Label propagation on graphs

Leonid E. Zhukov

School of Data Analysis and Artificial Intelligence
Department of Computer Science
National Research University Higher School of Economics

Network Science
Lecture outline

1. Label propagation problem
2. Collective classification
   - Iterative classification
3. Semi-supervised learning
   - Random walk based methods
   - Graph regularization
Label propagation

- Label propagation - labeling of all nodes in a graph structure
- Subset of nodes is labeled: categorical/numeric/binary values
- Extend labeling to all nodes on the graph (class/class probability/regression)
- Classification in networked data, network classification, structured inference, relational learning
Label propagation problem

- Structure can help only if labels/values of linked nodes are correlated
- Social networks show assortative mixing - bias in favor of connections between network nodes with similar characteristics:
  - homophily: similar characteristics $\rightarrow$ connections
  - influence: connections $\rightarrow$ similar characteristics
- Can apply to constructed (induced) similarity networks
Network classification

Supervised learning approach

- Given graph nodes $V = V_l \cup V_u$:
  - nodes $V_l$ given labels $Y_l$
  - nodes $V_u$ do not have labels
- Need to find $Y_u$
- Labels can be binary, multi-class, real values
- Features (attributes) can be computed for every node $\phi_i$:
  - local node features (if available)
  - link features available (labels from neighbors, attributes from neighbors, node degrees, connectivity patterns)
- Feature (design) matrix $\Phi = (\Phi_l, \Phi_u)$
Network learning components

- **Local classifier.** This is a local learned model, predicts node label based on node attributes. No network information.

- **Relational classifier.** Takes into account labels and attributes of node neighbors. Uses neighborhood network information.

- **Collective classifier.** Estimates unknown values together applying relational classifier iteratively. Strongly depends on network structure.
Relational classifiers

- Weighted-vote relational neighbor classifier:

\[ P(y_i = c|\mathcal{N}_i) = \frac{1}{Z} \sum_{i \in \mathcal{N}_i} A_{ij} P(y_j = c|\mathcal{N}_j) \]

- Network only Naive Bayes classifier:

\[ P(y_i = c|\mathcal{N}_i) = \frac{P(\mathcal{N}_i|c)P(c)}{P(\mathcal{N}_i)} \]

where

\[ P(\mathcal{N}_i|c) = \frac{1}{Z} \prod_{j \in \mathcal{N}_i} P(y_j = \hat{y}_j|y_i = c) \]
Iterative classification

(a) Step 1

(b) Step 2
Algorithm: Iterative classification method

Input: Graph $G(V, E)$, labels $Y_l$

Output: labels $\hat{Y}$

Compute $\Phi^{(0)}$

Train classifier on $(\Phi^{(0)}_l, Y_l)$

Predict $Y^{(0)}_u$

repeat
    Compute $\Phi^{(t)}_u$
    Train classifier on $(\Phi^{(t)}_l, Y^{(t)}_l)$
    Predict $Y^{(t+1)}_u$ from $\Phi^{(t)}_u$
until $Y^{(t)}_u$ converges;

$\hat{Y} \leftarrow Y^{(t)}$
Graph-based semi-supervised learning

Given partially labeled dataset

Data: $X = X_l \cup X_u$
- small set of labeled data $(X_l, Y_l)$
- large set of unlabeled data $X_u$

Similarity graph over data points $G(V, E)$, where every vertex $v_i$ corresponds to a data point $x_i$

Transductive learning: learn a function that predicts labels $Y_u$ for the unlabeled input $X_u$
Random walk methods

- Consider random walk with absorbing states - labeled nodes $V_l$
- Probability $\hat{y}_i[c]$ for node $v_i \in V_u$ to have label $c$,

$$\hat{y}_i[c] = \sum_{j \in V_l} p_{ij}^\infty y_j[c]$$

where $y_i[c]$ - probability distribution over labels,
$p_{ij} = P(i \rightarrow j)$ - one step probability transition matrix
- If output requires single label per node, assign the most probable
- In matrix form

$$\hat{Y} = P^\infty Y$$

where $Y = (Y_l, 0)$, $\hat{Y} = (Y_l, \hat{Y}_u)$
Random walk methods

- Random walk matrix: \( P = D^{-1}A \)
- Random walk with absorbing states

\[
P = \begin{pmatrix} P_{ll} & P_{lu} \\ P_{ul} & P_{uu} \end{pmatrix} = \begin{pmatrix} I & 0 \\ P_{ul} & P_{uu} \end{pmatrix}
\]

- At the \( t \to \infty \) limit:

\[
\lim_{t \to \infty} P^t = \begin{pmatrix} I \\ \left( \sum_{n=0}^{\infty} P_{uu}^n \right) P_{ul} \end{pmatrix} = \begin{pmatrix} I \\ (I - P_{uu})^{-1} P_{ul} \end{pmatrix}
\]
Matrix equation

\[
\begin{pmatrix}
\hat{Y}_l \\
\hat{Y}_u
\end{pmatrix} =
\begin{pmatrix}
I \\
(I - P_{uu})^{-1}P_{ul}
\end{pmatrix}
\begin{pmatrix}
0 \\
0
\end{pmatrix}
\begin{pmatrix}
Y_l \\
Y_u
\end{pmatrix}
\]

Solution

\[
\begin{align*}
\hat{Y}_l &= Y_l \\
\hat{Y}_u &= (I - P_{uu})^{-1}P_{ul}Y_l
\end{align*}
\]

\((I - P_{uu})\) is non-singular for all label connected graphs (is always possible to reach a labeled node from any unlabeled node)
Label propagation

Algorithm: Label propagation, Zhu et. al 2002

Input: Graph $G(V, E)$, labels $Y_l$

Output: labels $\hat{Y}$

Compute $D_{ii} = \sum_j A_{ij}$

Compute $P = D^{-1}A$

Initialize $Y^{(0)} = (Y_l, 0)$, t=0

repeat
  $Y^{(t+1)} \leftarrow P \cdot Y^{(t)}$
  $Y_l^{(t+1)} \leftarrow Y_l^{(t)}$
until $Y^{(t)}$ converges;

$\hat{Y} \leftarrow Y^{(t)}$

Solution: $\hat{Y} = \lim_{t \to \infty} Y^{(t)} = (I - P_{uu})^{-1}P_{ul} Y_l$
Algorithm: Label spreading, Zhou et. al 2004

Input: Graph $G(V, E)$, labels $Y_I$

Output: labels $\hat{Y}$

Compute $D_{ii} = \sum_j A_{ij}$,

Compute $S = D^{-1/2}AD^{-1/2}$

Initialize $Y^{(0)} = (Y_I, 0), t=0$

repeat

\[ Y^{(t+1)} \leftarrow \alpha SY^{(t)} + (1 - \alpha)Y^{(0)} \]

\[ t \leftarrow t + 1 \]

until $Y^{(t)}$ converges;

Solution: $\hat{Y} = (1 - \alpha)(I - \alpha S)^{-1}Y^{(0)}$
Regression on graphs

Find labeling $\hat{Y} = (\hat{Y}_l, \hat{Y}_u)$ that

- Consistent with initial labeling:
  \[
  \sum_{i \in V_l} (\hat{y}_i - y_i)^2 = ||\hat{Y}_l - Y_l||^2
  \]

- Consistent with graph structure (regression function smoothness):
  \[
  \frac{1}{2} \sum_{i,j \in V} A_{ij}(\hat{y}_i - \hat{y}_j)^2 = \hat{Y}^T (D - A) \hat{Y} = \hat{Y}^T L \hat{Y}
  \]

- Stable (additional regularization):
  \[
  \epsilon \sum_{i \in V} \hat{y}_i^2 = \epsilon ||\hat{Y}||^2
  \]
Minimization with respect to $\hat{Y}$, arg min$_{\hat{Y}} Q(\hat{Y})$

- **Label propagation** [Zhu, 2002]:

  \[
  Q(\hat{Y}) = \frac{1}{2} \sum_{i,j \in V} A_{ij}(\hat{y}_i - \hat{y}_j)^2 = \hat{Y}^T L \hat{Y}, \quad \text{with fixed } \hat{Y}_l = Y_l
  \]

- **Label spread** [Zhou, 2003]:

  \[
  Q(\hat{Y}) = \frac{1}{2} \sum_{i,j \in V} A_{ij} \left( \frac{\hat{y}_i}{\sqrt{d_i}} - \frac{\hat{y}_j}{\sqrt{d_j}} \right)^2 + \mu \sum_{i \in V} (\hat{y}_i - y_i)^2
  \]

  \[
  Q(\hat{Y}) = \hat{Y}^T L \hat{Y} + \mu \| \hat{Y} - Y \|^2
  \]

  \[
  L = I - S = I - D^{-1/2} A D^{-1/2}
  \]
Regularization on graphs

- Laplacian regularization [Belkin, 2003]

\[
Q(\hat{Y}) = \frac{1}{2} \sum_{ij \in V} A_{ij} (\hat{y}_i - \hat{y}_j)^2 + \mu \sum_{i \in V_i} (\hat{y}_i - y_i)^2
\]

\[
Q(\hat{Y}) = \hat{Y}^T L \hat{Y} + \mu \| \hat{Y}_l - Y_l \|^2
\]

- Use eigenvectors \((e_1..e_p)\) from smallest eigenvalues of \(L = D - A\):

\[Le_j = \lambda_j e_j\]

- Construct classifier (regression function) on eigenvectors

\[
Err(a) = \sum_{i \in V_l} (y_i - \sum_{j=1}^{p} a_j e_{ji})^2
\]

- Predict value (classify) \(\hat{y}_i = \sum_{j=1}^{p} a_j e_{ji}\), class \(c_i = \text{sign}(\hat{y}_i)\)
Algorithm:  Laplacian regularization, Belkin and Niyogy, 2003

Input: Graph $G(V, E)$, labels $Y_l$

Output: labels $\hat{Y}$

Compute $D_{ii} = \sum_j A_{ij}$

Compute $L = D - A$

Compute $p$ eigenvectors $e_1...e_p$ with smallest eigenvalues of $L$, $Le = \lambda e$

Minimize over $a_1...a_p$

$$\arg \min_{a_1,...,a_p} \sum_{i=1}^{l} (y_i - \sum_{j=1}^{p} a_j e_{ji})^2, \quad a = (E^T E)^{-1} E^T Y_l$$

Label $v_i$ by the sign($\sum_{j=1}^{p} a_j e_{ji}$)
Label propagation example
Label propagation example


