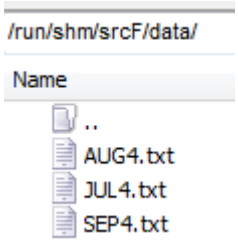
第1章 原始数据

共 3 个文件，如下。

/data/*4.txt



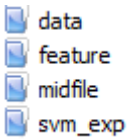
原始的网络数据格式为：

用户编号 用户朋友数目 用户属性数目 朋友列表 属性列表

3 1 4 2955 -16 -17 -18 -19

实例：如上一行数据表示，3 是用户 ID, 3 有 1 个朋友（id 是 2955），3 有 4 个属性（属性 id 分别是-16 -17 -18 -19）。

第2章 源码的框架：



模块先后依赖关系：
1.midFile
2.Feature
3.svm_exp
各模块的运行方法。
1. midfile/
 nohup ./midfile.sh > midfile.txt 2>&1 &
2.feature/
 nohup ./feature.sh > feature.txt 2>&1 &
3.svm_exp/
 nohup ./svmexp.sh > svmexp.txt 2>&1 &

“> midfile.txt 2>&1 &”意思是，标准输出核标准错误输出重定向到文件 midfile.txt 中。&是在后台运行的意思

第3章 midFile

此模块，产生一些中间文件。运行命令举例：
nohup ./midfile.sh > midfile.txt 2>&1 &

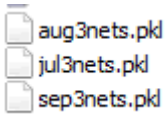
midfile.sh 内容，模块内的先后依赖关系：
nohup python main_net_pickle.py > net_pickle.txt 2>&1 &
wait
nohup python main_p_utarget.py > p_utarget.txt 2>&1 &
第一个执行完毕，第二个才能开始。

3.1 main_net_pickle.py

3.1.1 运行说明：

执行 main_net_pickle.py 代码。
运行时间约为 10 秒左右。

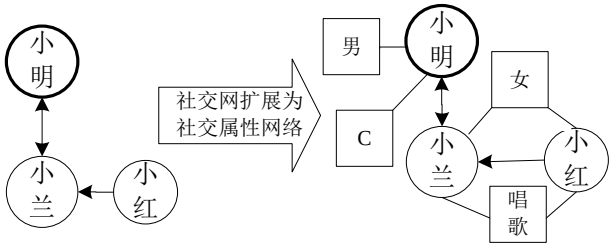
处理结果
读入 JUL4.txt,AUG4.txt,SEP4.txt 等 3 个原始文件，
读入网络快照原始数据 JUL4.txt/AUG4.txt/SEP4.txt，得到 jul3nets.pkl/aug3nets.pkl/sep3nets.pkl 文件。一个 pkl 文件存储一个网络快照的 3 个 nx.Graph 对象，分别对应着社交图，属性图，社交属性图。
最后删除 3 原始文件。
运行后得到如下 3 个文件，分别存储网络的 7,8,9 月份的网络快照。



3.1.2 函数 getsnet

函数功能：原始数据解析。读入某一个网络快照 JUL4.txt/AUG4.txt/SEP4.txt，得到 `snet`,`anet`,`sanet`，3 个 nx.Graph 对象，分别对应着社交图，属性图，社交属性图。
参数：文件名，如`'../data/JUL4.txt'`
输出：`snet`,`anet`,`sanet`。分别是社交网络，下图左；属性网络（下图右中减去下图左

中的边）；社交属性网络，下图右。



3.2 main_p_utarget.py

3.2.1 saveP_U_Utarget

函数功能：生成节点对样本和标签。读入某三个网络快照jul3nets.pkl/aug3nets.pkl/sep3nets.pkl，得到7-8月的节点对样本和标签和7.8.9月的节点对样本和标签，并分别存储到两个文件中。

无参数：

输入：3个文件（jul3nets.pkl，aug3nets.pkl，sep3nets.pkl）

输出：两个文件

```
ja78='../data/ja78P_U_Utarget.pkl'
jas78_9='../data/ja78_9P_U_Utarget.pkl'
。
```

ja78P_U_Utarget.pkl 存储社交网络网全图边对应的节点对集合和标签。

如果节点对（u,v）在7月份存在链接，标签为1，分到P集合中，即正样本集合。

如果节点对（u,v）在7月份不存在链接分到U集合中（即未标记样本集合），如果在8月份，建立链接，标签为1；反之为0.

一个样本形如[u,v,target]，u，v是节点，target是标签。

*'../data/ja78_9P_U_Utarget.pkl'*存储社交网络网全图边对应的节点对集合和标签。

如果节点对（u,v）在8月份存在链接，标签为1，分到P集合中。

如果节点对（u,v）在8月份不存在链接分到U集合中，如果在9月份，建立链接，标签为1；反之为0.

一个样本形如[u,v,target]，u，v是节点，target是标签。

第4章 Feature

此模块为特征分析。主要包括局部特征共同邻居、**Adamic-Adar 系数**（AA）、**Jaccard 系数**（JC）、**余弦相似度**（COS）的计算，并进行无监督实验；全局特征重启动随机游走的计算和无监督实验；时间特征的计算和和无监督实验。

模块运行命令为：

nohup ./feature.sh > feature.txt 2>&1 &

feature.sh 的内容和模块内部的先后依赖关系为：

```
nohup python main_cnsnwight_roc_cmd.py 1 sep aa > log/aasep.txt 2>&1 &
nohup python main_cnsnwight_roc_cmd.py 1 sep cn > log/aasep.txt 2>&1 &
nohup python main_timesanwight_roc_cmd.py  > log/timelog.txt 2>&1 &
nohup python main_randomwalk_sanwight_roc_cmd.py 0.93 10 > log/randomjulaug0.93_20.txt 2>&1 &
wait
python sanrwr_a.py &
python snrwr_a.py &
```

sanrwr_a.py 和 snrwr_a.py 要在 randomwalk 运行完毕才能运行。

4.1 WGraph.py

4.1.1 说明：

定义了类 WGraph。无运行。类中，主要包含了无权图，节点权，顶点权，及局部相似度共同邻居、**Adamic-Adar 系数**（AA）、**Jaccard 系数**（JC）、**余弦相似度**（COS）等计算。

类成员：

self.g 为无权图，为一个 nx.Graph，无向图对象。

`self.wnodes` 存储 self.g 的顶点权，为词典对象（dict）。

`self.wedges` 存储 self.g 的边权，为词典对象（dict）。

成员函数：

set_nodes_weight 计算顶点权 *self*.wnodes

set_edges_weight 计算边权 self.wedges

get_cn_len(*self*,u,v)计算节点对 u,v 的共同邻居的数量，返回值为共同邻居的

数量+1。（加 1，是为了避免结果为 0，因为有些计算要用节点对 u,v 的共同邻居的数量做分母，为避免除 0，所以加 1）

get_node_weight(*self*,u):返回节点 u 的顶点权重

get_edge_weight(*self*,u,v):返回边（u，v)的权重

get_cn_weight(*self*,u,t):计算节点对的共同邻居特征，

返回 4 个值，cnt,cntw,cntw2,cnte，意义分别为。

1.无权的共同邻居特征

2.顶点权的

$$CN\_VW(u,t)=\sum_{t'\in F(u)\cap F(t)} w(t')$$

3.顶点权的

$$CN\_VW(u,t)=\sum_{t'\in F(u)\cap F(t)} 1./w(t')$$

4，边权的

$$CN\_EW(u,t)=\sum_{t'\in F(u)\cap F(t)} (w(u,t')+w(t,t'))$$

get_aa_weight(*self*,u,v):和 get_cn_weight 类似，计算的是 AA 特征。也是返回 4 个值。

1.无权的 AA

2.顶点权的 AA

$$AA\_VW(u,t)=\sum_{t'\in F(u)\cap F(t)} \frac{w(t')}{\log(1+\sum_{z\in F(t')} w(z))}$$

3.顶点权的 AA

$$AA\_VW(u,t)=\sum_{t'\in F(u)\cap F(t)} \frac{1./w(t')}{\log(1+\sum_{z\in F(t')} 1./w(z))}$$

3，边权的 AA

$$CN\_EW(u,t)=\sum_{t'\in F(u)\cap F(t)} (w(u,t')+w(t,t'))$$

其他 cos , jc 等类似，不在列出。

4.2 main_cnsnwight_roc_cmd.py

运行说明：

此为局部特征无监督实验。仅使用一个特征（如 CN,JC,AA,COS）作为分类依据，。

命令行参数：

python main_cnsnwight_roc_cmd.py 1 jul aa

当第一个参数为 1 时，调用函数 **linktest1file**，**无监督**数据集为无向图的完全图的边对应的节点对集合。

第二个参数，指定了使用的网络月份，jul aug sep 分别指定使用 7\8\9 月，（jul3nets.pkl/aug3nets.pkl/sep3nets.pkl）。

第三个参数，指定了无监督实验使用的特征类型，AA（CN/ JC/ COS）就使用 AA。

AUC 等 log 结果信息存储在文件，filename='log/'+sys.argv[3]+sys.argv[2]，即 aajul 文件中。

python main_cnsnwight_roc_cmd.py 2 jul aug cn

当第一个参数为 2 时，调用函数 **linktest2file**。

第二个参数，指定了使用的旧月份，jul 就指定使用 7 月。

第三个参数，指定了使用的新月份，aug 就指定使用 8 月。

第四个参数，指定了无监督实验使用的特征类型，cn 就使用 CN

AUC 等 log 结果信息存储在文件，

filename='log/'+sys.argv[4]+sys.argv[2]+sys.argv[3]，即 cnjulaug 文件中。

4.2.1 linktest1file

函数功能：

将社交网络的完全图的边的节点对集合作为线性阈值分类器，无监督实验的数据样本。

其中在 JUL 中，存在链接的为正样本，其他为负样本。

使用 fff 计算的特征作为线性阈值分类依据，选取多个个阈值，对每一个阈值的分类结果，计算（tpr,fpr）,然后画 ROC（横纵坐标分别为 tpr,fpr），计算 AUC.

代码中也计算了准确率、召回率、F1 值等，（但是对不同的阈值，结果波动

也大，故不作为链接预测的评估标准）

输入参数

```
linktest1file(pklfile='../data/jul3nets.pkl',fff=WGraph.get_cn_weight):
```

Pklfile 指定了网络的快照，jul3nets.pkl 指定了 7 月份的网络快照，作为无监督实验的数据源。

Fff 指定了特征的计算方法，可以是 CN,AA,COS,JC 等等

输出：

代码178行179行的log信息给出了无监督实验的结果，**无监督**数据集为无向图的完全图的边对应的节点对集合。

```
logging.debug('cnroc_auc=%0.8f,cn_node_weight=%0.8f,cn_node_weight2=%0.8f,cn_edge_weight=%0.8f,'%(cnroc_auc,cn_node_weight,cn_node_weight2,cn_edge_weight))
```

此为社交网络上无权、WV1，WV2，WE等4中权重方法对应的AUC结果。

```
logging.debug('cnroc_auc=%0.8f,cn_node_weight=%0.8f,cn_node_weight2=%0.8f,cn_edge_weight=%0.8f,'%(sancnroc_auc,sancn_node_weight,sancn_node_weight2,sancn_edge_weight))
```

此为社交属性网络上无权、WV1，WV2，WE等4种权重方法对应的AUC结果。

4.2.2 Linktest2file

函数功能和输入参数意义同 4.2.1 节。

不同的是，正负样本的定义，和无监督实验数据集。

pkloldf 为网络过去的快照

pklnewf 为网络现在的快照

在 pkloldf 时刻的边集合为 E,完全图的边集合 A,无监督实验数据集为节点对结合 A-E

如果 A-E 中的节点对，在 pklnewf 中建立链接，则为正样本（即新增链接），其他为负样本（没有建立链接）.

```
linktest2file(pkloldf='../data/jul3nets.pkl',pklnewf='../data/aug3nets.pkl',fff=WGraph.get_cn_weight):
```

输出：

代码332行333行的log信息给出了无监督实验的结果，

*'../data/ja78P_U_Utarget.pkl'*文件或者*'../data/ja789P_U_Utarget.pkl'*中U集合。

logging.debug(*'cnroc_auc=%0.8f,cn_node_weight=%0.8f,cn_node_weight2=%0.8f,cn_edge_weight=%0.8f,'*%(cnroc_auc,cn_node_weight,cn_node_weight2,cn_edge_weight)) 此为社交网络上无权、WV1，WV2，WE等4中权重方法对应的AUC结果。

logging.debug(*'cnroc_auc=%0.8f,cn_node_weight=%0.8f,cn_node_weight2=%0.8f,cn_edge_weight=%0.8f,'*%(sancnroc_auc,sancn_node_weight,sancn_node_weight2,sancn_edge_weight))

此为社交属性网络上无权、WV1，WV2，WE等4中权重方法对应的AUC结果。

4.2.3 Getbestthreshold

功能是，得到一个（几个）候选最佳阈值，并以列表的形式返回。

输入参数

fpr, tpr, thresholds 来自 roc_curve 的返回值

返回：候选最佳阈值的列表（list）

4.3 main_randomwalk_sanwight_roc_cmd.py

4.3.1 功能说明：

此为计算重启动随机游走特征存储到双嵌套的字典对象中，最后以文件形式存储到硬盘。

命令行参数：(alpha=a,max_step=m)

a=float(sys.argv[1])，第一个参数为启动概率 *a*

m=int(sys.argv[2])，第二个参数为迭代次数。

执行命令形如：

```
nohup python main_randomwalk_sanwight_roc_cmd.py 0.93 10 > log/randomjulaug0.93_10.txt 2>&1 &
```

4.3.2 PersonalRank

(G,alpha,root,max_step):

函数功能

此为无权图上的重启动随机游走，计算游走者从节点 root 出发，访问其他节点的

概率，存储到 dict 对象中。

输入参数：

G 为 nx.Graph 对象，

Alpha 为启动概率

Root 为游走者的出发节点，

Max_step 为游走步数（迭代更新次数）

返回：

dict 对象，存储 root 到其他节点的访问概率

4.3.3 PersonalRankw

(G,alpha,root,max_step):

函数功能：

此为边权图上的重启动随机游走，计算游走者从节点 root 出发，访问其他节点的概率，存储到 dict 对象中。

输入参数：

G 为 WGraph 对象，其他参数同 **PersonalRank。**

返回：

dict 对象，存储 root 到其他节点的访问概率

4.3.4 get_sn_wsn_rwr

函数功能：

计算并存储两个双嵌套的 dict 对象（[rwr_sn,rwr_wsn]）分别为无权的社交网络上随机游走特征，边权的社交网络上的随机游走特征，随机游走特征计算见下式。

（$RWR\_WSAN(v_u,v)=(P(v_u,v)+P(v,v_u))/2$）

$P(v_u,v)$——指游走者从出发节点 v_u 出发，访问 v 的概率；

输入参数：

同 4.3.2.

(pkloldf = *'../data/jul3nets.pkl'*,alpha=0.5,max_step=20):

Pkloldf 参数为网络快照，存储 3 个网络，社交网络，属性网络，社交属性网络。

输出

为文件（文件名：pkloldf+str(alpha)+str(max_step)+*'snrwr'*），文件内存储了两个双嵌套的 dict 对象（[rwr_sn,rwr_wsn]），存储了全部节点对的重启动随机游走特征。

rwr_sn 存储的是社交网络上的 rwr 双嵌套的 dict 对象

rwr_wsn 存储的是边加权社交网络上的 rwr 双嵌套的 dict 对象。

4.3.5 get_san_wsan_rwr

与 get_sn_wsn_rwr 功能，输入输出与 4.3.4 几乎相同。
类似，不同是，在社交属性网络和边权社交属性网络上随机游走特征存储。

文件很大，输出文件名为
社交属性网络的文件为 pkloldf+str(alpha)+str(max_step)+'sanrwr'
每个文件内，存的两个双嵌套的 dict 对象（[rwr_san,rwr_wsan]）分别为无权的随机游走特征，边权的随机游走特征。
rwr_sn 和 rwr_wsn 都是双嵌套的字典对象,类似一个二位数组。
rwr_san 存储的是社交属性网络上的 rwr 双嵌套的 dict 对象
rwr_wsan 存储的是边加权社交属性网络上的 rwr 双嵌套的 dict 对象

4.4 sanrwr_a.py 和 snrwr_a.py

这两个文件类似。不同处，是一个是社交属性网络的 RWR 有权和无权的 AUC 结果；另一个是社交网络的 RWR 有权和无权的 AUC 结果。

4.4.1 sanrwr_a.py

linktest78file()
函数功能：
读入 4.3 中产生的社交属性网络的文件，计算无监督 AUC，并画出 ROC 曲线，数据集正负样本与 4.2.2 节相同。
输入文件：

```
alphas = [0.93]
for a in alphas:
'../data/jul3nets.pkl'+str(a)+'10sanrwr'
```

输出：
日志：filename='log/sanrwr_alpha_20.txt'
63 行代码
logging.debug('rwr auc = %0.8f'%rwroc_auc)显示的是无权社交属性网络的无监督 AUC.
67 行代码：
logging.debug('rwr wsn auc = %0.8f'%rwroc_auc) 显示的是边权社交属性网络的无监督 AUC.
运行方法为 python sanrwr_a.py &

4.4.2 snrwr_a.py

linktest78file()

函数功能：

读入 4.3 中产生的社交属性网络的文件，计算无监督 AUC，并画出 ROC 曲线，数据集正负样本与 4.2.2 节相同。

输入文件：

```
alphas = [0.93]
for a in alphas:
'../data/jul3nets.pkl'+str(a)+'10snrwr'
```

输出：

日志：filename='log/snrwr_alpha_20.txt'

63 行代码

logging.debug('rwr auc = %0.8f'%rwroc_auc)显示的是无权社交网络的无监督 AUC.

82 行代码：

logging.debug('rwr wsn auc = %0.8f'%rwroc_auc) 显示的是边权社 7 网络的无监督 AUC.

运行方法为 python snrwr_a.py &

4.5 main_timesanwight_roc_cmd.py

4.5.1 gettimegeneratorlen

函数功能：

计算下式的时间特征。

输入参数

(u,v,net,sanet,newnet,newsanet)

net,sanet,是 t’快照。newnet,newsanet 是 t 快照。

输出是 TIME(u,v)。

式中 $d_t(u)$——表示节点 u 在时间 t 的度数，$t>t'$。

$$TIME(u,v)=d_t(u)-d_{t'}(u)+d_t(v)-d_{t'}(v)$$

输出：

TIME(u,v)

4.5.2 linktest3file

功能：

计算世界特征，并进行无监督实验。数据集为*'../data/ja78_9P_U_Utarget.pkl'* 中的 U 集合。在 8 月 时刻的边集合为 E,完全图的边集合 A,无监督实验数据集为节点对结合 A-E

如果 A-E 中的节点对，在 9 月 中建立链接，则为正样本（即新增链接），其他为负样本（没有建立链接）.

输入参数：

(pkloldf = '../data/jul3nets.pkl',
pklnewf='../data/aug3nets.pkl',
pkltest='../data/sep3nets.pkl'):

输出：

log/timef.log 存储仅使用时间特征得到的无监督 AUC.

代码 122 行

```
logging.debug('timefroc_auc=%0.8f,timefsanroc_auc=%0.8f'%(timefroc_auc,
timefsanroc_auc))
```

第一个数是社交网络的时间特征的 AUC,第二个数是社交属性网络的时间特征的 AUC.

第5章 Svm_exp

本模块主要是做样本过滤和 SVM 实验。

运行；

nohup ./svmexp.sh > svmexp.txt 2>&1 &

svmexp.sh 文件内 wait 的命令标明了模块内部的先后依赖关系。

5.1 main_p_n_test.py

5.1.1 save78train_test

功能：

此程序用来 7-8 月的网络快照，产生 SVM 实验的过滤后的样本，包括 P,N,TEST（P 为正样本集合，N 为负样本集合，P 和 N 是训练集。TEST 是测试集）。

输入文件为：

ja78P_U_Utarget.pkl

jul3nets.pkl

输出文件为：

pntest78.pkl pntest89.pkl

存储样本每个样本形如[u,v,target]表示节点对 u,v 及其标签。

P 表示正样本集合

N 表示欠抽样后的负样本集合。P N 为训练集合。

Test 为测试集合。

5.1.2 save789train_test

功能：

此程序用来 7、8-9 月的网络快照产生 SVM 实验的过滤后样本，包括 P,N,TEST（P 为正样本集合，N 为负样本集合，P 和 N 是训练集。TEST 是测试集）。

输入文件为：

ja789P_U_Utarget.pkl

jul3nets.pkl aug3nets.pkl

输出文件为：

pntest89.pkl

存储样本每个样本形如[u,v,target]表示节点对 u,v 及其标签。

P 表示正样本集合

N 表示欠抽样后的负样本集合。P N 为训练集合。

Test 为测试集合。

5.2 main_san_time.py

功能：

此模块是用来做动态权重社交属性网和权重社交属性网的对比实验的，使用的快照是。首先提取训练集和测试集的特征向量。然后做 SVM 实验计算 AUC。粗细粒度的搜索最佳的 gamma 值。

命令行参数

```
st=int(sys.argv[1])
en=int(sys.argv[2])
num = int(sys.argv[3])
```

这三个参数，主要是用来产生 gamma 的，粗细粒度的搜索最佳的 gamma 值。

```
gammas  = np.linspace(start=st, stop=en, num=num, endpoint=False)
```

如：

```
>>> gammas  = np.linspace(start=1, stop=11, num=10, endpoint=False)
>>> gammas
array([  1.,   2.,   3.,   4.,   5.,   6.,   7.,   8.,   9.,  10.])
```

输入：

jas78='../data/pntest89.pkl'由5.1产生。

timefile = *'../data/aug3nets.pkltimef'*4.5 产生，时间特征

pklfile='../data/jul3nets.pkl0.9310sanrwr' 4.3 产生。随机游走特征

输出：

train_test=*'../data/789train_test_san.pkl'*存储训练集合和测试集合的特征向量和标签。

日志文件：

filename=*'log/san_789_nogrid_tiny_gamma%d-%d_%d_time.log'*%(st,en,num),

日志文件中有相应 AUC 结果，日志文件中已经标明。

5.2.1 getgeneratorlen(g):

计算 generator 对象 g 的个数并返回。

5.3 main_san.py 和 main_sn.py

和 5.2 类似。命令行参数和功能都是一样的。

5.3.1 main_san.py

功能：

用来做社交属性网络和权重社交属性网络的对比实验。

输入：

jas78='../data/pntest78.pkl'由5.1产生

pklfile='../data/jul3nets.pkl0.9310sanrwr' 4.3 产生。随机游走特征

输出：

train_test='../data/78train_test_san.pkl'存储训练集合和测试集合的特征向量和标签。

日志文件：

filename='log/san_78_nogrid_tiny_gamma%d-%d_%d_san.log'%(st,en,num)

，

日志文件中有相应 AUC 结果，日志文件中已经标明。

5.3.2 main_sn.py

功能：

用来做社交网络和权重社交网络的对比实验。

输入：

jas78='../data/pntest78.pkl'由5.1产生

pklfile='../data/jul3nets.pkl0.9310snrwr' 4.3 产生。随机游走特征

输出：

train_test='../data/78train_test_sn.pkl'存储训练集合和测试集合的特征向量和标签。

日志文件：

filename='log/san_78_nogrid_tiny_gamma%d-%d_%d_sn.log'%(st,en,num)

日志文件中有相应 AUC 结果，日志文件中已经标明。