Kernel-based Active Search on Graphs

1 Introducing AS on graphs

Here is the energy function used for AS:

$$E(f) = \sum_{i\in\mathcal{L}} (y_i - f_i)^2 D_{ii} + \lambda w_0 \sum_{i\in\mathcal{U}} (f_i - \pi)^2 D_{ii} + \sum_{i,j} (f_i - f_j)^2 A_{ii} \tag{1}$$

Here is the energy function rewritten using matrices, where f_L and f_U are the f-vector portions belonging to the labeled and unlabeled portions respectively (they have been rearranged WLOG):

$$E(f) = \begin{bmatrix} f_\mathcal{L} \\ f_\mathcal{U} \\ y \\ \pi \end{bmatrix}^T \left[\begin{array}{c|c} \begin{bmatrix} D_\mathcal{L} & 0 \\ 0 & \lambda w_0 D_\mathcal{U} \end{bmatrix} + \lambda(D-A) & \begin{bmatrix} -D_\mathcal{L} & 0 \\ 0 & -\lambda w_0 D_\mathcal{U} \end{bmatrix} \\ \hline \begin{bmatrix} -D_\mathcal{L} & 0 \\ 0 & -\lambda w_0 D_\mathcal{U} \end{bmatrix} & 0 \end{array}\right] \begin{bmatrix} f_\mathcal{L} \\ f_\mathcal{U} \\ y \\ \pi \end{bmatrix}$$

The minimizer is as follows (not proved here):

$$f^* = (I - A')^{-1} D' y' \tag{2}$$

where

$$A' = \begin{bmatrix} \frac{\lambda}{1+\lambda} I_\mathcal{L} & 0 \\ 0 & \frac{1}{1+w_0} I_\mathcal{U} \end{bmatrix} D^{-1} A, \quad D' = \begin{bmatrix} \frac{1}{1+\lambda} I_\mathcal{L} & 0 \\ 0 & \frac{w_0}{1+w_0} I_\mathcal{U} \end{bmatrix}, \quad y' = \begin{bmatrix} y_\mathcal{L} \\ \pi \end{bmatrix}$$

If we set $B = \begin{bmatrix} \frac{\lambda}{1+\lambda} I_\mathcal{L} & 0 \\ 0 & \frac{1}{1+w_0} I_\mathcal{U} \end{bmatrix}$, we have that $A' = BD^{-1}A, \quad D' = I - B$
Thus, we have our optimal solution:

$$f^* = (I - BD^{-1}A)^{-1}(I - B)y' \tag{3}$$

2 Kernel AS – Linear Kernel as similarity

Say $A = X^T X$ where $X = [x_1 \dots x_n]$, with n data points and r features.
Then $D = diag(X^T X \mathbb{1})$. (Precomputed in $O(nr)$).

Thus,

$$f^* = (I - \overline{B} X^T X)^{-1} q \tag{4}$$

where $\overline{B} = BD^{-1}, q = (I - B)y'$.

Here, we use the Kailath variant of the matrix inverse lemma:

$$(A+BC)^{-1} = A^{-1} - A^{-1}B(I+CA^{-1}B)^{-1}CA^{-1}$$

We have:

$$f^* = (I - \overline{B}X^TX)^{-1}q = (I + (\overline{B}X^T)(I - X\overline{B}X^T)^{-1}X)q$$

Thus,

$$f^* = q + \overline{B}X^T(I - X\overline{B}X^T)^{-1}Xq \tag{5}$$

The main power obtained from this representation is that the inverse is now over an $r \times r$ matrix as opposed to an $n \times n$ matrix. The inverse can precomputed in $O(r^2n + r^3)$. So the entire precomputation is in $O(r^2n)$ assuming $n > r$.
We want to compute the updates in $O(r^2 + nr)$.

2.1 Updates to f

We have precomputed $(I - X\overline{B}X^T)^{-1}$. One element in $\overline{B}$ changes.

$$\overline{B}' = \overline{B} - \gamma e_i e_i^T$$

where e_i is the i^{th} standard basis vector.
Let $K = (I - X\overline{B}X^T)$.
Then,

$$K' := I - X\overline{B}'X^T$$

$$= K + \gamma X e_i e_i^T X^T$$

$$= K + \gamma x_i x_i^T$$

Here, $\gamma = -\left(\frac{\lambda}{1+\lambda} - \frac{1}{1+w0}\right) D_{ii}^{-1}$.

Woodbury's Matrix inversion formula:

$$(A+UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}$$

Using this, we have:

$$K'^{-1} = K^{-1} - K^{-1}(\gamma x_i)(1 + \gamma x_i^T K^{-1} x_i)^{-1} x_i^T K^{-1}$$

$$= K^{-1} - \frac{\gamma K^{-1} x_i x_i^T K^{-1}}{1 + \gamma x_i^T K^{-1} x_i}$$

Thus,

$$K'^{-1} = K^{-1} - \frac{\gamma (K^{-1}x_i)(K^{-1}x_i)^T}{1 + \gamma x_i^T K^{-1} x_i} \tag{6}$$

Further, only one element in q changes. $q_i' = y_i \frac{1}{1+\lambda}$
Thus, with the update to the inverse, $f^* = q' + \overline{B}' X^T K'^{-1} X q'$. This takes $O(rn)$.

2.2 Impact factor computation

This is a continuation of Kyle's notes on computing the Impact Factor (IM).
A few points to note: I use the variable $P = SD$ where Kyle uses $B = SD$. S is some diagonal matrix which defines the relative weights of the edges to pseudo-nodes from labeled or unalebed nodes.
From Kyle's notes, we have the change in f given node i is chosen to be labeled:

$$\Delta f(i) = [(y_i - \pi)P^k_{i,i} + \delta P(y_i - f^{k+1}_i)]M^{-1}_{.,i} \tag{7}$$

where $M = D + P^k - A$, $\delta P = P^{k+1}_{i,i} - P^k_{i,i}$.
Fixing $y_i = 1$ to compute the IM, we have:

$$\begin{aligned}\Delta \widetilde{f}(i) &= [P^k_{i,i} - \pi P^k_{i,i} + \delta P - \delta P \widetilde{f}^{k+1}_i]M^{-1}_{.,i} \\ &= [P^{k+1}_{i,i} - \pi P^k_{i,i} - \delta P \widetilde{f}^{k+1}_i]M^{-1}_{.,i}\end{aligned}$$

After solving for f^{k+1}_i and subtracting out f^k_i, we have that the i^{th} element of $\Delta \widetilde{f}(i)$ is:

$$\Delta \widetilde{f}_i(i) = \frac{(P^{k+1}_{i,i} - \pi P^k_{i,i} - \delta P f^k_i)M^{-1}_{i,i}}{1 + \delta P M^{-1}_{i,i}} \tag{8}$$

Now, in order to compute the IM, we need, for each unlabeled node i,
$\Delta F_i = \sum\limits_{i \in \mathcal{U}} \Delta \widetilde{f}_j$ where the label of i is now set to 1. In order to compute that, we first need the following:

$$\Delta \widetilde{f} = \begin{bmatrix} \Delta \widetilde{f}_1(1) \\ \vdots \\ \Delta \widetilde{f}_n(n) \end{bmatrix}$$

Given that, we can find

$$\Delta F = \left[\overrightarrow{L} - \pi \overrightarrow{U} - (\overrightarrow{L} - \overrightarrow{U}) \circ (f^k + \Delta \widetilde{f})\right] \circ M^{-1}u \tag{9}$$

where the $\overrightarrow{L} = \frac{1}{\lambda}D$, $\overrightarrow{U} = w_0 D$. We can use these because each element of ΔF assumes that we are labelling that index as positive. Thus, the $P^{k+1}_{i,i}$ value will always be that of a labeled node in this calculation. u is the vector whose entry i is 1 if it is unlabeled and 0 otherwise.
Then, since the IM is computed as a sum over all changes in the unlabeled node EXCEPT the node currently picked as a potential positive, we have that the final IM is:

$$IM^k = f^k \circ (\Delta F - \Delta \widetilde{f}) \tag{10}$$

2.2.1 Rewriting M^{-1}

We first show that $M^{-1} = (I - BD^{-1}A)^{-1}(I - B)P^{-1}$.

We know that $P = SD$ where $S = \begin{bmatrix} \frac{1}{\lambda} I_{\mathcal{L}} & 0 \\ 0 & w_0 I_{\mathcal{U}} \end{bmatrix}$ (since $\lambda = \frac{1-\eta}{\eta}$).

Now, consider the following:

$$\begin{aligned}
M^{-1} &= (D+P-A)^{-1} \\
&= (D(I+S)-A)^{-1} \\
&= ((I+S)-D^{-1}A)^{-1}D^{-1} \\
&= (I-(I+S)^{-1}D^{-1}A)^{-1}(I+S)^{-1}D^{-1} \\
&= (I-BD^{-1}A)^{-1}BD^{-1} \qquad \text{(It can be easily verified that } B^{-1}=I+S) \\
&= (I-BD^{-1}A)^{-1}(I-B)S^{-1}D^{-1} \qquad \text{(It can be easily verified that } S=B^{-1}(I-B))) \\
&= (I-BD^{-1}A)^{-1}(I-B)P^{-1}
\end{aligned}$$

Making use of $A = X^TX$, we have that:

$$M^{-1} = (I + (\overline{B}X^T)(I - X\overline{B}X^T)^{-1}X)(I-B)P^{-1} \tag{11}$$

Most of this is already being computed in our updates.

2.2.2 Computing ΔF given $\Delta\widetilde{f}$

Given $\Delta\widetilde{f}$, we can compute $\left[\overrightarrow{L} - \pi\overrightarrow{U} - (\overrightarrow{L} - \overrightarrow{U}) \circ (f^k + \Delta\widetilde{f})\right]$ in $O(n)$ time as it is just sums or element-wise multiplications.
Further, $M^{-1}u$ can be computed as f is.

- $z = (I-B)P^{-1}u$ changes only one element each iteration and can be updated.
- $(I+(\overline{B}X^T)(I-X\overline{B}X^T)^{-1}X)z$ can then be computed in $O(rn)$ time by cascading the matrix-vector multiplication in. There is never an $\Omega(n^2)$ operation done here.

Thus, these operations are still within the time constraints of the original algorithm.

2.2.3 Computing $\Delta\widetilde{f}$ via updates

We want to compute:

$$\Delta\widetilde{f} = \begin{bmatrix} \Delta\widetilde{f}_1(1) \\ \vdots \\ \Delta\widetilde{f}_n(n) \end{bmatrix}$$

where each element is given by equation 8. This can be written as:

$$\Delta\widetilde{f} = \left[\overrightarrow{L} - \pi\overrightarrow{U} - (\overrightarrow{L} - \overrightarrow{U}) \circ f^k\right] \circ diag(M^{-1}) \circ diag\left(\frac{1}{1 + (\overrightarrow{L} - \overrightarrow{U})M^{-1}}\right) \tag{12}$$

After computing $diag(M^{-1})$, we can compute the rest of this in $O(n)$ time. Consider the following for $diag(M^{-1})$:

$$\begin{aligned}
diag(M^{-1}) &= diag((I + (\overline{B}X^T)(I - X\overline{B}X^T)^{-1}X)(I - B)P^{-1}) \\
&= diag(I + (\overline{B}X^T)(I - X\overline{B}X^T)^{-1}X) \circ diag((I - B)P^{-1}) \\
&= (1 + diag((\overline{B}X^T)(I - X\overline{B}X^T)^{-1}X)) \circ diag((I - B)P^{-1})
\end{aligned}$$

Thus,

$$diag(M^{-1}) = (1 + diag(\overline{B}) \circ diag(X^T(I - X\overline{B}X^T)^{-1}X)) \circ diag((I - B)P^{-1}) \tag{13}$$

This tells us that what we need to do is store and update
$J = diag(X^T(I - X\overline{B}X^T)^{-1}X)$ every iteration. The rest of the operations are either sums or point-wise multiplications which take $O(n)$ each iteration.
Here's how we update it:

- Initialize $j_i = x_i^T K^{-1} x_i$ and then $J = \begin{bmatrix} j_1 \\ \vdots \\ j_n \end{bmatrix}$ where $K = (I - X\overline{B}X^T)$ as defined before. This computation takes $O(nr^2)$ time.
- Then, as we update K, we can also update J. Here, t is the index of the point to be labeled.

  $$\begin{aligned}
  j_i' &= x_i^T K'^{-1} x_i \\
  &= x_i^T \left(K^{-1} - \frac{\gamma(K^{-1}x_t)(K^{-1}x_t)^T}{1 + \gamma x_t^T K^{-1} x_t} \right) x_i^T
  \end{aligned}$$

  Therefore,

  $$j_i' = j_i - c \cdot (x_i^T(K^{-1}x_t))^2 \tag{14}$$

  where $c = \dfrac{\gamma}{1 + \gamma x_t^T K^{-1} x_t}$ (computed in the updates to f). Since $(K^{-1}x_t)$ is also computed while updating f, updating each element j_i only takes $O(r)$ time.

Thus, computing the entire J vector only takes $O(nr)$ per iteration. Once we have J, we can compute $diag(M^{-1})$ in $O(n)$ time.

2.2.4 Computing IM after having both ΔF and $\Delta \widetilde{f}$

This is just a point-wise multiply or difference between vectors and takes $O(n)$ time every iteration. Thus, the entire procedure of computing the IM is within the time constraints of the original algorithm.