Stat260/CS294: Spectral Graph Methods **Lecture 15 - 03/12/2015**

Lecture: Some Practical Considerations (4 of 4)

Lecturer: Michael Mahoney *Scribe: Michael Mahoney*

Warning: these notes are still very rough. They provide more details on what we discussed in class, but there may still be some errors, incomplete/imprecise statements, etc. in them.

15 More on diffusions and semi-supervised graph construction

Today, we will continue talking about the connection between non-linear dimensionality reduction methods and diffusions and random walks. We will also describe several methods for constructing graphs that are "semi-supervised," in the sense that that there are labels associated with some of the data points, and we will use these labels in the process of constructing the graph. These will give rise to similar expressions that we saw with the unsupervised methods, although there will be some important differences.

15.1 Introduction to diffusion-based distances in graph construction

Recall from last time that we have our graph G, and if we run a random walk to the asymptotic state, then we converge to the leading non-trivial eigenvector. Here, we are interested in looking at the pre-asymptotic state, i.e., at a time t such that $1 \ll t \ll \infty$, and we want to define similarity between vertex x and z, i.e., a metric between nodes x and z, such that x and z are close if $P_t(x, \cdot)$ and $P_t(z, \cdot)$ are close. Although there are other distance notions one could use, we will work with the ℓ_2 distance. In this setting, the ℓ_2 distance is defined as

$$\begin{aligned} D_t^2(x,z) &= ||P_t(x,\cdot) - P_t(z,\cdot)||_{1/\phi_0}^2 \\ &= \sum_y \frac{(P_t(x,y) - P_t(z,y))^2}{\phi_0(y)}. \end{aligned} \tag{1}$$

Suppose the transition matrix P has q left and right eigenvectors and eigenvalues $|\lambda_0| \geq |\lambda_1| \geq \ldots \geq |\lambda_n| \geq 0$ s.t.

$$\begin{aligned} \phi_j^T P &= \lambda_j \phi_j^T \\ P\psi_j &= \lambda \psi_j, \end{aligned}$$

where we note that $\lambda_0 = 1$ and $\psi_0 = \vec{1}$. Normalize s.t. $\phi_k \psi_\ell = \delta_{k\ell}$ so that

$$\begin{aligned} \|\phi_\ell\|_{1/\phi_0}^2 &= \sum_x \frac{\phi_\ell^2(x)}{\phi_0(x)} = 1 \\ \|\psi_\ell\|_{\phi_0}^2 &= \sum_x \psi_\ell^2(x)\phi_0(x) = 1, \end{aligned}$$

i.e., normalize the left (resp. right) eigenvector of P w.r.t. $1/\phi_0$ (resp. ϕ_0). If $P_t(x,y)$ is the kernel of the t^{th} iterate of P, then we have the following spectral decomposition:

$$P_t(x,y) = \sum_j \lambda_j^t \psi_j(x)\phi_j(y). \tag{2}$$

(This is essentially a weighted PCA/SVD of P^t, where as usual the first k terms provide the "best" rank-k approximation, where the "best" is w.r.t. $\|A\|^2 = \sum_x \sum_y \phi_0(x)A(x,y)^2\phi_0(y)$.) If we insert Eqn. (2) into Eqn. (1), then one can show that the L_2 distance is:

$$\begin{aligned} D_t^2(x,z) &= \sum_{j=1}^{n-1} \lambda_j^{2t}(\psi_j(x) - \psi_j(z))^2 & (3)\\ &\approx \sum_{j=1}^{k} \lambda_j^{2t}(\psi_j(x) - \psi_j(z))^2 & (4)\\ &\approx \sum_{j=1}^{k} (\psi_j(x) - \psi_j(z))^2. & (5) \end{aligned}$$

Eqn. (3) says that this provides a legitimate distance, and establishing this is on HW2. The approximation of Eqn. (4) holds if the eigenvalues decay quickly, and the approximation of Eqn. (5) holds if the large eigenvalues are all roughly the same size. This latter expression is what is provided by LE.

15.2 More on diffusion-based distances in graph construction

Recall from last time that LE can be extended to include all of the eigenvectors and also to weighting the embedding by the eigenvalues. This gives rise to an embedding that is based on diffusions, sometimes it is called Diffusion Maps, that defines a distance in the Euclidean space that is related to a diffusion-based distance on the graph. In particular, once we choose k in some way, then here is the picture of the Diffusion Map:

$$\text{Map } \Psi_t : X \to \begin{pmatrix} \lambda_1^t \psi_1(x) \\ \vdots \\ \lambda_k^t \psi_k(x) \end{pmatrix}$$

and so

$$\begin{aligned} D_t^2(x,z) &\approx \sum_{j=1}^{k} \lambda_j^{2t}(\psi_j(x) - \psi_j(z))^2 \\ &\approx ||\psi_t(x) - \psi_t(z)||^2. \end{aligned}$$

This is a low-dimensional embedding, where the Euclidean distance in the embedded space corresponds to the "diffusion distance" in the original graph. Here is the picture of the relationships:

$$\begin{array}{ccc} G & \leftrightarrow & R^n \\ | & & | \\ \text{diffusion} & \leftrightarrow & ||\cdot||_2 \end{array}$$

Fact: This defines a distance. If you think of a `diffusion` notion of distance in a graph G, this identically equals the Euclidean distance $||\cdot||_2$ between $\Psi(i)$ and $\Psi(j)$. The diffusion notion of distance is related to the commute time between node i and node j. We will describe this next time when we talk about resistor networks.

In particular, Laplacian Eigenmaps chooses $k = k^*$, for some fixed k^* and sets $t = 0$. Under certain nice limits, this is close to the Laplace-Beltrami operator on the hypothesized manifold. But, more generally, these results will hold even if the graph is not drawn from such a nice setting.

- None of this discussion assumes that the original vector data come from low-dimensional manifolds, although one does get a nice interpretation in that case.
- It is fair to ask what happens with spectral decomposition when there is not manifold, e.g., a star graph or a constant-degree expander. Many of the ideas wetland about earlier in the semester would be relevant in this case.

15.3 A simple result connecting random walks to NCUT/conductance

There are connections with graph partitioning, but understanding the connection with random walkers opens up a lot of other possibilities. Here, we describe one particularly simple result: that a random walker, starting in the stationary distribution, goes between a set and its complement with probability that depends on the NCUT of that set. Although this result is simple, understanding it will be good since it will open up several other possibilities: what if one doesn't run a random walk to the asymptotic state; what if one runs a random walk just a few steps starting from an arbitrary distribution; when does the random walk provide regularization; and so on?

Here, we provide an interpretation for the NCUT objective (and thus related to normalized spectral clustering as well as conductance): when minimizing NCUT, we are looking for a suit through the graph s.t. the random walk seldom transitions from A to $\bar{A}$, where $A \subset V$. (This result says that conductance/NCUT is not actually looking for good cuts/partitions, but instead should be interpreted as providing bottlenecks to diffusion.)

Lemma 1. *Let $P = D^{-1}W$ be a random walk transition matrix. Let $(X_t)_{t\in\mathbb{Z}^+}$ be the random walk starting at $X_0 = \pi$, i.e., starting at the stationary distribution. For disjoint subsets $A, B \subset V$, let $\mathbb{P}[B|A] = \mathbb{P}[X_1 \in B|X_0 \in A]$. Then,*

$$NCUT\left(A, \bar{A}\right) = \mathbb{P}\left[\bar{A}|A\right] + \mathbb{P}\left[A|\bar{A}\right].$$

Proof. Observe that

$$\begin{aligned}
\mathbb{P}\left[X_0 \in A \text{ and } X_1 \in B\right] &= \sum_{i\in A, j\in B} \mathbb{P}\left[X_0 = i \text{ and } X_1 = j\right] \\
&= \sum_{i\in A, j\in B} \pi_i P_{ij} \\
&= \sum_{i\in A, j\in B} \frac{d_i}{\mathrm{Vol}(V)} \frac{W_{ij}}{d_i} \\
&= \frac{1}{\mathrm{Vol}(V)} \sum_{i\in A, j\in B} W_{ij}
\end{aligned}$$

From this, we have that

$$\begin{aligned}\mathbb{P}\left[X_1 \in B | X_0 \in A\right] &= \frac{\mathbb{P}\left[X_0 \in A \text{ and } X_1 \in B\right]}{\mathbb{P}\left[X_0 \in A\right]} \\ &= \left(\frac{1}{\mathrm{Vol}(V)} \sum_{i \in A, j \in B} W_{ij}\right) \left(\frac{\mathrm{Vol}(A)}{\mathrm{Vol}(V)}\right)^{-1} \\ &= \frac{1}{\mathrm{Vol}(V)} \sum_{i \in A, j \in B} W_{ij}\end{aligned}$$

The lemma follows from this and the definition of NCUT. □

15.4 Overview of semi-supervised methods for graph construction

Above, we constructed graphs/Laplacians in an unsupervised manner. That is, there were just data points that were feature vectors without any classification/regression labels associated with them; and we constructed graphs from those unlabeled data points by using various NN rules and optimizing objectives that quantified the idea that nearby data points should be close or smooth in the graph. The graphs were then used for problems such as data representation and unsupervised graph clustering that don't involve classification/regression labels, although in some cases they were also used for classification or regression problems.

Here, we consider the situation in which some of the data have labels and we want to construct graphs to help make predictions for the unlabeled data. In between the extremes of pure unsupervised learning and pure supervised learning, there are semi-supervised learning, transductive learning, and several other related classes of machine learning methods. It is in this intermediate regime where using labels for graph construction is most interesting.

Before proceeding, here are a few things to note.

- If unlabeled data and labeled data come from the same distribution, then there is no difference between unlabeled data and test data. Thus, this transductive setup amounts to being able to see the test data (but not the labels) before doing training. (Basically, the transductive learner can look at all of the data, including the test data with labels and as much training data as desired, to structure the hypothesis space.)
- People have used labels to augment the graph construction process in an explicit manner. Below, we will describe an example of how this is done implicitly in several cases.
- We will see that linear equations of a certain form that involve Laplacian matrices will arise. This form is rather natural, and it has strong similarities with what we saw previously in the unsupervised setting. In addition, it also has strong similarities with local spectral methods and locally-biased spectral methods that we will get to in a few weeks.

We will consider three approaches (those of Joachims, ZGL, and Zhao et al.). Each of these general approaches considers constructing an augmented graph, with extra nodes, s and t, that connect to the nodes that have labels. Each can be understood within the following framework. Let

$L = D - A = B^T C B$, where B is the unweighted edge-incidence matrix. Recall, then, that the s,t-mincut problem is:

$$\min_{x \text{ s.t. } x_s=1, x_t=0} \|Bx\|_{C,1} = \min_{x \text{ s.t. } x_s=1, x_t=0} \sum_{(u,v)\in E} C_{(u,v)} |x_u - x_v|.$$

The ℓ_2 minorant of this s,t-mincut problem is:

$$\min_{x \text{ s.t. } x_s=1, x_t=0} \|Bx\|_{C,2} = \min_{x \text{ s.t. } s=1, x_t=0} \left(\sum_{(u,v)\in E} C_{(u,v)} |x_u - x_v|^2 \right)^{1/2}.$$

This latter problem s equivalent to:

$$\min_{x \text{ s.t. } x_s=1, x_t=0} \frac{1}{2} \|Bx\|_{C,2}^2 = \min_{x \text{ s.t. } x_s=1, x_t=0} \sum_{(u,v)\in E} C_{(u,v)} |x_u - x_v|^2 = \min_{x \text{ s.t. } x_s=1, x_t=0} x^T L x.$$

The methods will construct Laplacian-based expressions by considering various types of s-t mincut problems and then relaxing to the associated ℓ_2 minorant.

15.5 Three examples of semi-supervised graph construction methods

The Joachims paper does a bunch of things, e.g., several engineering heuristics that are difficult to relate to a general setting but that no doubt help the results in practice, but the following is essentially what he does. Given a graph and labels, he wants to find a vector $\vec{y}$ s.t. it satisfies the bicriteria:

1. it minimizes $y^T L y$, and
2. it has values
$$\begin{cases} 1 & \text{for nodes in class } j \\ -1 & \text{for nodes not in class } j \end{cases}$$

What he does is essentially the following. Given $G = (V, E)$, he adds extra nodes to the node set:

$$\begin{cases} s & \text{with the current class} \\ t & \text{with the other class} \end{cases}$$

Then, there is the issue about what to add as weights when those nodes connect to nodes in V. The two obvious weights are 1 (i.e., uniform) or equal to the degree. Of course, other choices are possible, and that is a parameter one could play with. Here, we will consider the uniform case. Motivated by problem with mincut (basically, what we described before, where it tends to cut off small pieces), Joachims instead considers NCUT and so arrives at the following problem:

$$Y = (D_S + L)^{-1} S,$$

where D_S is a diagonal matrix with the row sums of S on the diagonal, and S is a vector corresponding to the class connected to the node with label s. Note that since most of the rows of S

equal zero (since they are unlabeled nodes) and the rest have only 1 nonzero, D_S is a diagonal indicator vector that is sparse.

ZGL is similar to Joachims, except that they strictly enforce labeling on the labeled samples. Their setting is that they are interpreting this in terms of a Gaussian random field on the graph (which is like a NN method, except that nearest labels are captures in terms of random vectors on G). They provide hard constraints on class labels. In particular, they want to solve

$$\begin{array}{ll} \min\limits_{y} & \frac{1}{2} y^T L y \\ \text{s.t.} & y_i = \begin{cases} 1 & \text{node } i \text{ is labeled in class } j \\ 0 & \text{node } i \text{ is labeled in another class} \\ \text{free} & \text{otherwise} \end{cases} \end{array}$$

This essentially involves constructing a new graph from G ,where the labels involve adding extra nodes, s and t, where s links to the current class and t links to the other class. For ZGL, weights $= \infty$ between s and the current class as well as between t and the other class.

Zhao, et al. extend this to account explicitly for the bicriteria that:

1. nearby points should have the same labels; and
2. points on the same structure (clusters of manifold) have the same label.

Note that Point 1 is a "local" condition, in that it depends on nearby data points, while Point 2 is a "global" condition, in that it depends on large-scale properties of clusters. The point out that different semi-supervised methods take into account these two different properties and weight them in different ways. Zhao et al. try to take both into account by "iteratively" spreading ZGL to get a classification function that has local and global consistency properties and is sufficiently smooth with respect to labeled and unlabeled data.

In more detail, here is the Zhao et al. algorithm.

1. Form the affinity matrix W from an rbf kernel.
2. Construct $\tilde{W} = D^{-1/2} W D^{-1/2}$.
3. Iterate $Y(t+1) = \alpha \tilde{W} Y + (1-\alpha)\, S$, where $\alpha \in (0,1)$ is a parameter, and where S is a class label vector.
4. Let Y^* be the limit, and output it.

Note that $\tilde{Y}^* = (1-\alpha)\left(1 - \alpha \tilde{W}\right) S$. Alternatively,

$$Y = (L + \alpha D)^{-1} S = (D - \beta A)^{-1} S,$$

where Y is the prediction vector, and where S is the matrix of labels. Here, we have chosen/defined $\alpha = \frac{1-\beta}{\beta}$ to relate the two forms.

Then the "mincut graph" has $G = (V, E)$, with nodes s and t such that

$$\begin{cases} s & \text{connects to each sample labeled with class } j \text{ with weight } \alpha \\ t & \text{connects to } \textit{all} \text{ nodes in } G \text{ (except } s) \text{ with weight } \alpha(d_i - s_i) \end{cases}$$

The ℓ_2 minorant of this mincut graph has:

$$\min_{x \text{ s.t. } x_s=1, x_t=0} \frac{1}{2}\|Bx\|^2_{C,2} = \min_{x \text{ s.t. } x_s=1, x_t=0} \frac{1}{2} \begin{pmatrix} 1 \\ y \\ 0 \end{pmatrix}^T \begin{pmatrix} \alpha e^T s & -\alpha s^T & 0 \\ -\alpha s & \alpha D + L & \alpha(d-s) \\ 0 & -\alpha(d-s)^T & \alpha e^T (d-s) \end{pmatrix} \begin{pmatrix} 1 \\ y \\ 0 \end{pmatrix}.$$

In this case, y solves $(\alpha D + L)\, y = \alpha s$.

We will see an equation of this form below in a somewhat different context. But for the semi-supervised learning context, we can interpret this as a class-specific smoothness constraint. To do so, define

$$A(Y) = \frac{1}{2} \left(\sum_{ij=1}^{n} W_{ij} \| \frac{1}{\sqrt{D_{ii}}} Y_i - \frac{1}{\sqrt{D_{jj}}} Y_j \|^2 - \mu \sum_{j=1}^{n} \|Y_i - S_i\|^2 \right)$$

to be the "cost" associated with the prediction Y. The first term is a "smoothness constraint," which is the sum of local variations, and it reflects that a good classification should not change much between nearby points. The second term is a "fitting constraint," and it says that a good classification function should not change much from the initial assignment provided by the labeled data. The parameter μ givers the interaction between the two terms. To see how this gives rise to the previous expression, observe that

$$\frac{\partial Q(Y)}{\partial Y}|_{Y=Y^*} = Y^* - WY^* + \mu\,(Y^* - S) = 0,$$

from which it follows that

$$Y^* - \frac{1}{1+\mu} WY^* - \frac{\mu}{1+\mu} S = 0.$$

If we define $\alpha = \frac{1}{1+\mu}$ and $\beta = \frac{\mu}{1+\mu}$, so that $\alpha + \beta = 1$, then we have that

$$(I - \alpha W)\, Y^* = \beta S,$$

and thus that

$$Y^* = \beta\,(1 - \alpha W)^{-1}\, S.$$