Lecture 17: Semi-supervised learning on graphs

We introduced the idea of diffusion maps last lecture. The mathematical background & analysis, specially in connection to diffusion processes & elliptic operators is quite deep & contemporary. (see ref. on Canvas)

Ignoring the mathematical details, the construction of diffusion maps is quite simple:

Step 1) Construct Graph $G = \{X, W\}$

Step 2) Compute eigenpairs $\{\lambda_j, \underline{q}_j\}$ of $D^{-1}W$

(equivalently eig pairs $\{\sigma_j, \underline{q}_j\}$ of L_{RW} set $\lambda_j = 1 - \sigma_j$)

Step 3) Define diff. map

$$\mathcal{F}_t : X \to \mathbb{R}^n, \quad \mathcal{F}_t(\underline{x}_i) = (\lambda_1^t q_1(i), \ldots, \lambda_n^t q_n(i))^T$$

Of course, thinking about diff. maps this way we immediately see that the choice of L_{RW} is not crucial & we can define such maps for other graph Laplacians.

Let $L = I - A$ be a graph Laplacian with an arbitrary adjacency matrix (normalized, RW, etc) with eigenpairs $\{\sigma_j, \underline{q}_j\}$, set $\lambda_j = 1 - \sigma_j$. Define

$$\mathcal{F}_t(\underline{x}_i) = \left(\lambda_1^t q_1(i), \ldots, \lambda_n^t q_n(i)\right)$$

* Understanding whether your choice of A leads to a diff. process is an interesting theoretical question.

Now observe that given a diffusion map $\mathcal{F}_t$ we can define a kernel

$$K_t : X \times X \to \mathbb{R}, \quad K_t^n(\underline{x}_1, \underline{x}_\ell) = \sum_{j=1}^{n} \lambda_j^{2t} q_j(i) q_j(\ell)$$

Recall $|X| = n$ Under some technical assptios, we can show that if $\underline{x}_i \overset{iid}{\sim} U(M)$ (some compact smooth manifold) Then $\lim_{n \to \infty} K_t^n \to K_t$ where K_t is a "Heat kernel" ie, it describes a diffusion process on our manifold M.

This suggests, the RKHS of kernels such as above, are good candidates for designing alg. that can automatically take advantage of an existing manifold M on which the data lies.

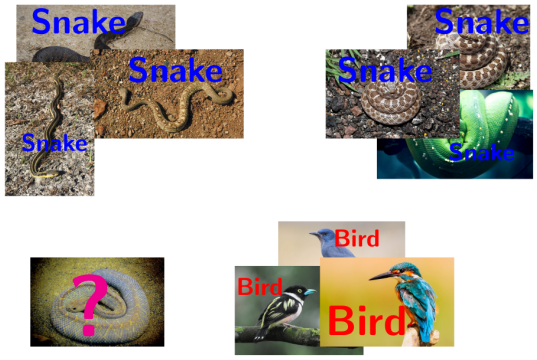
17.1 Semi-supervised learning with manifold regularization

We will now use the above intuition to introduce another approach to exploiting spectral geometric information of graph Laplacians in the context of semi-supervised Learning (SSL)

supervised learning

Train data $\{x_i, y_i\}_{i=1}^N$

predict $y_{N+1}(x_{N+1})$



Unsupervised learning

Given unlabelled data $\{x_i\}_{i=1}^N$ discover structure in X.

Given $\{x_i, y_i\}_{i=1}^N$ & $\{x_i\}_{N+1}^M$

Predict $y_i(x_i)$ for $i = N+1, \ldots, M$.

explore clusters & labels simultaneously

We will introduce an approach for SSL based on the so called manifold regularization technique of Belkin et al (2006) which is indeed nothing more than RKHS regression.

Let $X = \{\underline{x}_1, \dots, \underline{x}_M\} \subset \mathcal{M} \subset \mathbb{R}^d$ & suppose we are given (possibly noisy) labels $Y = (Y_1, \dots, Y_N) \in \{-1, +1\}^N$ where $N < M$ (typically we would like $N \ll M$!)

Goal: Predict $f(\underline{x}_i)$ for $i = N+1, \dots, M$

First, we need a model for the labels & a data misfit term as we did before in classification. For this we will use probit. Recall:

$$\text{(model)} \quad Y_i = \operatorname{sign}(f(\underline{x}_i) + \varepsilon_i), \quad \varepsilon_i \overset{iid}{\sim} \nu$$

$$\text{(Probit misfit)} \quad L(f; \underline{Y}) = -\sum_{j=1}^{N} \log \Psi(y_j f(\underline{x}_j))$$

Since our goal is to exploit possible clustering / geometric info in X to get better

labelling it is natural for us to build a graph on X.

Step 1) Build graph $G = \{X, W\}$

(prox. on k-NN graph with weight function $\eta : [0, \infty) \to [0, \infty)$ defined by user)

eg. $\eta(t) = \exp\left(\frac{-t^2}{2r^2}\right)$ & set $w_{ij} = \eta(\|\underline{x}_i - \underline{x}_j\|_2)$

Note, the choice of the radius $r > 0$ is crucial in getting a good graph.

A surprisingly good heuristic:

In most applications we simply compute the pairwise distances $\rho_{ij} = \|\underline{x}_i - \underline{x}_j\|_2$ & simply set r to be the 1/25 quantile of the ρ_{ij}'s. This can be further fine tuned by CV if needed.

Step 2) Define a graph Laplacian L on G

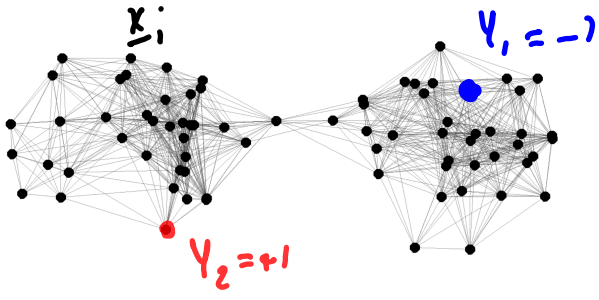
(your fav. normalization, eg

$$L = D - W$$

Step 3) Since we are only interested in labeling $\underline{x}_{N+1}, \dots, \underline{x}_M$ that are additional nodes on the graph we can think of the latent function f in the probit model as a function on the vertices of the graph

$$f : V \longrightarrow \mathbb{R}, \qquad f \equiv \underline{f} \in \mathbb{R}^M$$

That is $f(i) = f_i$.



Then we define the probit optimization problem

$$\underline{f}^* = \underset{\underline{f} \in \mathbb{R}^M}{\arg\min} \; -\sum_{j=1}^{N} \log \Psi(f_j, y_j) + \lambda \, \underline{f}^T C \, \underline{f}$$

where we take $C = (L + \tau^2 I)^{-\alpha}$ for parameters $\tau^2, \alpha > 0$.

- First, observe this is completely analogous to our formulation of Probit on general state spaces

$$f^* = \underset{f \in H}{\operatorname{argmin}} -\sum_{j=1}^{M} \log \Psi(f(x_j) y_j) + \lambda \|f\|_K^2$$

↑ RKHS Norm

By this observation we readily infer that $\underline{f}^T C \underline{f}$ must be some kind of an RKHS norm!

- In fact our kernel here turns out to be $K: U \times U \to \mathbb{R}$, $K(i,j) = C_{ij}$.

Then we can write down a rep. thm. (Next lecture).

- Suppose L is symm so we can write $L = U \Sigma U^T$ then $C = U(\Sigma + \tau^2 I)^{-\alpha} U$

that is $$C = \sum_{j=1}^{M} \frac{1}{(\sigma_j + \tau^2)^\alpha} \underline{u}_j \underline{u}_j^T$$

suppose there are P clusters in X then $\sigma_1 = \sigma_2 = \dots = \sigma_P = 0$ & $\sigma_{P+1} = G(1)$ then

$$C = \underbrace{\sum_{j=1}^{P} \frac{1}{\tau^{2\alpha}} \underline{u}_j \underline{u}_j^T}_{G(\tau^{-2\alpha})} + \underbrace{\sum_{j=P+1}^{M} \frac{1}{(\sigma_j + \tau^2)^\alpha} \underline{u}_j \underline{u}_j^T}_{G(1)}$$