Outline

- PageRank
- Semi-supervised Learning
- Label Propagation
- Manifold regularization
PageRank
Text based ranking systems (a dominated approach in the early 90s)
- Compute the similarity between query and websites (documents)
- Keywords are a very limited way to express a complex information
- Need to rank websites by “popularity”, “authority”, ...

PageRank:
- Developed by Brin and Page (1999)
- Determine the authority and popularity by hyperlinks
PageRank

Main idea: estimate the ranking of websites by the link structure.

Topology of Websites

- Transform the hyperlinks to a directed graph:
- The adjacency matrix $A$ such that $A_{ij} = 1$ if page $j$ points to page $i$
Normalize the adjacency matrix so that the matrix is a stochastic matrix (each column sum up to 1)

$P_{ij}$: probability that arriving at page $i$ from page $j$

$P$: a stochastic matrix or a transition matrix
Random walk: step 1

- Random walk through the transition matrix
- Start from \([1, 0, 0, 0]\) (can use any initialization)
Random walk: step 1

- Random walk through the transition matrix
- $x^{t+1} = Px^t$
Random walk: step 2

- Random walk through the transition matrix

![Diagram of a random walk through a transition matrix]
Random walk: step 2

- Random walk through the transition matrix
- $x^{t+2} = Px^{t+1}$
PageRank (convergence)

- Start from an initial vector $\mathbf{x}$ with $\sum_i x_i = 1$ (the initial distribution)
- For $t = 1, 2, \ldots$
  \[ x^{t+1} = P x^t \]
- Each $x^t$ is a probability distribution (sums up to 1)
- Converges to a stationary distribution $\pi$ such that
  \[ \pi = P \pi \]

if $P$ satisfies the following two conditions:

1. $P$ is irreducible: for all $i, j$, there exists some $t$ such that $(P^t)_{i,j} > 0$
2. $P$ is aperiodic: for all $i, j$, we have $\text{gcd}\{t : (P^t)_{i,j} > 0\} = 1$

- $\pi$ is the unique right eigenvector of $P$ with eigenvalue 1.
PageRank

- How to guarantee convergence?
  - Add the possibility of jumping to a random node with small probability $\alpha$, we get the commonly used PageRank

$$\pi = (\alpha P + (1 - \alpha)ve^T)\pi$$

- $v = \frac{1}{n}e = [\frac{1}{n} \frac{1}{n} \ldots \frac{1}{n}]^T$ is commonly used

- Personalized PageRank: $v = e_i$
PageRank: Iterative Algorithm

**Input:** Transition matrix $P$ and personalization vector $v$

1. Initial $x_i^{(0)} = \frac{1}{n}$ for $i = 1, 2, \ldots, n$
2. for $t = 1, 2, \ldots$ do
   - $x^{(t+1)} \leftarrow \alpha Px^{(t)} + (1 - \alpha)v$
3. end for

Time complexity: $O(\text{nnz}(P))$ per iteration
Label Propagation
Semi-supervised Learning

- Given both labeled and unlabeled data
- Is unlabeled data useful?

Figure from Wikipedia
Semi-supervised Learning

- Two approaches:
  - Graph-based algorithm (label propagation)
  - Graph-based regularization (manifold regularization)
## Inductive vs Transductive

<table>
<thead>
<tr>
<th>Supervised (labeled data)</th>
<th>Inductive (Generalize to unseen data)</th>
<th>Transductive (Doesn’t generalize to unseen data)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM, ...</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Semi-supervised (labeled + unlabeled)</td>
<td>Manifold Regularization</td>
<td>Label propagation</td>
</tr>
</tbody>
</table>

- **Inductive** methods allow generalization to unseen data.
- **Transductive** methods do not generalize to unseen data.
- **Supervised** learning uses labeled data.
- **Semi-supervised** learning uses labeled and unlabeled data.
Main Idea

- **Smoothness Assumption**
  
  If two data points are similar, then the output labels should also be similar.
Smoothness Assumption

If two data points are similar, then the output labels should also be similar.

- Measure the similarity between data points (similarity graph)
- How to enforce the output labels to be similar?
Assume we have $n$ data points $x_1, \ldots, x_n$

Define the similarity matrix $S \in \mathbb{R}^{n \times n}$

$$S_{ij} = \text{similarity}(x_i, x_j)$$

The similarity function can be defined by many ways, for example,

$$\text{similarity}(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2)$$

$S$ is a dense $n \times n$ matrix $\Rightarrow$ High computational cost

Usually, a sparse similarity graph is preferred

Usually, a $k$-nearest neighbors graph is used:

$S_{ij} \neq 0$ only when $j$ is in $i$’s $k$-nearest neighbors
Transductive setting using label propagation

- **Input:**
  - \( \ell \) labeled training samples \( x_1, \ldots, x_\ell \) with labels \( y_1, \ldots, y_\ell \) (each \( y_i \) is a \( k \) dimensional label vector for multiclass/multilabel problems)
  - \( u \) unlabeled training samples \( x_{\ell+1}, \ldots, x_{\ell+u} \)

- **Output:** labels \( y_{\ell+1}, \ldots, y_{\ell+u} \) for all unlabeled samples

- **Main idea:** propagate labels through the similarity matrix
Label Propagation

- \( T \in \mathbb{R}^{(\ell+u) \times (\ell+u)} \): transition matrix such that
  \[
  T_{ij} = P(j \rightarrow i) = \frac{S_{ij}}{\sum_{k=1}^{\ell+u} S_{kj}}
  \]
- \( Y \in \mathbb{R}^{(\ell+u) \times k} \): the label matrix. The first \( \ell \) rows are labels \( y_1, \ldots, y_\ell \). The rest \( u \) rows are initialized by 0 (will not affect the results).
- The Algorithm:
  Repeat the following steps until convergence:
  Step 1: Propagate labels by the transition matrix: \( Y \leftarrow TY \)
  Step 2: Normalize each row of \( Y \)
  Step 3: Reset the first \( \ell \) rows of \( Y \) to be \( y_1, \ldots, y_\ell \)
The algorithm will converge to a simple solution!

Step 1 and step 2 can be combined into

\[ Y \leftarrow \bar{T}Y, \]

where \( \bar{T} \) is the row-normalized matrix of \( T \)

We focus on row \( \ell + 1 \) to \( \ell + u \) (defined by \( Y_U \))

\[
\begin{pmatrix}
Y_L \\
Y_U
\end{pmatrix}
\leftarrow
\begin{pmatrix}
\bar{T}_{\ell\ell} & \bar{T}_{\ell u} \\
\bar{T}_{ul} & \bar{T}_{uu}
\end{pmatrix}
\begin{pmatrix}
Y_L \\
Y_U
\end{pmatrix}
\]
So \( Y_U \leftarrow \bar{T}_{ul} Y_L + \bar{T}_{uu} Y_U \)

Run for infinite number of iteration, we have

\[
Y^*_U = \lim_{t \to \infty} \bar{T}_{uu} Y^0_U + \left( \sum_{i=1}^{t} \bar{T}_{uu}^{i-1} \right) \bar{T}_{ul} Y_L
\]

vanish when \( t \to \infty \)

\[
= \lim_{t \to \infty} \left( \sum_{i=1}^{t} \bar{T}_{uu}^{i-1} \right) \bar{T}_{ul} Y_L
\]

\[
= (I - \bar{T}_{uu})^{-1} \bar{T}_{ul} Y_L
\]

\( O(\text{nnz}(S)kT) \) for both power iteration and linear system solvers (\( T: \) number of iterations)

In the following, we show another interpretation.
Graph Laplacian:

- **Graph Laplacian:**
  \[
  L = D - S, \quad \text{where } D \text{ is a diagonal matrix with } D_{ii} = \sum_j S_{ij}
  \]

- **L is positive semi-definite (if S is nonnegative)**

- **Main property:** for any vector \( z \),
  \[
  z^T L z = \sum_{i,j} S_{ij} (z_i - z_j)^2
  \]

- **Measure the non-smoothness of \( z \) according the the similarity matrix \( S \)**

- **Which \( z \) minimizes the non-smoothness?**
  Eigenvector with the smallest eigenvalue.
Another form of label propagation

Consider we only have one label

Another equivalent form of label propagation:

$$\arg\min_{\hat{y}} \sum_{i,j} S_{ij}(\hat{y}_i - \hat{y}_j)^2 = \hat{y}^T L \hat{y} := f(\hat{y})$$

s.t. $\hat{y}_{1:\ell} = y$

where $\hat{y} \in \mathbb{R}^{\ell+u}$ is the estimation of the labels.

Optimal solution: $\nabla_{\hat{y}_{\ell+1:u}} f(\hat{y}) = 0$

$$\Rightarrow \begin{pmatrix} D_{\ell\ell} - S_{\ell\ell} & -S_{\ell u} \\ -S_{u\ell} & D_{uu} - S_{uu} \end{pmatrix} \begin{pmatrix} \hat{y}_\ell \\ \hat{y}_u \end{pmatrix} = \begin{pmatrix} ? \\ 0 \end{pmatrix}$$

$$\Rightarrow (D_{uu} - S_{uu})\hat{y}_u - S_{ul}\hat{y}_\ell = 0$$

$$\Rightarrow \hat{y}_u = (D_{uu} - S_{uu})^{-1} S_{ul}\hat{y}_\ell$$

$$\Rightarrow \hat{y}_u = (I - D_{uu}^{-1} S_{uu})^{-1} D_{uu}^{-1} S_{ul}\hat{y}_\ell$$

$$\Rightarrow \hat{y}_u = (I - \bar{T}_{uu})^{-1} \bar{T}_{ul}\hat{y}_\ell$$
Experimental Results (Zhu et al., 2003)

**Figure 3.** Harmonic energy minimization on digits “1” vs. “2” (left) and on all 10 digits (middle) and combining voted-perceptron with harmonic energy minimization on odd vs. even digits (right)

**Figure 4.** Harmonic energy minimization on PC vs. MAC (left), baseball vs. hockey (middle), and MS-Windows vs. MAC (right)
Manifold Regularization
Empirical Risk Minimization with Manifold Regularization

- Setting: given labeled training data $x_1, \ldots, x_\ell$ with labels $y_1, \ldots, y_\ell$ and unlabeled training data $x_{\ell+1}, \ldots, x_{\ell+u}$, obtain a function $f$ to predict the label of new instances.
- We assume $f$ is a linear function ($f(x) = w^T x$)
- Empirical risk minimization:

$$\min_w \sum_{i=1}^{\ell} \ell(y_i, w^T x_i) + \lambda R(w),$$

where $\ell(\cdot)$ is the loss function and $R(\cdot)$ is the regularization.
- Only use labeled data.
Empirical Risk Minimization with Manifold Regularization


- We assume \( f \) is a linear function (\( f(x) = w^T x \))

- Empirical risk minimization with Manifold Regularization:

\[
\min_w \sum_{i=1}^{l} \ell(y_i, w^T x_i) + \lambda R(w) + \beta \sum_{i,j} S_{ij}(w^T x_i - w^T x_j)^2
\]

\[
= \sum_{i=1}^{l} \ell(y_i, w^T x_i) + \lambda R(w) + \beta \hat{y}^T L \hat{y},
\]

where \( \hat{y} = [w^T x_1, \ w^T x_2, \ldots, \ w^T x_{l+u}]^T \)

- Use manifold regularization to ensure that similar data points have the similar output labels
Manifold Regularization

- Trains an “inductive” classifier: can generalize to unseen instances
- Can be extended to other function classes (e.g., kernel methods)
- Can be used in other ML problems
  - Ranking, Matrix Completion, …
Experimental Comparisons

Performance of RLS, LapRLS

Performance of SVM, LapSVM
Next class: midterm exam

Questions?