Collective classification in large scale networks

Jennifer Neville
Departments of Computer Science and Statistics
Purdue University
The data mining process

Network data is:
- heterogeneous and interdependent,
- partially observed/labeled,
- dynamic and/or non-stationary,
- and often drawn from a single network

...thus many traditional methods developed for i.i.d. data do not apply
Example predictive models for network domains

- **Email networks**: Predict organizational roles from communication patterns
- **Scientific networks**: Predict paper topics from properties of cited papers
- **Social networks**: Predict personal preferences from characteristics of friends
- **Gene/protein networks**: Predict protein function from interaction patterns
- **World wide web**: Predict content changes from properties of hyperlinked pages
- **Organizational networks**: Predict group effectiveness from communication patterns
Attribute prediction in networks
Network autocorrelation is ubiquitous

- **Marketing**
  - Product/service adoption among communicating customers (Domingos & Richardson ‘01, Hill et al ‘06)

- **Advertising**
  - On-line brand adv. (Provost et al. ‘09)

- **Fraud detection**
  - Fraud status of cellular customers who call common numbers (Fawcett & Provost ‘97, Cortes et al ‘01)
  - Fraud status of brokers who work at the same branch (Neville & Jensen ‘05)

- **Biology**
  - Functions of proteins located in together in cells (Neville & Jensen ‘02)
  - Tuberculosis infection among people in close contact (Getoor et al ‘01)

- **Movies**
  - Box-office receipts of movies made by the same studio (Jensen & Neville ‘02)

- **Web**
  - Topics of hyperlinked web pages (Chakrabarti et al ‘98, Taskar et al ‘02)

- **Business**
  - Industry categorization of corporations that share common boards members (Neville & Jensen ‘00)
  - Industry categorization of corporations that co-occur in news stories (Bernstein et al ‘03)

- **Citation analysis**
  - Topics of coreferent scientific papers (Taskar et al ‘01, Neville & Jensen ‘03)
Exploiting network autocorrelation with collective classification

Use label propagation/approximate inference to **collectively** classify unobserved nodes in partially labeled network
How do we learn models for collective classification?
In practice we have a single partially-labeled network

How do we learn a model and make predictions?

*Within-network learning*
Main approaches to collective classification

<table>
<thead>
<tr>
<th>Graph regularization</th>
<th>No learning</th>
<th>Add links</th>
<th>Learn weights</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRF (Zhu et al.'03); wvRN</td>
<td>wvRN+ (Macskassy '07);</td>
<td>GNetMine (Ji et al.'10);</td>
<td></td>
</tr>
<tr>
<td>(Macskassy et al.'07)</td>
<td>GhostEdges (Gallagher et al.'08);</td>
<td>LNP (Shi et al.'11);</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SCRN (Wang et al.'13)</td>
<td>LGCW (Dhurandhar et al.'13)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Probabilistic modeling</td>
<td>Learn from labeled only</td>
<td>Add graph features</td>
<td>Semi-sup. learning</td>
</tr>
<tr>
<td>RMN (Taskar et al.'01); MLN</td>
<td>SocialDim (Tang et al.'09);</td>
<td>LBC (Lu et al.'03);</td>
<td></td>
</tr>
<tr>
<td>(Richardson et al.'06); RDN</td>
<td>RelationStr (Xiang et al.'10);</td>
<td>PL-EM (Xiang et al.'08);</td>
<td></td>
</tr>
<tr>
<td>(Neville et al.'06)</td>
<td></td>
<td>CC-HLR (McDowell et al.'12);</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RDA (Pfeiffer et al.'14)</td>
<td></td>
</tr>
</tbody>
</table>
Graph regularization perspective
Graph-based Semi-supervised Learning
Labeled and Unlabeled Data as a Graph

- Idea: Construct a graph connecting similar data points
- Let the hidden/observed labels be random variables on the nodes of this graph (i.e. the graph is an MRF)
- Intuition: Similar data points have similar labels
- Information “propagates” from labeled data points
- Graph encodes intuition

Gaussian Random Fields: Zhu, Ghahramani, Lafferty ICML’03
Same idea applied to relational data to incorporate **guilt by association** into collective classification (wvRN—Macskassy and Provost MRDM’03)
Weighted-vote Relational Neighbor (wvRN)

• To find the label of a node $v_i$, perform a weighted vote among its neighbors $\mathcal{N}(v_i)$:

$$P(y_i = 1|\mathcal{N}(v_i)) = \frac{1}{|\mathcal{N}(v_i)|} \sum_{v_j \in \mathcal{N}(v_i)} P(y_j = 1|\mathcal{N}(v_j))$$

• Collective classification iteratively recomputes probabilities based on current beliefs of neighbors until convergence
Apply model to make predictions collectively

Small world graph
Labeled nodes: 30%
Autocorrelation: 50%
Apply model to make predictions **collectively**

Random graph
Labeled nodes: 30%
Autocorrelation: 50%
Extensions

- Graph regularization is used in mainstream ML for i.i.d. data — links are added to account for instance similarity

- In the network community, the links are given apriori and represent relations among the instances, so work has focused on:
  
  - Adding extra links/weights to improve performance:
    - Macskassy (AAAI’07) incorporates several types of links, weighting each by assortativity
    - Gallagher et al. (KDD’08) add “ghost edges” based on random walks
    - Wang et al. (KDD’13) weight edges based on social context features
• **Learning how to weight existing and added links:**
  
  • Ji et al. (PKDD’10) learn how to weight meta-paths in heterogeneous networks
  
  • Shi et al. (CIKM’11) learn how to weight different types of latent links
  
  • Dhurandhar et al. (JAIR’13) learn how to weight individual links
Probabilistic modeling perspective
Machine learning 101

1. Data representation
2. Knowledge representation
3. Objective function
4. Search algorithm

Relational data

Relational models

Social networks

$P_G(y|x) = \frac{1}{Z} \prod_{T \in T} \prod_{C \in C_T(G)} \Phi_T(x^C, y^C; \theta_T)$
There has been a great deal of work on templated graphical model representations for relational data.

Since model representation is also graphical we need to distinguish data networks from model networks.
Data network
Gender?  Married?  Politics?  Religion?

Data network
1 Data representation

Estimate joint distribution: \( \mathbb{P}(Y_i | \{X_i\}_i, G) \)

or conditional distribution: \( \mathbb{P}(Y_i | X_i, X_R, Y_R) \)

Note we often have only a single network for learning.
Define structure of graphical model

Relational template
Model template

Relational template

\[ \begin{align*}
  Y_i & \quad Y_j \\
  Y_i & \quad X_i^1 \\
  Y_i & \quad X_i^2 \\
  Y_i & \quad X_i^3
\end{align*} \]
Model template + Data network
Model network
(rolled-out graphical model)
Learn model parameters from fully labeled network

\[ P(y_G|x_G) = \frac{1}{Z(\theta, x_G)} \prod_{T \in T} \prod_{C \in C(T(G))} \Phi_T(x_C, y_C; \theta_T) \]
Apply model to make predictions in another network drawn from the same distribution

Model template

Model rolled out on test network
(assumption: network is drawn from same distribution as training)

Test network
Collective classification uses full joint, rolled out model for inference… but labeled nodes impact the final model structure
Collective classification uses full joint, rolled out model for inference… but labeled nodes impact the final model structure.

The structure of “rolled-out” relational **graphical models** are determined by the structure of the underlying **data network**, including location + availability of labels.

…this can **impact performance** of learning and inference methods via representation, objective function, and search algorithm.
Networks are much, much larger in practice... and often there’s only one partially labeled network.
Approach #1: Ignore unlabeled during learning

- Drop out unlabeled nodes; **Training data** = labeled part of network
- Model is defined via local conditional; optimize params using pseudolikelihood

\[
P(Y|X, E, \Theta_Y)\]

\[
\hat{\Theta}_Y = \arg\max_{\Theta_Y} \sum_{v_i \in V_L} \log P_Y(y_i | Y_{MBL}(v_i), x_i, \Theta_Y)
\]

\[
\hat{\Theta}_Y = \arg\max_{\Theta_Y} \{\text{summation over local log conditionals}\}
\]
Approach #1: Apply learned model to remainder

- **Test data** = full network (but only evaluate on unlabeled)
  - Labeled nodes seed the inference
- Use approximate inference to collectively classify (e.g., Variational, Gibbs)

\[
P(Y | X, E, \Theta_Y)
\]

For unlabeled instances, iteratively estimate:

\[
P_Y(y_i | Y_{MB}(v_i), x_i, \Theta_Y)
\]
Approach #2: Add graph features during learning

- **Training data** = labeled nodes in network; Added features to incorporate unlabeled graph structure
  - Tang and Liu (KDD’09) find low-rank approximation of graph structure, then use as features
  - Xiang et al. (WWW’10) use unsupervised learning to model relationship strength and weight edges
Approach #3: Semi-supervised learning

- Use entire network to jointly learn parameters and make inferences about class labels of unlabeled nodes

- Lu and Getoor (ICML’03) use relational features and ICA

- McDowell and Aha (ICML’12) combine two classifiers with label regularization
Semi-supervised relational learning

- Relational Expectation Maximization (EM) (Xiang & Neville ’08)

Predict labels with collective classification

Use predicted probabilities during optimization (in local conditionals)
How does relational EM perform?

- Works well when network has a moderate amount of labels
- If network is sparsely labeled, it is often better to use a model that is not learned
- Why? In sparsely labeled networks, errors from the collective classification compound during propagation.

Both learning and inference require approximation and network structure impacts errors.
Impact of approximations in semi-supervised RL

- Does over propagation during prediction happen in real world, sparsely labeled networks? YES.

![Graphs showing impact of approximations in semi-supervised RL](image)

- Conditional: *Relational Naive Bayes*

- Conditional: *Relational Logistic Regression*
Finding #1: Network structure can bias inference in partially-labeled networks; maximum entropy constraints correct for bias
Effect of relational biases on R-EM

- We compared CL-EM and PL-EM and examined the distribution of predicted probabilities on a real world dataset
  - Amazon Co-occurrence (SNAP)
  - Varied class priors, 10% Labeled

- **Overpropagation error** during inference causes PL-EM to collapse to single prediction
- **Worse** on sparsely labeled datasets

```
Need method to correct bias for any method based on local (relational) conditional
```
Maximum entropy inference for PL-EM (Pfeiffer et al. WWW’15)

- Correction to inference (E-Step)
  - Enables estimation with the pseudolikelihood (M-Step)
- Idea: The proportion of negatively predicted items should equal the proportion of negatively labeled items
  - Fix: Shift the probabilities up/down
- Repeat for each inference itr

Transform probabilities to logit space:
\[ h_i = \sigma^{-1}(P(y_i = 1)) \]

Compute offset location:
\[ \phi = P(0) \cdot |V_U| \]

Adjust logits:
\[ h_i = h_i - h(\phi) \]

Transform back to probabilities:
\[ P(y_i) = \sigma(h_i) \]

Corrected probabilities are used to retrain during PL-EM (M-Step)
Experimental results - Correction effects

Max entropy correction removes bias due to over propagation in collective inference
Experimental results - Large patent dataset

Correction allows relational EM to improve over competing methods in sparsely labeled domains

Note: McDowell & Aha (ICML’12) may correct same effect, but during estimation rather than inference
Finding #2:
Network structure can bias learning in partially-labeled networks; *modeling uncertainty corrects for bias*
Impact of approximations in semi-supervised RL

- **Over correction** also happens during **parameter estimation** in semi-in real world, sparsely labeled networks.

![Diagram](image-url)

Relational Naive Bayes  
Relational Logistic Regression
Impact of approximations in semi-supervised RL

• Relational EM:

E-Step: \( P_{\tilde{\mathbf{y}}} (y_i | \mathbf{Y}_{\mathcal{MB}} (v_i), \mathbf{x}_i, \Theta_{\mathcal{Y}}) \) (For all unlabeled instances)

M-Step: \( \hat{\Theta}_{\mathcal{Y}} = \arg \max_{\Theta_{\mathcal{Y}}} \sum_{\mathbf{Y}_U \in \mathcal{Y}_U} P_{\mathcal{Y}} (\mathbf{Y}_U) \sum_{v_i \in \mathcal{V}_L} \log P_{\hat{\mathcal{Y}}} (y_i | \tilde{\mathbf{Y}}_{\mathcal{MB}} (v_i), \mathbf{x}_i, \Theta_{\mathcal{Y}}) \)

Collective Inference Approximation

Pseudolikelihood Approximation

• Over-propagation error
  - Both classes neighbor yellow… thus yellow neighbors are

Accounting for uncertainty in parameter estimates will correct for this bias
Relational data augmentation (*Pfeiffer et al. ICDM’14*)

- Data augmentation is a Bayesian version of EM
  - Parameters are RVs
  - Compute *posterior* predictive distribution (Tanner & Wong ’87)

- We developed a relational version of data augmentation
  - Final inference is over a *distribution* of parameter values
  - Requires prior distributions over parameters and sampling methods

### Table

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Predictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed Point</td>
<td>Stochastic</td>
</tr>
<tr>
<td>Fixed Point</td>
<td>Relational EM</td>
</tr>
<tr>
<td>Stochastic</td>
<td>Relational Stochastic EM</td>
</tr>
</tbody>
</table>

### Equations

Alternate Between:

- Gibbs sample of labels
  \[
  \tilde{Y}_U^t \sim P_t^{\tilde{\Theta}_{\tilde{Y}}} (Y_U | Y_L, X, E, \tilde{\Theta}_{\tilde{Y}}^{t-1})
  \]

- Sample Parameters
  \[
  \tilde{\Theta}_{\tilde{Y}}^t \sim P_t (\Theta_{\tilde{Y}} | \tilde{Y}_U^t, Y_L, X, E)
  \]

Final Parameters:

- \[
  \hat{\Theta} = \frac{1}{T} \sum_t \tilde{\Theta}_{\tilde{Y}}^t
  \]

Final Inference:

- \[
  \hat{Y}_U^t = \frac{1}{T} \sum_t \tilde{Y}_U^t
  \]
Experimental results - Amazon DVD

Relational EM’s instability in sparsely labeled domains causes poor performance
Experimental results - Facebook

Relational data augmentation can outperform relational stochastic EM

Relational EM

Relational SEM

Relational DA

Relational EM

Relational DA

Naive Bayes

Logistic Regression

Percentage of Graph Labeled

MAE
Finding #3: Implicit assumption is that nodes of the same type should be **identically distributed**—but many relational representations cannot ensure this holds for varying graph structures.
I.I.D. assumption revisited

• Current relational models do not impose the same \textbf{marginal invariance} condition that is assumed in IID models, which can impair generalization

\[ p(y_A|\mathbf{x}_A) \neq p(y_E|\mathbf{x}_E) \] due to varying graph structure

• Markov relational network representation does not allow us to explicitly specify the form of the marginal probability distributions, thus it is difficult to impose any equality constraints on the marginals
Is there an alternative approach?

- **Goal**: Combine the marginal invariance advantages of IID models with the ability to model relational dependence
  - Incorporate node attributes in a general way (similar to IID classifiers)
- **Idea**: Apply copulas to combine marginal models with dependence structure

**Copula theory**: can construct n-dimensional vector of arbitrary marginals while preserving the desired dependence structure.
Let’s start with a reformulation of IID classifiers...

• General form of probabilistic binary classification: \( p(y_i = 1) = F(\eta(x_i)) \)
  - e.g., Logistic regression

• Now view \( F \) as the CDF of a distribution symmetric around 0 to obtain a
  **latent variable formulation**:
  \[
  z_i \sim p(z_i = z | x_i = x) = f(z - \eta(x_i)) \\
  y_i = \text{sign}(z_i)
  \]

• \( z \) is a continuous variable, capturing random effects that are not present in \( x \)
• \( p \) is the corresponding PDF of \( F \)

• In IID models, the random effect for each instance is independent, thus can be integrated out

• When links among instances are observed, the **correlations** among their class labels can be
  modeled through dependence among the \( z \)’s

• **Key question**: How to model the dependence among \( z \)’s while **preserving the marginals**?
Copula Latent Markov Network (CLMN)

The CLMN model

- Sample $t$ from the desired joint dependency: $(t_1, t_2, \ldots, t_n) \sim \phi$

- Apply marginal transformation to obtain the latent variable $z$: $z_i = F_i^{(-1)}(\Phi_i(t_i))$

- Classification: $y_i = \text{sign}(z_i)$
**CLMN implementation**

*Dependence structure*—Joint PDF $\phi$ over latent variables $t$:

$$
\phi(t) = \frac{1}{Z} \exp\left(-\frac{1}{2} \left( \sum_{i,j:(i,j) \in E} (t_i - t_j)^2 \right) - \frac{\epsilon}{2} \sum_{i=1}^{n} t_i^2 \right)
$$

*Gaussian Markov network*

*Marginal structure*—each PDF $f$ is an independent classifier of latent variable $z$:

$$
f_i(z_i|w, x_i) = \frac{e^{-(z_i - w^T x_i)}}{(1 + e^{-(z_i - w^T x_i)})^2}
$$

*Logistic regression*

*Glue between joint dependence and marginal models*—marginal CDF transform:

$$
\Phi_i(t_i) = u_i = F_i(z_i)
$$

*Classification*—discrete label is determined from latent variable sign:

$$
y_i = \text{sign}(z_i)
$$

**Estimation**:  
- First, learn marginal model as if instances were IID  
- Next, learn the dependence model conditioned on the marginal model... *but GMN has no parameters to learn*

**Inference**:  
- Conditional inference in copulas have not previously been considered for large-scale networks  
- For *efficient* inference, we developed a message passing algorithm based on EP
**Key idea:** Ensuring that nodes with varying graph structure have identical marginals improves learning.
Conclusion

• Relational models have been shown to significantly improve network predictions through the use of joint modeling and collective inference.

• *But we need to understand the impact of real-world graph characteristics on model/algorithms in order to better exploit network information for learning and prediction.*

• A careful consideration of interactions between:

  **data representation, knowledge representation, objective functions, and search algorithms**

  will improve our understanding of the choices/mechanisms that affect performance and drive the development of new algorithms.
What can network science offer to RML?

• Consider effects of graph structure on learning/inference
  • ML methods learn a model conditioned on graph \( P(Y|\{X\}_n, G) \)

• When is a new graph “drawn from the same distribution”? How to model distributions of networks? (see Moreno et al. KDD’13)

• If two networks are drawn from different distributions, how can we transfer a model learned on one network to apply it in another? (see Niu et al. ICDMW’15)
What can network science offer to ML?

• How does network structure impact performance of ML methods?
  • Need to study the properties of the rolled out model network, comparing learning setting with inference setting
  • ML researchers study the effects of generic graph properties on graphical model inference. Should consider network characteristics like connected components, density, clustering, …

• We are starting to study this with synthetically generated attributed networks (see Pfeiffer et al. WWW’14)
Questions?

neville@cs.purdue.edu
www.cs.purdue.edu/~neville