Social Network Analysis Homework 3

Team: 1

Student 1: r03922096 洪立遠

Student 2: b00902057 陳煥元

Student 3: b00902104 楊勖

How to run your program

|  |
| --- |
| Just type make to run our python code  我們有用到pickle、sys、http.client、operator、numpy、scipy等librarys |

Experiments for training data

我們只有紀錄最後兩次的public，edge沒有記錄下來；Forest Fire因為結果不佳沒有記錄完。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Dataset** | **Degree distribution KL** | **Closeness centrality** | **Edge attribute 1** |
| Hybrid | Public data  (node:100) | 0.2046 | 590.110000 |  |
| Hybrid | Public data  (node:300) | 0.2064 | 379.900000 |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Dataset** | **Node attribute 1** | **Node attribute 2** | **Node attribute 3** | **Node attribute 4** | **Node attribute 5** |
| Hybrid | Public data (node:100) | 0.0002416840553 | 0.07622502658 | 0.06744258391 | 0.01478105399 | 0.0376256893 |
| Hybrid | Public data (node:300) | 0.004757778633 | 0.04924035683 | 0.07505610773 | 0.007756301002 | 0.218599239 |

Model description

|  |  |
| --- | --- |
| **Model** | **Description** |
| Forest Fire | 1. Forest Fire就是先選擇一個點當作種子，再利用 Geometric Distribution 產生一個變數 W， Geometric Distribution 的 mean 為 1-pfpf，而 Pf 就是這個function 所需要輸入的參數。 2. 產生 W 以後再從 Seed 點中選 W 個 neighbors 去 visit，被 visit 的點對他再產生一個 W 去選擇，一直重複下去直到找不到點去選為止。 |
| Hybrid | 1. 前一半的點選擇max{(node degree – node graph degree) \* node attribute △KL Divergence} 2. 後一半的點選擇max{(node degree – node graph degree)} |

Others (optional)

|  |
| --- |
| 一、 Query model  我們首先嘗試運用現有的採樣模型，在閱讀了Jure Leskovec等人所著 Sampling in large graph [1] 這篇文章後，決定利用 Forest fire model。  Forest Fire就是先選擇一個點當作種子，再利用 Geometric Distribution 產生一個變數 W， Geometric Distribution 的 mean 為 1-pfpf，而 Pf 就是這個function 所需要輸入的參數。  產生 W 以後再從 Seed 點中選 W 個 neighbors 去 visit，被 visit 的點對他再產生一個 W 去選擇，一直重複下去直到找不到點去選為止。  論文建議參數P在應在0到0.4之間去做會有比較好的效果，應此我們採取 pf = 0.2 去做，但是他的 Degree distribution 之 KL Divergence 落於 0.8432。  另外有考慮由於 Numpy 所產生的 Geometric Distribution 之 mean 為 1pf所以得方程式1pf=1pf1-pf=10.2得到 Pf 為 112下去測試後其 KL Divergence並沒有下降太多，仍然在 0.73 多  最後也有考慮過把第一次 uniform Select 的點 改成實際 Degree 最高的點下去跑，也沒有看到進步的效果。  評估 Forest Fire Model 效果不佳原因為，Query 得到的點多為Degree 較多的點，所以需要想辦法 Scale Off。而參考的文章[1] 也有提到， Sampling Algorithm 在 Sample 超過 原圖 25% 的點時，才會有較佳的效果。  因為Forest fire model的效果不佳，所以轉念一想，乾脆每次Query都Query degree最大的node，這樣可以得到最多的information，就會更加接近原圖的distribution。  不過單純從nod的degree最大去選，可能會有誤判的情況是，其實query node的一大部分neighbor已經在自己的sub graph中了，所以有效的node information其實沒有degree那麼多了。所以改良後，我們選擇(graph.node[n][‘degree’]-graph.degree(n))最大的點，作為query node，而整體的performance，也確實變得較好。  再來，我們考慮全域與區域的差別。全域即指每一次query都去找自己的graph中，node information會最大的點來query，而區域則是每一次的query candidate只限定在上一次query node的neighbor中。  我們發現使用區域的方式，效果較好，因為可能其物理意義是，先把這一個區域的graph作更深的了解，然後再walk到下一個點，而不像是全域中的做法，每次都在這個graph中跳來跳去。  完成了degree部分，也就是選區域中最大information的node來query，我們打算來把node attribute distribution的部分加入演算法中。  我們用了△KL Divergence作為measure的依據，也就是要選擇 node attribute △KL Divergence最大的點來query，為了要有2個distibution可以做KL Divergence，我們把那個node拿掉，與不拿掉來做為比較。這樣的物理意義是，選擇graph中最奇怪的那個人，作為query的對象。  經public graph的測試，發現node attribute range愈小的attribute，△KL Divergence的效果愈差，也就是說單純最大information去query，KL Divergence也就會很小了。依據此結論，我們只有考慮了node attribute 0與attribute 2這兩個node attribute的△KL Divergence。  最終，與最大information這樣的概念融合，我們選擇(graph.node[n][‘degree’]-graph.degree(n))\*△KLDivergencey最大的點來query，其中△KLDivergencey只有考慮node attribute 0 與node attribute 2。  二、Output strategy   1. degree distribution   考慮到由於我們在選點時都是挑Degree 大的點所選擇，決定將我們產生好的 list直接reverse，把第一項跟最後一 Swap, 第二項, 跟倒數第二項 Swap…. 依此類堆。這樣做的目的是考量到我們每次都選到 Degree 高的點，所以Degree 低的點反而比較少。  這樣做的好處是比較簡單，容易實作。壞處則是當 Query 次數提升時，KL Divergence 並沒有下降很多。  2. top 100 Closeness  作法本來是使用Sample好的圖經過 Networkx本身的函示去計算得到前 10最大的點當作答案，後來發現 Closeness 的公式為oao2.JPG。根據論文[2]，Degree較大的點，分母會比較小，比較容易成為Closeness大的點，所以就將 Degree Sort後輸出當作答案。效果不錯，在public Graph 中，ATR 從 6869 降到 590.11，且 Query 越多次效果越好。  3. node attribute distribution  由於sample出的點並不能涵蓋到所有的attribute，我們使用add-1 smoothing來估計沒觀察到的attribute之出現機率。  三、Reference  [1] Jure Leskove,and Christos Faloutsos, “Sampling in large graph”, Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, Pages 631-636  [2] Kazuya Okamoto, Wei Chen, and Xiang-Yang Li, "Ranking of Closeness Centrality for Large-Scale Social Networks", FAW '08 Proceedings of the 2nd annual international workshop on Frontiers in Algorithmics |