FINAL PROJECT FOR THE COURSE ADVANCED FOUNDATIONS OF MACHINE LEARNING: ONLINE LEARNING OVER GRAPHS[*]

PAUL SOPHER LINTILHAC[†] AND THOMAS NANFENG LI[‡]

MAY 9$^{\text{TH}}$, 2017

1 Introduction to Graph

1.1 Concepts and Definitions

We first summarise some key concepts of graph theory, for more detailed knowledge, we refer to Bapat (2014). A ***simple graph***, that is, graph without loops and parallel edges, $G\,(V,\,E)$ consists of a finite set of ***vertices*** $V(G)$ and a set of ***edges*** $E(G)$ consisting of distinct, unordered pairs of vertices. $V(G) = \{v_1,\,v_2,\,\cdots,\,v_n\}$ is called the vertex set with $n = |V(G|$, $E(G) = \{e_{ij}\}$ is called the edge set with $m = |E(G)|$. An edge e_{ij} connects vertices v_i and v_j if they are ***adjacent*** or neighbours, which is denoted by $v_i \sim v_j$. The number of neighbours of a vertex v is called the ***degree*** of v and is denoted by $d\,(v)$, therefore, for each vertex, $d\,(v_i) = \sum_{v_i \sim v_j} e_{ij}$. If all the vertices of a graph have the same degree, the graph is ***regula***r, the vertices of an ***Eulerian Graph*** have even degree. A graph is ***complete*** if there is an edge between every pair of vertices.

$H\,(G)$ is a $\boldsymbol{sub-graph}$ of G if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. A sub-graph $H\,(G)$ is an $\boldsymbol{induced\ sub-graph}$ of G if two vertices of $V(H)$ are adjacent if and only if they are adjacent in G. A ***clique*** is a complete sub-graph of a graph. A ***path*** of k vertices is a sequence of k distinct vertices such that consecutive vertices are adjacent. A ***cycle*** is a connected sub-graph where every vertex has exactly two neighbours. A graph containing no cycles is a ***forest***. A connected forest is a ***tree***.

We define incidence matrix of graph. Let $G\,(V,\,E)$ be a graph with $V(G) = \{v_1,\,v_2,\,\cdots,\,v_n\}$ and $E(G) = \{e_1,\,e_2,\,\cdots,\,e_m\}$. Suppose each edge of $G\,(V,\,E)$ is assigned an orientation, which is arbitrary but fixed. The vertex-edge ***incidence matrix*** of $G\,(V,\,E)$, denoted by $\boldsymbol{Q}(G)$, is the $n \times m$ matrix defined as follows. The rows and the columns of $\boldsymbol{Q}(G)$ are indexed by $V(G)$ and $E(G)$, respectively. The $(i,\,j)$ entry of $\boldsymbol{Q}(G)$ is 0 if vertex i and edge e_j are not incident, and otherwise it is -1 or 1 according as e_j originates or terminates at i, respectively. For instance, the incidence matrix $\boldsymbol{Q}(G)$ of the graph that is shown in figure 1.1 is

$$\boldsymbol{Q}(G) = \begin{bmatrix} 1 & -1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & 1 & -1 \end{bmatrix}. \tag{1.1}$$

We introduce adjacency matrix of graph. Let $G\,(V,\,E)$ be a graph with $V(G) = \{v_1,\,v_2,\,\cdots,\,v_n\}$ and $E(G) = \{e_1,\,e_2,\,\cdots,\,e_m\}$. The ***adjacency matrix*** of $G\,(V,\,E)$, denoted by $\boldsymbol{A}(G)$, is the $n \times n$ matrix defined as follows. The rows and the columns of $\boldsymbol{A}(G)$ are indexed by $V(G)$. If $i \neq j$ then the $(i,\,j)$ entry of $\boldsymbol{A}(G)$ is 0 for vertices i and j non-adjacent, and the $(i,\,j)$ entry is 1 for i and j adjacent. The $(i,\,j)$ entry of $\boldsymbol{A}(G)$ is 0 for $i = j = 1,\,\cdots,n$. For instance, the adjacency matrix $\boldsymbol{A}(G)$ of the graph that is shown in

[*] New York University Courant Institute of Mathematical Sciences. Spring 2017. Professor Mehryar Mohri, Ph.D.
[†] New York University School of Engineering. psl274@nyu.edu
[‡] New York University School of Engineering. nl747@nyu.edu

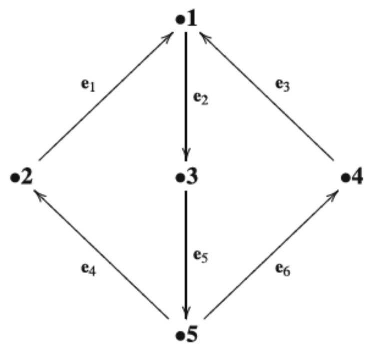

Figure 1.1: Example of Incidence Matrix of Graph

figure 1.2 is

$$\boldsymbol{A}(G)=\begin{bmatrix}0&1&1&1&0\\1&0&1&0&0\\1&1&0&1&1\\1&0&1&0&1\\0&0&1&1&0\end{bmatrix}. \tag{1.2}$$

Clearly $\boldsymbol{A}$ is a symmetric matrix with zeros on the diagonal. The (i, j) entry of $\boldsymbol{A}^k$ is the number of walks of length k from i to j.

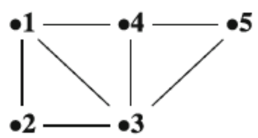

Figure 1.2: Example of Incidence Matrix of Graph

We define degree matrix of graph. Let $G(V, E)$ be a graph with $V(G)=\{v_1, v_2, \cdots, v_n\}$ and $E(G)=\{e_1, e_2, \cdots, e_m\}$. The ***degree matrix*** $\boldsymbol{D}(G)$ for $G(V, E)$ is a $n\times n$ diagonal matrix defined as

$$\boldsymbol{D}(G)_{i,j}:=\begin{cases}d(v_i) & \text{if } i=j\\ 0 & \text{otherwise.}\end{cases}$$

According to this definition, the degree matrix of figure 1.2 is

$$\boldsymbol{A}(G)=\begin{bmatrix}3&0&0&0&0\\0&2&0&0&0\\0&0&4&0&0\\0&0&0&3&0\\0&0&0&0&2\end{bmatrix}. \tag{1.3}$$

Weighted graph $G(V, E, W)$ is a graph with real edge weights given by $w : E \to \mathbb{R}$. Here, the weight $w(e)$ of an edge e indicates the similarity of the incident vertices, and a missing edge corresponds to zero similarity. The ***weighted adjacency matrix*** $\boldsymbol{W}(G)$ of the graph $G(V, E, W)$ is defined by

$$\boldsymbol{W}_{ij}:=\begin{cases}w(e) & \text{if } e=(i,j)\in E\\ 0 & \text{otherwise.}\end{cases}. \tag{1.4}$$

The weight matrix $\boldsymbol{W}(G)$ can be, for instance, the k-nearest neighbour matrix $\boldsymbol{W}(G)_{ij} = 1$ if and only if vertex v_i is among the k-nearest neighbours of v_j or vice versa, and is 0 otherwise. Another typical weight matrix is given by the Gaussian kernel of width σ

$$\boldsymbol{W}(G)_{ij} = e^{-\frac{\left\|v_i - v_j\right\|^2}{2\sigma^2}}. \tag{1.5}$$

Then the ***degree matrix*** for weighted graph $\boldsymbol{D}(G)$ is defined by

$$\boldsymbol{D}(G)_{i,i} := \sum_j \boldsymbol{W}(G)_{ij} \tag{1.6}$$

The graph Laplacian $\boldsymbol{L}(G)$ is defined in two different ways. The ***normalized graph Laplacian*** is

$$\boldsymbol{L}(G) := \boldsymbol{I} - \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{W} \boldsymbol{D}^{-\frac{1}{2}}, \tag{1.7}$$

and the ***unnormalized graph Laplacian*** is

$$\boldsymbol{L}(G) := \boldsymbol{D} - \boldsymbol{W}. \tag{1.8}$$

Let us consider an example to understand the graph Laplacian of the graph that is shown in figure 1.3. Suppose $\boldsymbol{f} : V \to \mathbb{R}$ is a real-valued function on the set of the vertices of graph $G(V, E)$ such that it

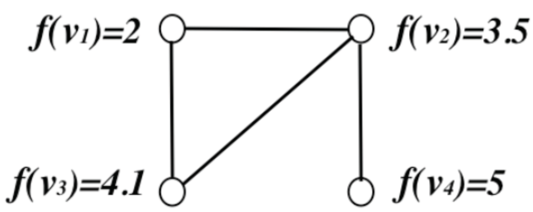

Figure 1.3: Real-Valued Functions on a Graph

assigns a real number to each graph vertex. Therefore, $\boldsymbol{f} = (f(v_1), f(v_2), \cdots, f(v_n))^T \in \mathbb{R}^n$ is a vector indexed by the vertices of graph. Its adjacency matrix is

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}. \tag{1.9}$$

Hence, the eigenvectors of the adjacency matrix, $\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x}$, can be viewed as eigenfunctions $\boldsymbol{A}\boldsymbol{f} = \lambda \boldsymbol{f}$. The adjacency matrix can be viewed as an operator

$$\begin{gathered} \boldsymbol{g} = \boldsymbol{A}\boldsymbol{f} \\ g(i) = \sum_{i \sim j} f(j), \end{gathered} \tag{1.10}$$

and it can also be viewed as a quadratic form

$$\boldsymbol{f}^T \boldsymbol{A} \boldsymbol{f} = \sum_{e_{ij}} f(i) f(j). \tag{1.11}$$

Assume that each edge in the graph have an arbitrary but fixed orientation, which is shown in figure 1.4.

Then the incidence matrix of the graph is

$$\boldsymbol{Q} = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix}. \tag{1.12}$$

Therefore the co-boundary mapping of the graph $\boldsymbol{f} \rightarrow \boldsymbol{Qf}$ implies $(\boldsymbol{Qf})(e_{ij}) = f(v_j) - f(v_i)$ is

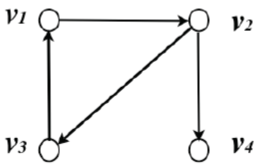

Figure 1.4: Orientation of the Graph

$$\begin{bmatrix} f(2) - f(1) \\ f(1) - f(3) \\ f(3) - f(2) \\ f(4) - f(2) \end{bmatrix} = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} f(1) \\ f(2) \\ f(3) \\ f(4) \end{bmatrix}. \tag{1.13}$$

If we let

$$\boldsymbol{L} = \boldsymbol{Q}^T \boldsymbol{Q}, \tag{1.14}$$

then we have

$$(\boldsymbol{Lf})(v_i) = \sum_{v_i \sim v_j} [f(v_i) - f(v_j)]. \tag{1.15}$$

Hence, the connection between the Laplacian and the adjacency matrices is

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A} = \begin{bmatrix} 2 & -1 & -1 & 0 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 2 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix}, \tag{1.16}$$

where the degree matrix $\boldsymbol{D}$ is

$$\boldsymbol{D} = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{1.17}$$

If we consider undirected weighted graphs, which is each edge e_{ij} is weighted by w_{ij}, then the Laplacian as an operator is

$$(\boldsymbol{Lf})(v_i) = \sum_{v_i \sim v_j} w_{ij} [f(v_i) - f(v_j)]. \tag{1.18}$$

Its quadratic form is

$$\boldsymbol{f}^T \boldsymbol{Lf} = \frac{1}{2} \sum_{e_{ij}} w_{ij} [f(v_i) - f(v_j)]^2. \tag{1.19}$$

The intuition behind a Laplacian matrix is the following. If, for instance, we apply the Laplacian operator of formula 1.16 to the real-valued functions $\boldsymbol{f} = (f(v_1), f(v_2), f(v_3), f(v_4))^T$ of the set of the vertices of graph $G(V, E)$, we have

$$(\boldsymbol{Lf})(v_i) = \begin{bmatrix} 2 & -1 & -1 & 0 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 2 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} f(v_1) \\ f(v_2) \\ f(v_3) \\ f(v_4) \end{bmatrix}. \tag{1.20}$$

For simplicity, let us only look at the first element

$$\begin{aligned} (\boldsymbol{Lf})(v_i)_1 &= \begin{bmatrix} 2 & -1 & -1 & 0 \end{bmatrix} \begin{bmatrix} f(v_1) \\ f(v_2) \\ f(v_3) \\ f(v_4) \end{bmatrix} \\ &= 2f(v_1) - f(v_2) - f(v_3) \\ &= -[f(v_2) - 2f(v_1) + f(v_3)] \\ &= -[f(v_2) - f(v_1) - f(v_1) + f(v_3)] \end{aligned} \tag{1.21}$$

If we label $f(v_1) = f_k$, $f(v_2) = f_{k+1}$, and $f(v_3) = f_{k-1}$, then we have

$$(\boldsymbol{Lf})(v_i)_1 = -[f_{k+1} - 2f_k + f_{k-1}]. \tag{1.22}$$

We recall that the second order derivative can be approximated by

$$\begin{aligned} f''(x) &= \frac{\frac{f(x+\Delta x)-f(x)}{\Delta x} - \frac{f(x)-f(x-\Delta x)}{\Delta x}}{\Delta x} \\ &= \frac{f(x+\Delta x) - 2f(x) + f(x-\Delta x)}{(\Delta x)^2}. \end{aligned} \tag{1.23}$$

Hence, we observe that the graph Laplacian is the negative numerator of the finite difference approximation of the second derivative. The Laplacian matrix $\boldsymbol{L}$ is symmetric and positive semi-definite, and it has n non-negative, real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. The number of 0 eigenvalues of the Laplacian matrix $\boldsymbol{L}$ is the number of connected components, because each connected component forms a block in the Laplacian matrix that only has edges within itself, and each block is the Laplacian for a small connected component and it has one zero eigenvalue, so the number of zeros is the number of blocks is the number of connected components.

1.2 Basic Idea for Semi-Supervised Learning over Graphs

The common denominator of semi-supervised learning algorithms over graphs is that the data are represented by the vertices of a graph, the edges of which are labelled with the pairwise distances of the incident vertices, and a missing edge corresponds to infinite distance. Most graph methods refer to the graph by utilizing the graph Laplacian.

Given the graph $G(V, E, W)$, a simple idea for semi-supervised learning label propagation is to propagate labels on the graph. Starting with vertices $1, 2, \cdots, l$ labelled l with their known label1 or -1 and nodes $l+1, \cdots, n$ labelled with 0, each vertex starts to propagate its label to its neighbours, and the process is repeated until convergence. Bengio, Delalleau and Roux (2006) proposed a label propagation scheme based on the Jacobi iterative method for linear systems. Estimated labels on both labelled and unlabelled data are denoted by $\hat{\boldsymbol{Y}} = \left(\hat{\boldsymbol{Y}}_l, \hat{\boldsymbol{Y}}_u\right)$, where where $\hat{\boldsymbol{Y}}_l$ may be allowed to differ from the given labels $\boldsymbol{Y}_l = (y_1, y_2, \cdots, y_l)$. It uses an additional regularization term ϵ for better numerical stability, which is shown in algorithm 1.1.

Zhou et al. (2004) proposed a similar label propagation algorithm, see algorithm 1.2, that uses graph Laplacian. At each step a vertex i receives a contribution from its neighbours j and an additional small

Algorithm 1.1 Jacobi Iterative Label Propagation Algorithm

Compute weight matrix $\boldsymbol{W}$ from formula 1.5 such that $\boldsymbol{W}_{ii} = 0$
Compute the diagonal degree matrix $\boldsymbol{D}$ by $\boldsymbol{D}_{ii} = \sum_j \boldsymbol{W}_{ij}$
Choose a parameter $\alpha \in (0, 1)$ and a small $\epsilon > 0$
$\mu = \frac{\alpha}{1-\alpha} \in (0, +\infty)$
Compute the diagonal matrix $\boldsymbol{A}$ by $\boldsymbol{A}_{ii} = \boldsymbol{I}_l(i) + \mu \boldsymbol{D}_{ii} + \mu\epsilon$
Initialize $\hat{\boldsymbol{Y}}^{(0)} = (y_1, y_2, \cdots, y_l, 0, 0, \cdots, 0)$
Iterate

$$\hat{\boldsymbol{Y}}^{(t+1)} = \boldsymbol{A}^{-1}\left(\mu \boldsymbol{W}\hat{\boldsymbol{Y}}^{(t)} + \hat{\boldsymbol{Y}}^{(0)}\right)$$

until convergence to $\hat{\boldsymbol{Y}}^{(\infty)}$
Label point v_i by the sign of $\hat{y}_i^{(\infty)}$

contribution given by its initial value.

Algorithm 1.2 Graph Laplacian Label Propagation Algorithm

Compute weight matrix $\boldsymbol{W}$ from formula 1.5 for $i \neq j$ and $\boldsymbol{W}_{ii} = 0$
Compute the diagonal degree matrix $\boldsymbol{D}$ by $\boldsymbol{D}_{ii} = \sum_j \boldsymbol{W}_{ij}$
Compute the normalized graph Laplacian $\boldsymbol{L} = \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{W}\boldsymbol{D}^{-\frac{1}{2}}$
Initialize $\hat{\boldsymbol{Y}}^{(0)} = (y_1, y_2, \cdots, y_l, 0, 0, \cdots, 0)$
Choose a parameter $\alpha \in [0, 1]$
Iterate

$$\hat{\boldsymbol{Y}}^{(t+1)} = \alpha \boldsymbol{L}\hat{\boldsymbol{Y}}^{(t)} + (1-\alpha)\,\hat{\boldsymbol{Y}}^{(0)}$$

until convergence to $\hat{\boldsymbol{Y}}^{(\infty)}$
Label point v_i by the sign of $\hat{y}_i^{(\infty)}$

Bibliography

Bapat, Ravindra B. (2014). *Graphs and Matrices*. London, UK: Springer-Verlag. 193 pp. ISBN: 9781447165682 (cit. on p. 1).

Bengio, Yoshua, Olivier Delalleau and Nicolas Le Roux (2006). *Semi-Supervised Learning*. Cambridge, Massachusetts, USA: MIT Press. 528 pp. ISBN: 9780262033589 (cit. on p. 5).

Herbster, Mark, Massimiliano Pontil and Lisa Wainer (2005). "Online Learning Over Graphs". In: *22nd International Conference on Machine Learning*.

Mohri, Mehryar, Afshin Rostamizadeh and Ameet Talwalkar (2012). *Foundations of Machine Learning*. Cambridge, Massachusetts, USA: The MIT Press. 432 pp. ISBN: 9780262018258.

Zhou, Dengyong, Olivier Bousquet, Thomas Navin Lal, Jason Weston and Bernhard Schölkopf (2004). *Advances in Neural Information Processing Systems 16*. Cambridge, Massachusetts, USA: MIT Press. 1728 pp. ISBN: 9780262201520 (cit. on p. 5).

Zhu, Xiaojin and Zoubin Ghahramani (2002). "Learning from Labelled and Unlabelled Data with Label Propagation". In: *Technical Report CMU-CALD-02-107, Carnegie Mellon University*.