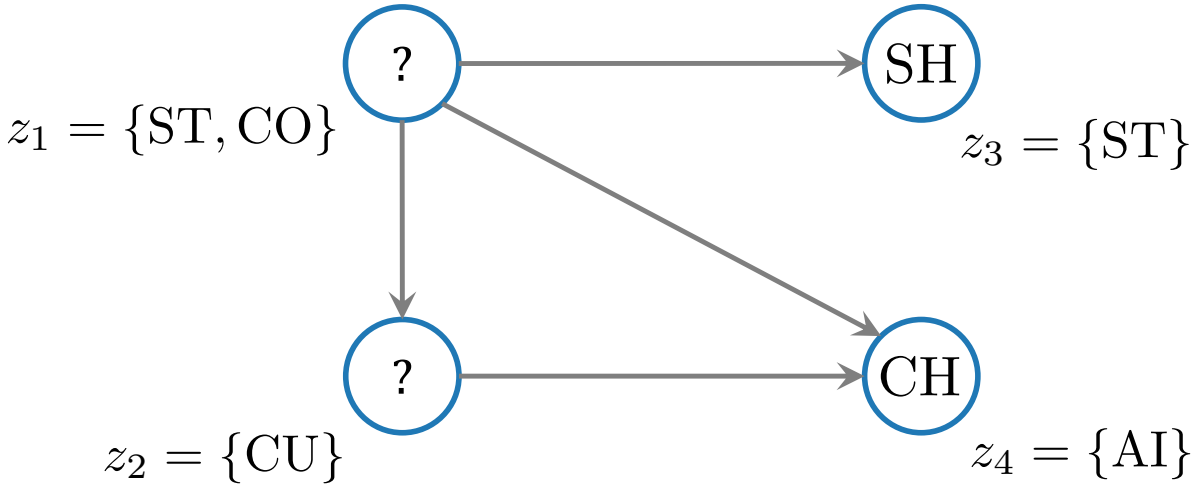
- Bootstrapping



$$\boldsymbol{a}_1^{(0)} = \overset{\text{ST}\ \ \text{CO}\ \ \text{CU}\ \ \text{AI}}{(1,\ 1,\ 0,\ 0)} \oplus \texttt{aggregate}(\qquad\qquad)$$

Iterative Classification

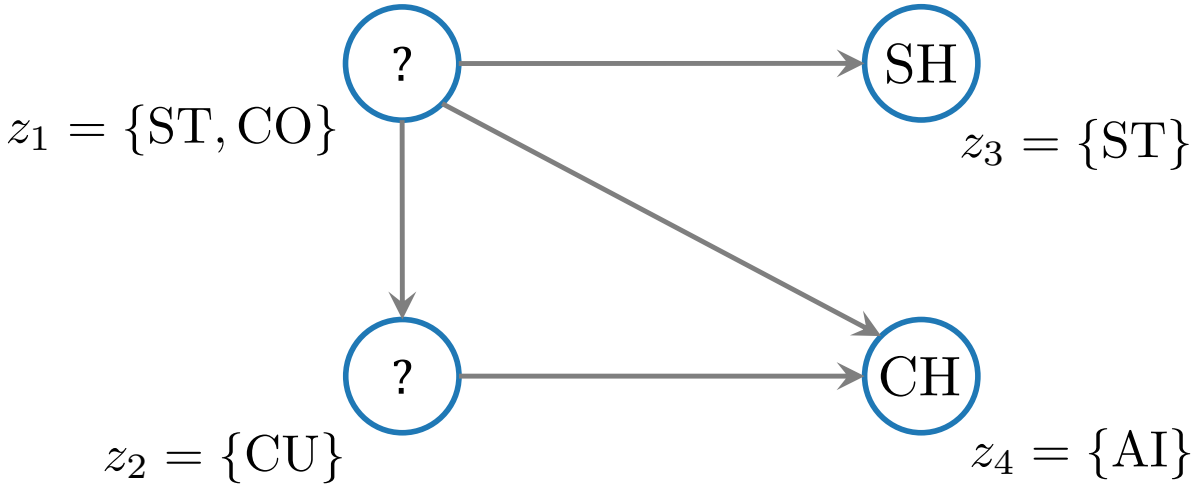
- Bootstrapping



$$\boldsymbol{a}_1^{(0)} = \overset{\text{ST}\ \ \text{CO}\ \ \text{CU}\ \ \text{AI}}{(1,\ 1,\ 0,\ 0)} \oplus \texttt{aggregate}(y_2, x_3, x_4)$$

Iterative Classification

- Bootstrapping

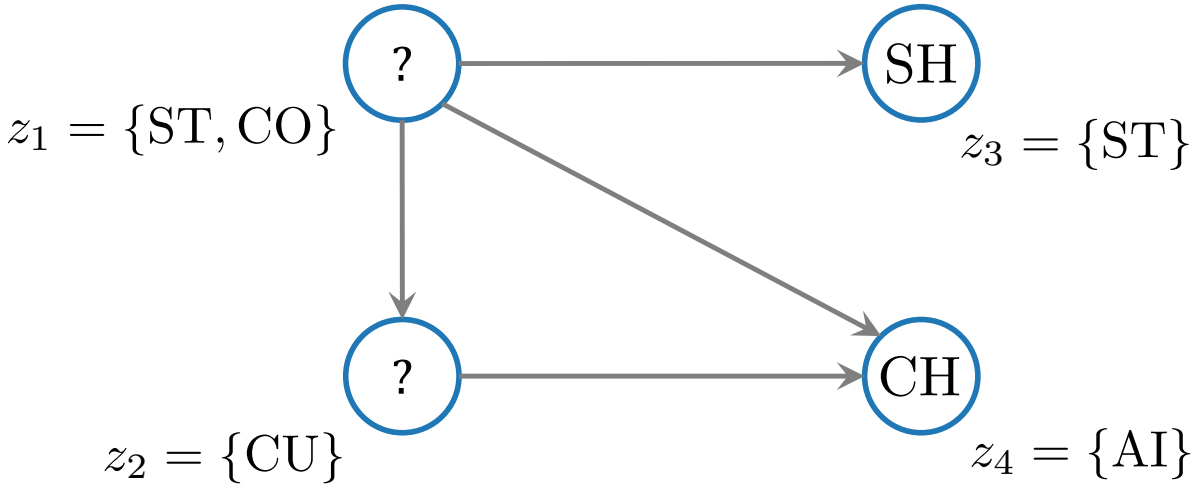


$$\boldsymbol{a}_1^{(0)} = \overset{\text{ST CO CU AI}}{(1,\ 1,\ 0,\ 0)} \oplus \texttt{aggregate}(y_2, x_3, x_4)$$

$$\overset{\text{SH CH}}{(1,\ 1)}$$

Iterative Classification

- Bootstrapping



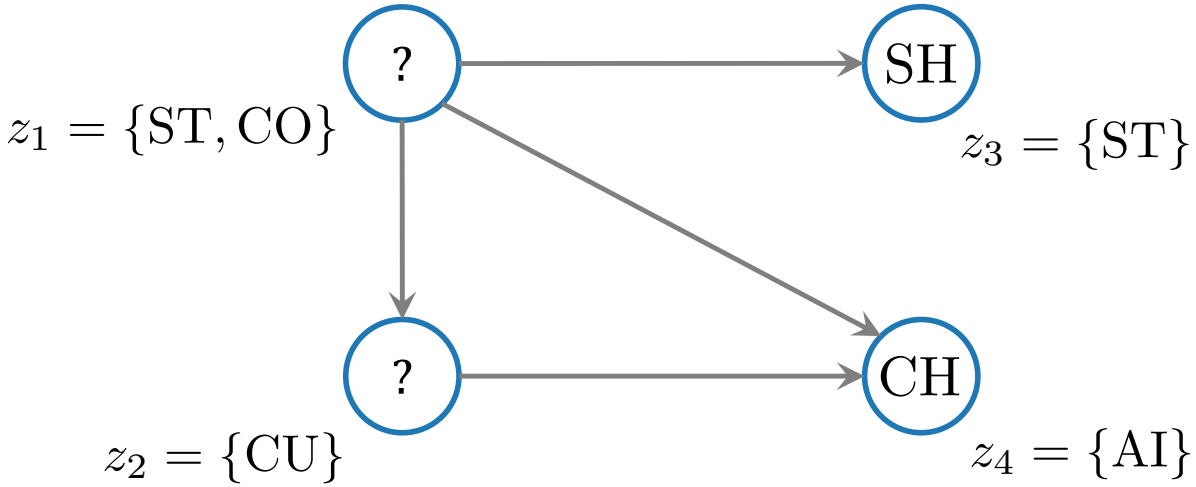
$$\boldsymbol{a}_1^{(0)} = \overset{\text{ST CO CU AI}}{(1,\ 1,\ 0,\ 0)} \oplus \texttt{aggregate}(y_2, x_3, x_4)$$

$$\boldsymbol{a}_1^{(0)} = (1,\ 1,\ 0,\ 0,\ 1,\ 1)$$

$$\overset{\text{SH CH}}{(1,\ 1)}$$

Iterative Classification

- Bootstrapping



$$\boldsymbol{a}_1^{(0)} = \overset{\text{ST CO CU AI}}{(1,\ 1,\ 0,\ 0)} \oplus \texttt{aggregate}(y_2, x_3, x_4)$$

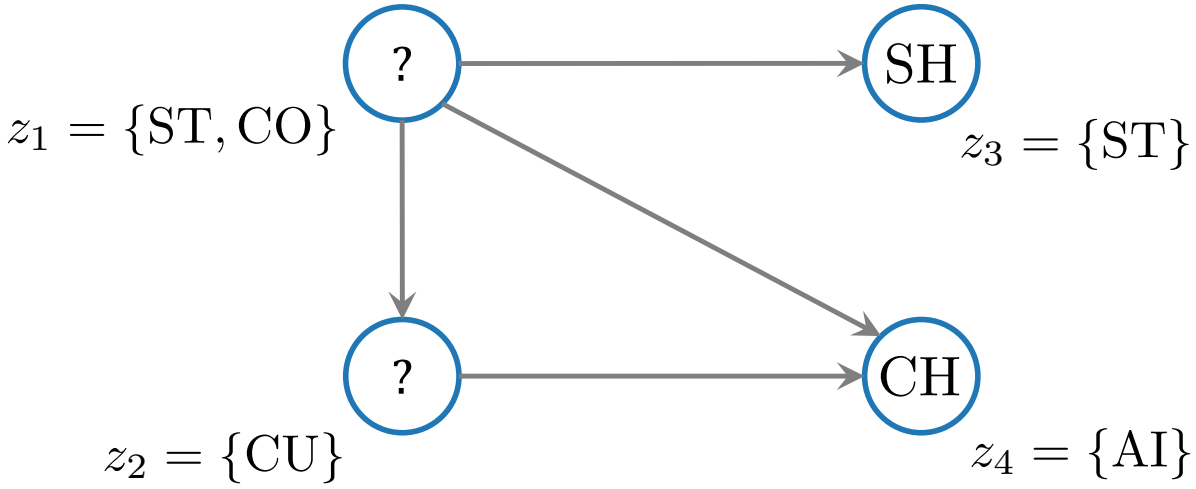
$$\boldsymbol{a}_1^{(0)} = (1,\ 1,\ 0,\ 0,\ 1,\ 1)$$

$$\overset{\text{SH CH}}{(1,\ 1)}$$

$$\boldsymbol{a}_2^{(0)} = (0,\ 0,\ 1,\ 0) \oplus \texttt{aggregate}(y_1, x_4) = (0,\ 0,\ 1,\ 0,\ 0,\ 1)$$

Iterative Classification

- Bootstrapping

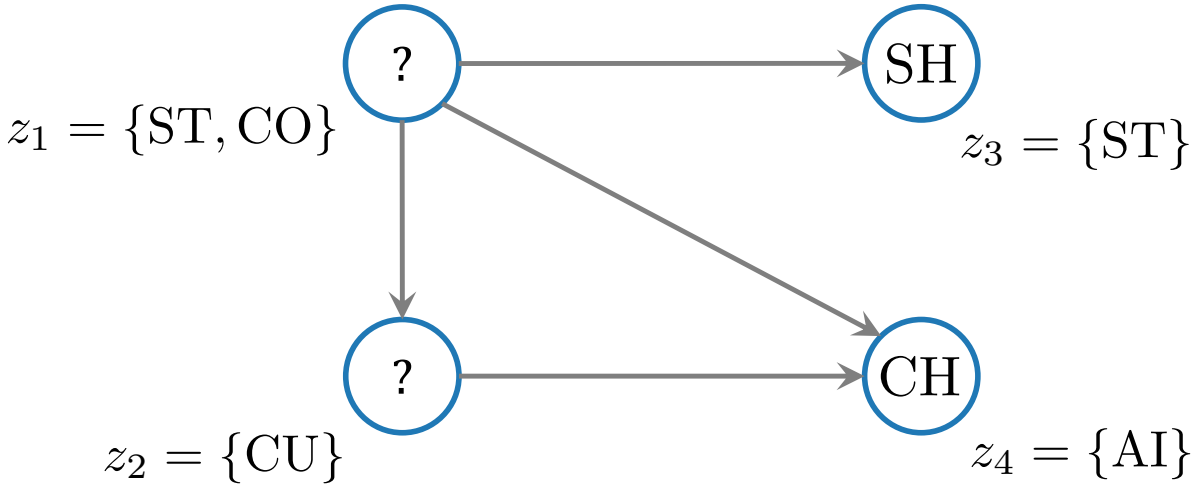


$$\boldsymbol{a}_1^{(0)} = (1,\ 1,\ 0,\ 0,\ 1,\ 1) \qquad y_1 = f(\boldsymbol{a}_1)$$

$$\boldsymbol{a}_2^{(0)} = (0,\ 0,\ 1,\ 0,\ 0,\ 1) \qquad y_2 = f(\boldsymbol{a}_2)$$

Iterative Classification

- Bootstrapping

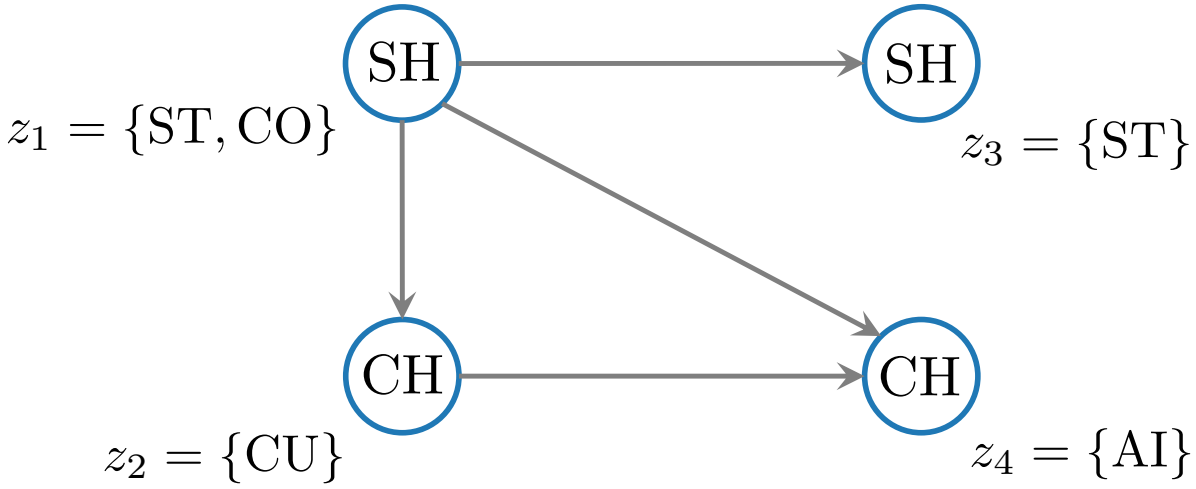


$$\boldsymbol{a}_1^{(0)} = (1,\ 1,\ 0,\ 0,\ 1,\ 1) \qquad y_1 = f(\boldsymbol{a}_1) \longrightarrow \text{SH}$$

$$\boldsymbol{a}_2^{(0)} = (0,\ 0,\ 1,\ 0,\ 0,\ 1) \qquad y_2 = f(\boldsymbol{a}_2) \longrightarrow \text{CH}$$

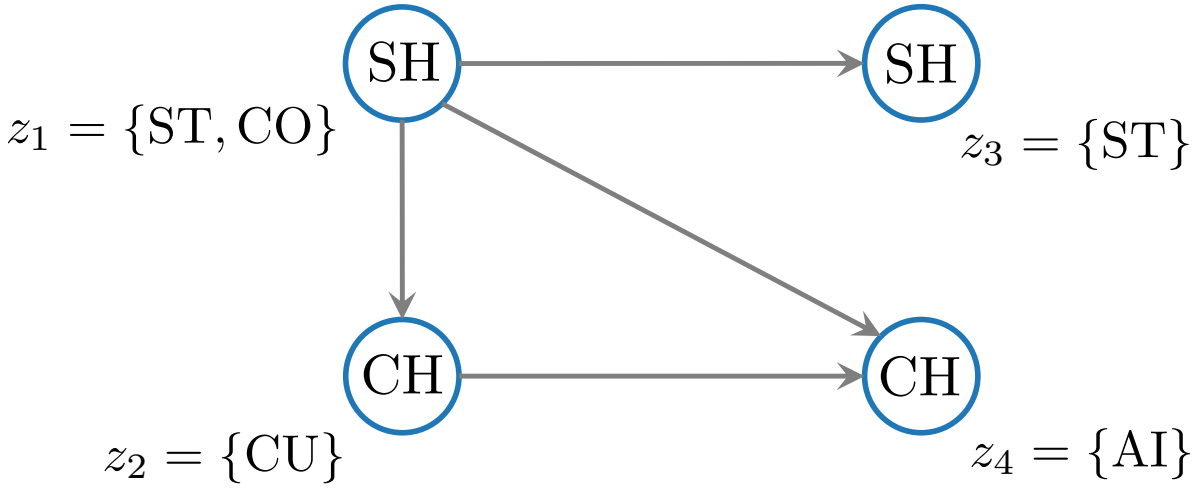
Iterative Classification

- Iteration 1:



Iterative Classification

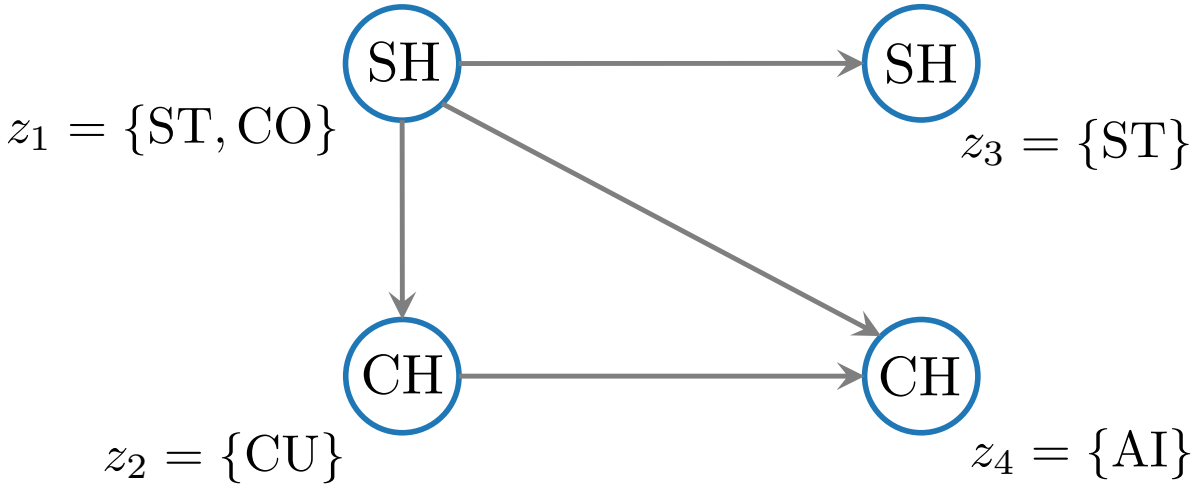
- Iteration 1:



$$\mathcal{O} = \{Y_1, Y_2\}$$

Iterative Classification

- Iteration 1:



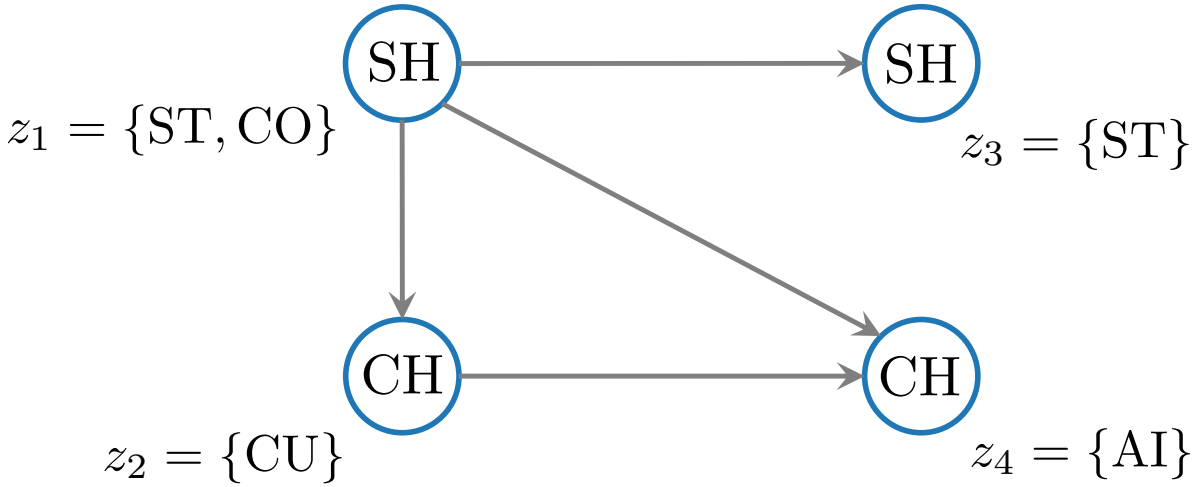
$$\mathcal{O} = \{Y_1, Y_2\}$$

$$\boldsymbol{a}_1^{(1)} = (1,\ 1,\ 0,\ 0) \oplus \texttt{aggregate}(y_2, x_3, x_4)$$

$$\boldsymbol{a}_2^{(1)} = (0,\ 0,\ 1,\ 0) \oplus \texttt{aggregate}(y_1, x_4)$$

Iterative Classification

- Iteration 1:



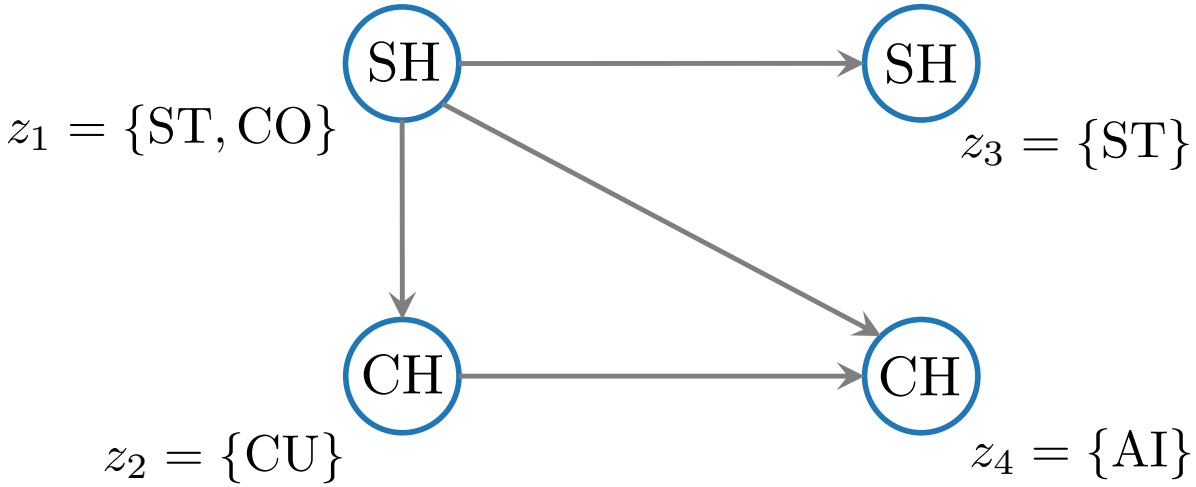
$$\mathcal{O} = \{Y_1, Y_2\}$$

$$\boldsymbol{a}_1^{(1)} = (1,\ 1,\ 0,\ 0) \oplus \texttt{aggregate}(y_2, x_3, x_4) = (1,\ 1,\ 0,\ 0,\ 1,\ 2)$$

$$\boldsymbol{a}_2^{(1)} = (0,\ 0,\ 1,\ 0) \oplus \texttt{aggregate}(y_1, x_4) = (0,\ 0,\ 1,\ 0,\ 1,\ 1)$$

Iterative Classification

- Iteration 1:



$$\boldsymbol{a}_1^{(1)} = (1,\ 1,\ 0,\ 0,\ 1,\ 2) \qquad y_1 = f(\boldsymbol{a}_1) \longrightarrow \text{SH}$$

$$\boldsymbol{a}_2^{(1)} = (0,\ 0,\ 1,\ 0,\ 1,\ 1) \qquad y_2 = f(\boldsymbol{a}_2) \longrightarrow \text{CH}$$

Gibbs Sampling

- Another local method for collective classification is using **Gibbs sampling** algorithm.
- Similar to iterative classifier, Gibbs sampling needs access to a **local classifier** f.
- This classifier can be **trained** using the **labeled data** $\mathcal{X}$ on the graph G.
- The trained classifier f is then used to **initialize the labels** for unlabeled nodes $Y_i \in \mathcal{Y}$ of the graph.

Gibbs Sampling

- Another local method for collective classification is using **Gibbs sampling** algorithm.
- Similar to iterative classifier, Gibbs sampling needs access to a **local classifier** f.
- This classifier can be **trained** using the **labeled data** $\mathcal{X}$ on the graph G.
- The trained classifier f is then used to **initialize the labels** for unlabeled nodes $Y_i \in \mathcal{Y}$ of the graph.
- In Gibbs sampling, we assume that the local classifier f approximates the **conditional probability distribution** of labels y_i given information from neighbor nodes $N(Y_i)$.

Gibbs Sampling

- After initializing the unknown labels y_i in the bootstrapping phase, labels y_i are updated according to some ordering $\mathcal{O}$ for a **burn-in period**.

Gibbs Sampling

- After initializing the unknown labels y_i in the bootstrapping phase, labels y_i are updated according to some ordering $\mathcal{O}$ for a **burn-in period**.
- Then, given the node order $\mathcal{O}$ over nodes $\mathcal{Y}$, we **sample** labels from the conditional probability distribution.

Gibbs Sampling

- After initializing the unknown labels y_i in the bootstrapping phase, labels y_i are updated according to some ordering $\mathcal{O}$ for a **burn-in period**.
- Then, given the node order $\mathcal{O}$ over nodes $\mathcal{Y}$, we **sample** labels from the conditional probability distribution.
- Then, we **count** the total number of samples $c_{k,i}$ of each label $l_k \in \mathrm{L}$ for each node $Y_i \in \mathcal{Y}$.

Gibbs Sampling

- After initializing the unknown labels y_i in the bootstrapping phase, labels y_i are updated according to some ordering $\mathcal{O}$ for a **burn-in period**.
- Then, given the node order $\mathcal{O}$ over nodes $\mathcal{Y}$, we **sample** labels from the conditional probability distribution.
- Then, we **count** the total number of samples $c_{k,i}$ of each label $l_k \in \mathrm{L}$ for each node $Y_i \in \mathcal{Y}$.
- After collecting T samples, we assign the **most sampled** label to the corresponding node Y_i.

$$y_i = \arg\max_k c_{i,k}$$

Gibbs Sampling

- There are a number of **downsides** to using Gibbs sampling
  - The method is **slow**.
  - Convergence tests are **expensive**.

Global Object Functions

- There are a number of **downsides** to using Gibbs sampling
  - The method is **slow**.
  - Convergence tests are **expensive**.

- Another approach is to optimize a **global object functions** to recover unlabeled data.
- These approaches simultaneously work on the **whole graph** as opposed to using separate local attribute vectors.

Global Object Functions

- There are a number of **downsides** to using Gibbs sampling
  - The method is **slow**.
  - Convergence tests are **expensive**.

- Another approach is to optimize a **global object functions** to recover unlabeled data.
- These approaches simultaneously work on the **whole graph** as opposed to using separate local attribute vectors.
- In these methods, we represent the problem as an **undirected probabilistic graphical model**, also called a Markov random field or Markov network.

Markov Random Field

- A **pairwise Markov random field** is an undirected graph, in which:
  - Each node represents a **random variable**.
  - Each edge represents a **correlation** between two random variables.

Markov Random Field

- A **pairwise Markov random field** is an undirected graph, in which:
  - Each node represents a **random variable**.
  - Each edge represents a **correlation** between two random variables.
- The nature of correlations are quantified using **clique potentials**.
- Clique potentials are functions that map **joint assignments** on a set of nodes to how **favorable** these assignment are.

Markov Random Field

- The set of **clique potentials** Ψ in the model **map** unobserved nodes or pairs containing at least **one unobserved** node to a non-negative real number.

Markov Random Field

- The set of **clique potentials** Ψ in the model **map** unobserved nodes or pairs containing at least **one unobserved** node to a non-negative real number.
  - For each variable Y_i,

$$\psi_i : L \rightarrow \mathbb{R}^{\geq 0}$$

Markov Random Field

- The set of **clique potentials** Ψ in the model **map** unobserved nodes or pairs containing at least **one unobserved** node to a non-negative real number.
  - For each variable Y_i,

$$\psi_i : L \rightarrow \mathbb{R}^{\geq 0}$$

  - For each pair of correlated variables (Y_i, X_j),

$$\psi_{ij} : L \rightarrow \mathbb{R}^{\geq 0}$$

where $Y_i \in \mathcal{Y}$ and $X_j \in N(Y_i) \cap \mathcal{X}$.

Markov Random Field

- The set of **clique potentials** Ψ in the model **map** unobserved nodes or pairs containing at least **one unobserved** node to a non-negative real number.
  - For each variable Y_i,

$$\psi_i : L \rightarrow \mathbb{R}^{\geq 0}$$

  - For each pair of correlated variables (Y_i, X_j),

$$\psi_{ij} : L \rightarrow \mathbb{R}^{\geq 0}$$

    where $Y_i \in \mathcal{Y}$ and $X_j \in N(Y_i) \cap \mathcal{X}$.

  - For each pair of correlated variables (Y_i, Y_j),

$$\psi_{ij} : L \times L \rightarrow \mathbb{R}^{\geq 0}$$

    where $Y_i, Y_j \in \mathcal{Y}$.

Markov Random Field

- A pairwise Markov random field is defined as the pair $\langle G, \Psi \rangle$, where G is an **undirected** graph and Ψ is the set of **clique potentials** corresponding to it.
- Then, a **conditional probability** distribution

$$p(\mathbf{y} \mid \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{Y_i \in \mathcal{Y}} \phi_i \left(y_i\right) \prod_{(Y_i, Y_j) \in E} \psi_{ij} \left(y_i, y_j\right)$$

is associated with the pairwise Markov random field, with $Z(\boldsymbol{x})$ denoting the normalization constant and

$$\phi_i \left(y_i\right) = \psi_i \left(y_i\right) \prod_{(Y_i, X_j) \in E} \psi_{ij} \left(y_i\right)$$

Loopy Belief Propagation

- Exact inference of computing the distribution over label variables for a general graph **topology** is intractable.
- We use an approximate scheme called **loopy belief propagation** to do inference.
- Loopy belief propagation is an approximate inference based on **message passing** between variables.
- In current problem, random variables Y_i send their belief about the unlabeled neighboring node Y_j's label as a message $m_{i \to j}$ from Y_i to Y_j .
- Therefore, $m_{i \to j}$ is simply a mapping $L \to \mathbb{R}^{\geq 0}$.

Loopy Belief Propagation

- **Applying** loopy belief propagation to pairwise Markov random field $\langle G, \Psi \rangle$ can be expressed in two steps
  - Sending **messages**

$$m_{i \to j}\left(y_j\right) = \alpha \sum_{y_i \in L} \psi_{ij}\left(y_i, y_j\right) \phi_i\left(y_i\right) \prod_{Y_k \in \mathcal{N}_i \cap \mathcal{Y} \setminus Y_j} m_{k \to i}\left(y_i\right)$$

  - Computing **belief**

$$b_i\left(y_i\right) = \alpha \phi_i\left(y_i\right) \prod_{Y_j \in \mathcal{N}_i \cap \mathcal{Y}} m_{j \to i}\left(y_i\right)$$

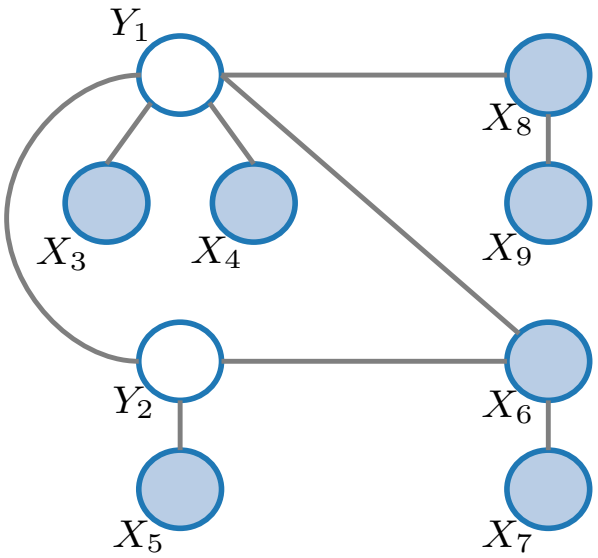
where alpha is a normalization factor, ensuring that

$$\sum_{y_j} m_{i \to j}\left(y_j\right) = 1$$

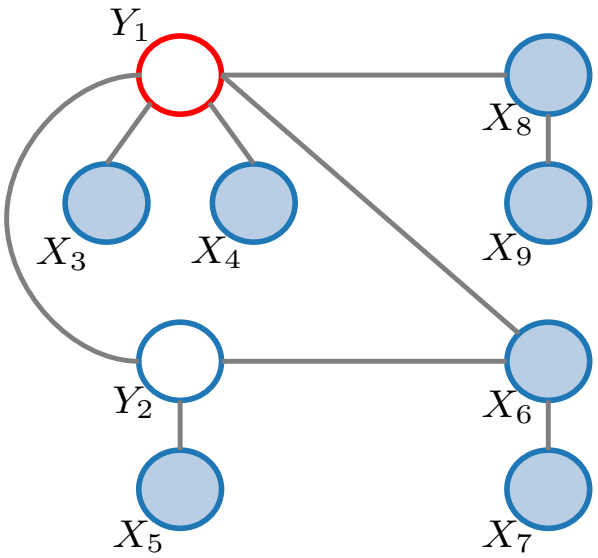
and

$$\sum_{y_i} b_i\left(y_i\right) = 1$$

Loopy Belief Propagation

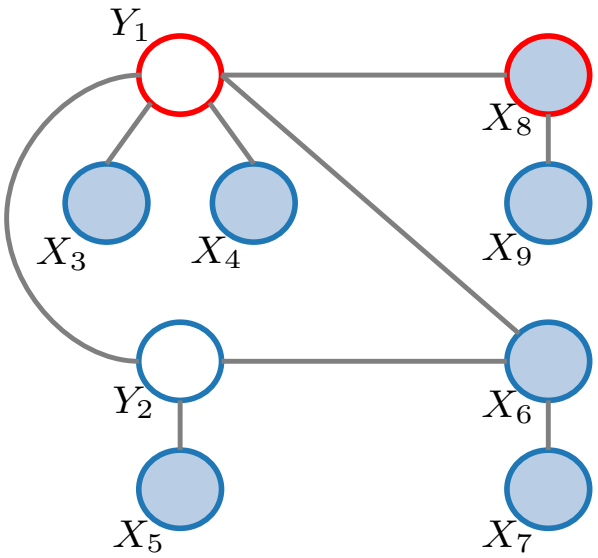


Loopy Belief Propagation



$$\begin{matrix} & \text{SH} & \text{CH} \\ \psi_1 = & (0.5, & 0.5) \\ \psi_2 = & (0.5, & 0.5) \end{matrix}$$

Loopy Belief Propagation



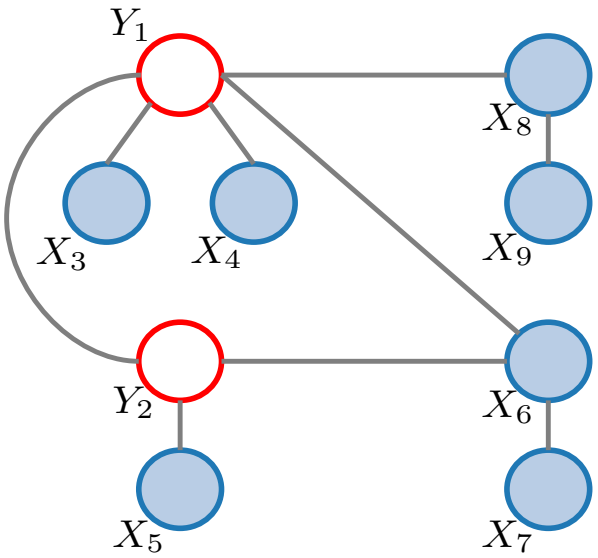
SH CH

$$\psi_1 = (0.5,\ 0.5)$$
$$\psi_2 = (0.5,\ 0.5)$$

SH CH

$$\psi_{13} = (0.6,\ 0.4)$$
$$\psi_{14} = (0.4,\ 0.6)$$
$$\psi_{16} = (0.1,\ 0.9)$$
$$\psi_{18} = (0.8,\ 0.2)$$
$$\psi_{25} = (0.1,\ 0.9)$$
$$\psi_{26} = (0.1,\ 0.9)$$

Loopy Belief Propagation

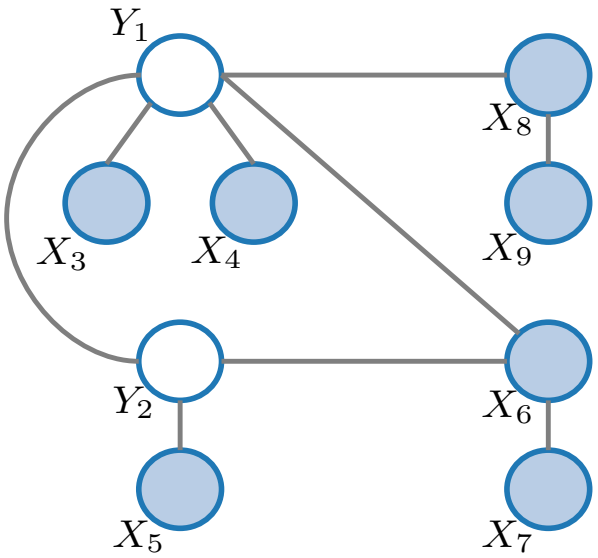


$$\begin{matrix} & \text{SH} & \text{CH} \\ \psi_1 = (0.5, & 0.5) \\ \psi_2 = (0.5, & 0.5) \end{matrix}$$

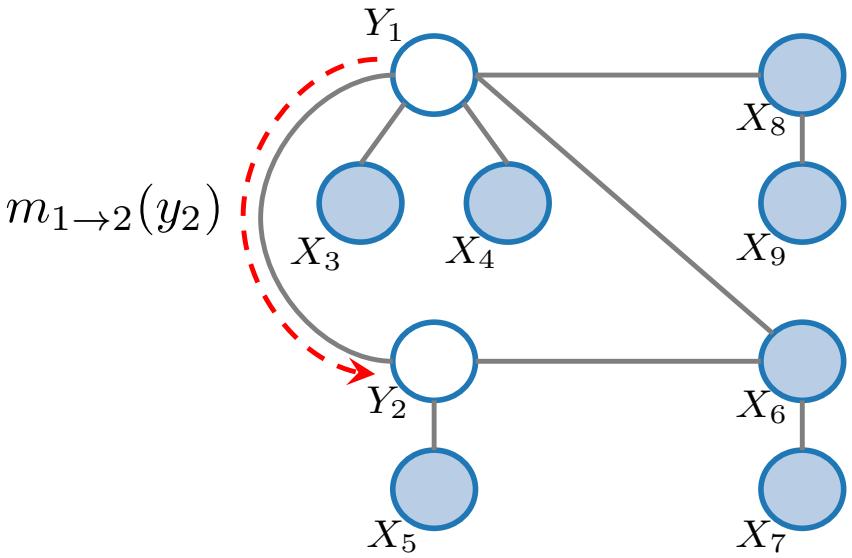
$$\begin{matrix} & \text{SH} & \text{CH} \\ \psi_{13} = (0.6, & 0.4) \\ \psi_{14} = (0.4, & 0.6) \\ \psi_{16} = (0.1, & 0.9) \\ \psi_{18} = (0.8, & 0.2) \\ \psi_{25} = (0.1, & 0.9) \\ \psi_{26} = (0.1, & 0.9) \end{matrix}$$

$$\begin{matrix} & \text{SH} & \text{SH} & \text{CH} & \text{CH} \\ & \text{SH} & \text{CH} & \text{SH} & \text{CH} \\ \psi_{12} = (0.9, & 0.1, & ,0.1, & 0.9) \end{matrix}$$

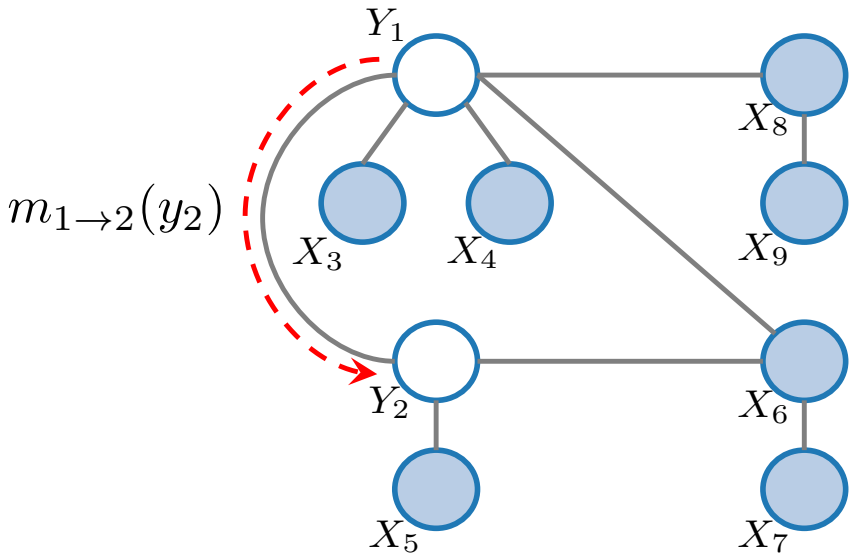
Loopy Belief Propagation



Loopy Belief Propagation

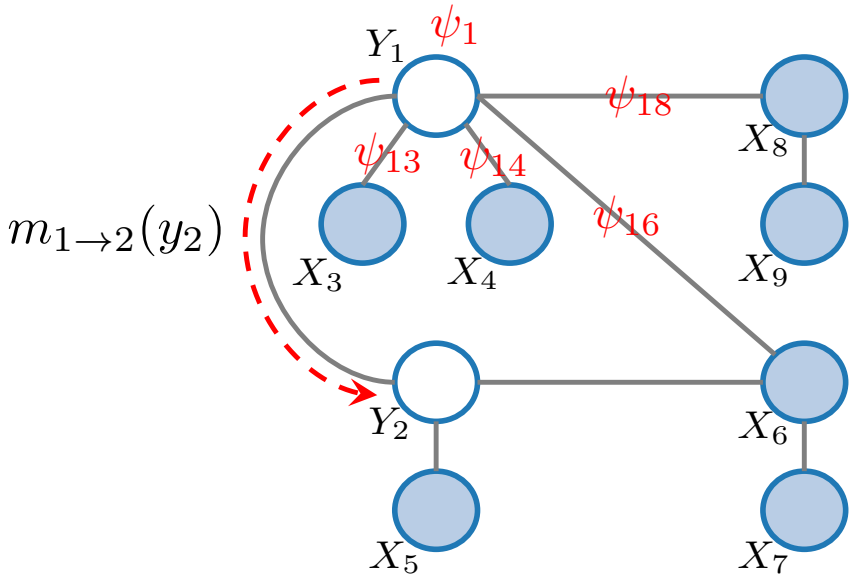


Loopy Belief Propagation



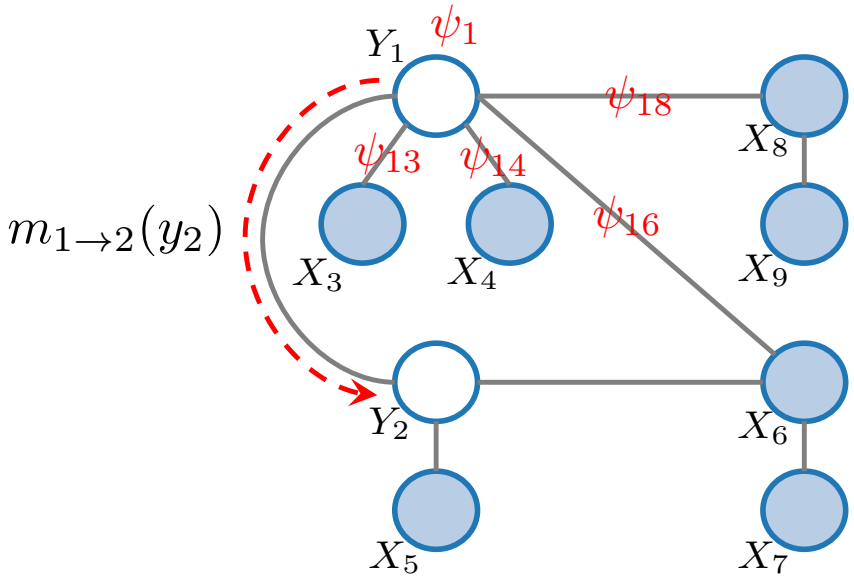
$$\phi_i = \psi_i \prod_{(Y_i, X_j) \in E} \psi_{ij}$$

Loopy Belief Propagation



$$\phi_i = \psi_i \prod_{(Y_i, X_j) \in E} \psi_{ij}$$

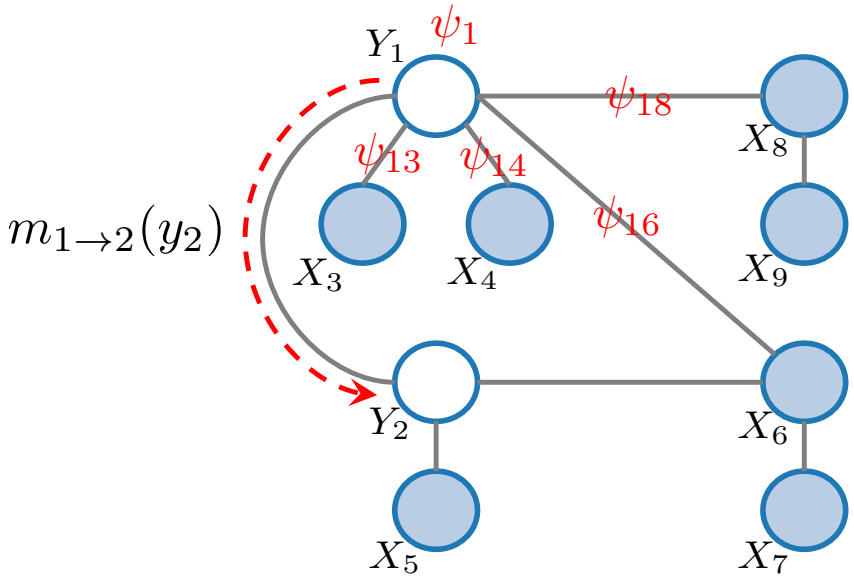
Loopy Belief Propagation



$$\psi_1 = (0.5,\ 0.5)$$
$$\psi_{13} = (0.6,\ 0.4)$$
$$\psi_{14} = (0.4,\ 0.6)$$
$$\psi_{16} = (0.1,\ 0.9)$$
$$\psi_{18} = (0.8,\ 0.2)$$

$$\phi_i = \psi_i \prod_{(Y_i, X_j) \in E} \psi_{ij} = \psi_1 \psi_{13} \psi_{14} \psi_{16} \psi_{18}$$

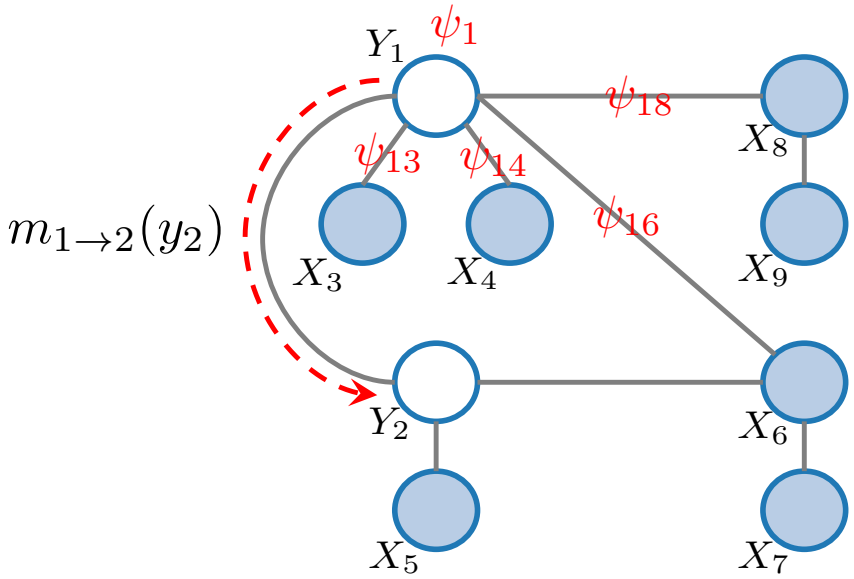
Loopy Belief Propagation



$$\psi_1 = (0.5,\ 0.5)$$
$$\psi_{13} = (0.6,\ 0.4)$$
$$\psi_{14} = (0.4,\ 0.6)$$
$$\psi_{16} = (0.1,\ 0.9)$$
$$\psi_{18} = (0.8,\ 0.2)$$

$$\phi_i = \psi_i \prod_{(Y_i, X_j) \in E} \psi_{ij} = \psi_1 \psi_{13} \psi_{14} \psi_{16} \psi_{18} = (0.0096,\ 0.0216)$$

Loopy Belief Propagation

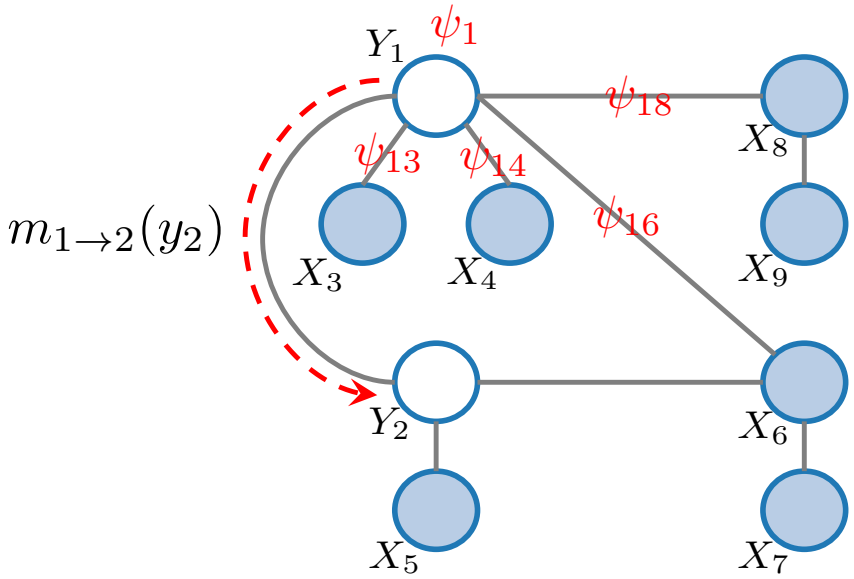


$$\psi_1 = (0.5,\ 0.5)$$
$$\psi_{13} = (0.6,\ 0.4)$$
$$\psi_{14} = (0.4,\ 0.6)$$
$$\psi_{16} = (0.1,\ 0.9)$$
$$\psi_{18} = (0.8,\ 0.2)$$

$$\phi_i = \psi_i \prod_{(Y_i, X_j) \in E} \psi_{ij} = \psi_1 \psi_{13} \psi_{14} \psi_{16} \psi_{18} = (0.0096,\ 0.0216)$$

$$m_{i \to j}(y_j) = \sum_{y_i \in L} \psi_{ij}(y_i, y_j)\, \phi_i(y_i) \prod_{Y_k \in \mathcal{N}_i \cap \mathcal{Y} \setminus Y_j} m_{k \to i}(y_i)$$

Loopy Belief Propagation



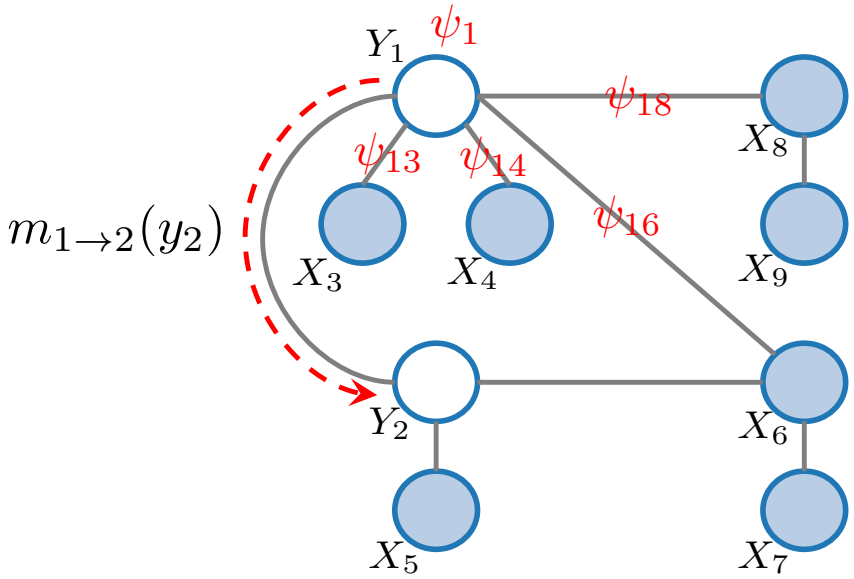
$$\psi_1 = (0.5,\ 0.5)$$
$$\psi_{13} = (0.6,\ 0.4)$$
$$\psi_{14} = (0.4,\ 0.6)$$
$$\psi_{16} = (0.1,\ 0.9)$$
$$\psi_{18} = (0.8,\ 0.2)$$

$$\phi_i = \psi_i \prod_{(Y_i, X_j) \in E} \psi_{ij} = \psi_1 \psi_{13} \psi_{14} \psi_{16} \psi_{18} = (0.0096,\ 0.0216)$$

$$m_{i\to j}(y_j) = \sum_{y_i \in L} \psi_{ij}(y_i, y_j)\, \phi_i(y_i) \prod_{Y_k \in \mathcal{N}_i \cap \mathcal{Y} \setminus Y_j} m_{k\to i}(y_i)$$

$$= \sum_{y_1 \in L} \psi_{12}(y_1, y_2)\, \phi_1(y_1) \prod_{Y_k \in \mathcal{N}_1 \cap \mathcal{Y} \setminus Y_2} m_{k\to 1}(y_1)$$

Loopy Belief Propagation



$$\psi_1 = (0.5,\ 0.5)$$
$$\psi_{13} = (0.6,\ 0.4)$$
$$\psi_{14} = (0.4,\ 0.6)$$
$$\psi_{16} = (0.1,\ 0.9)$$
$$\psi_{18} = (0.8,\ 0.2)$$

$$\phi_i = \psi_i \prod_{(Y_i, X_j) \in E} \psi_{ij} = \psi_1 \psi_{13} \psi_{14} \psi_{16} \psi_{18} = (0.0096,\ 0.0216)$$

$$m_{i\to j}(y_j) = \sum_{y_i \in L} \psi_{ij}(y_i, y_j)\, \phi_i(y_i) \prod_{Y_k \in \mathcal{N}_i \cap \mathcal{Y} \setminus Y_j} m_{k\to i}(y_i)$$

$$= \sum_{y_1 \in L} \psi_{12}(y_1, y_2)\, \phi_1(y_1) \prod_{Y_k \in \mathcal{N}_1 \cap \mathcal{Y} \setminus Y_2} m_{k\to 1}(y_1) = \sum_{y_1 \in L} \psi_{12}(y_1, y_2)\, \phi_1(y_1)$$

Summary

- Relational Data
- Collective classification
  - Local conditional classifiers
  - Global optimizers
- Iterative classification
- Bootstrapping
- Gibbs sampling
- Markov random field
- Clique potentials
- Loopy belief propagation
- Message passing