实验二报告: Complex Network Analysis

0.项目文件结构

项目文件结构如图1所示:

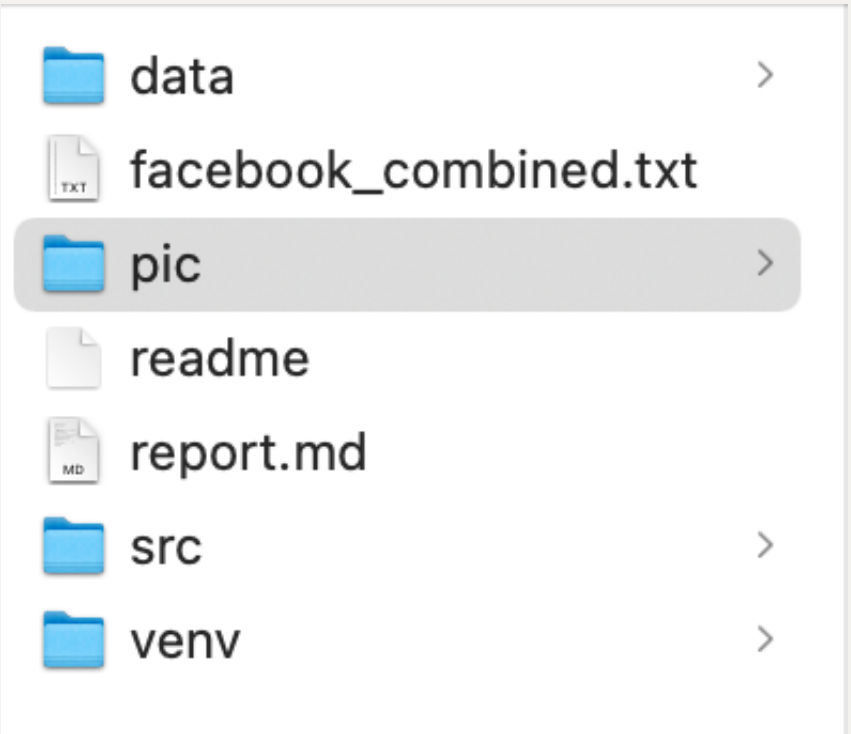

图1.项目文件结构

其中`data`是数据文件，`pic`存放项目截图，`src`存放源码文件，`venv`是用pycharm配置的虚拟环境。

源码的文件结构如图2所示

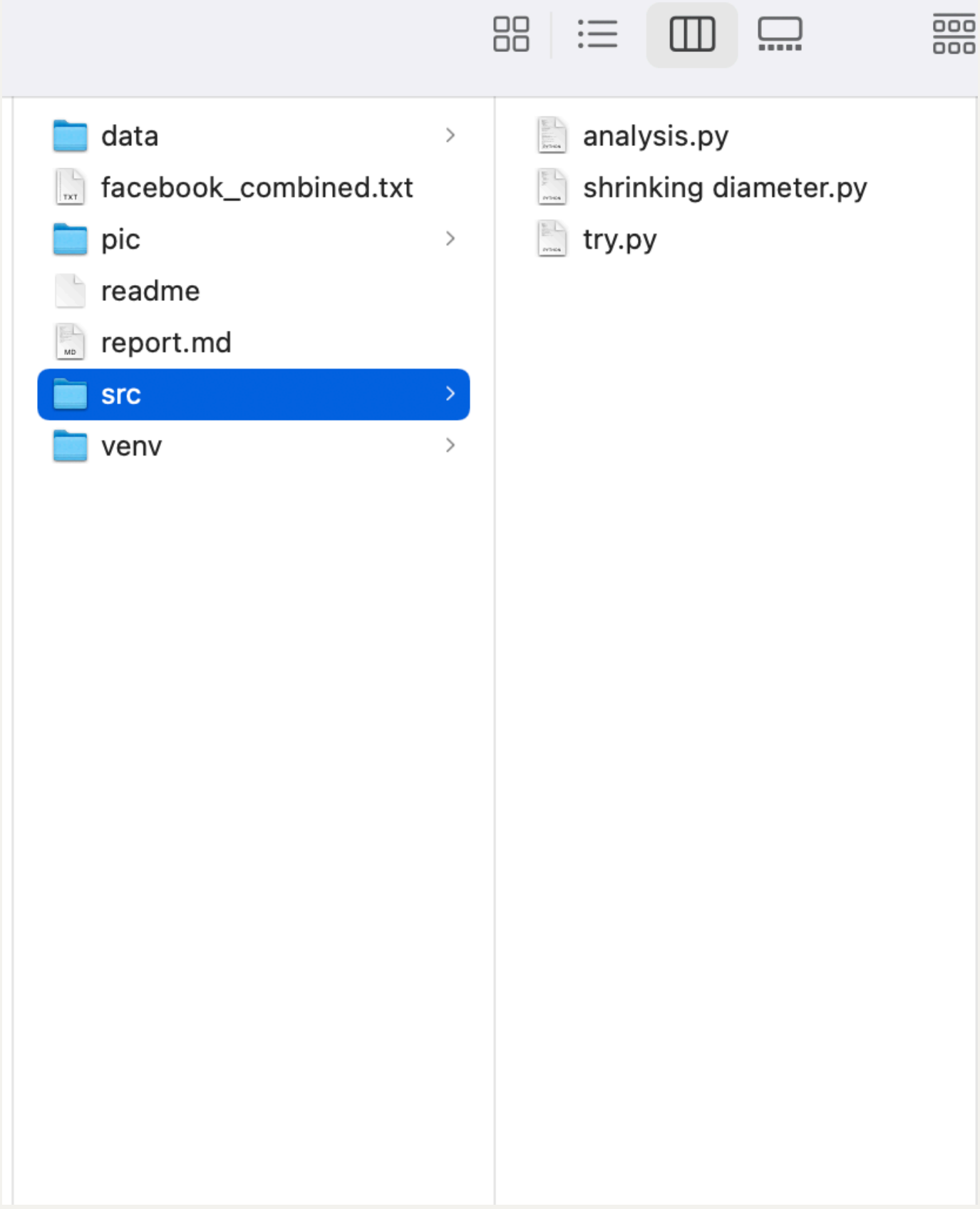

图2.src结构图

`analysis. py`是本次实验的主要内容，`shrinking diameter.py`文件是直径收缩部分的源代码，`try.py`是绘制network community时的第一次尝试。

1. 准备工作

首先在Pycharm中下载networkx和community的库。接着在Standford网站中下载数据集。

> *数据集链接：https://snap.stanford.edu/data/ego-Facebook.html*

在lab2文件夹和data文件夹内部各放置一份数据，如图3所示：

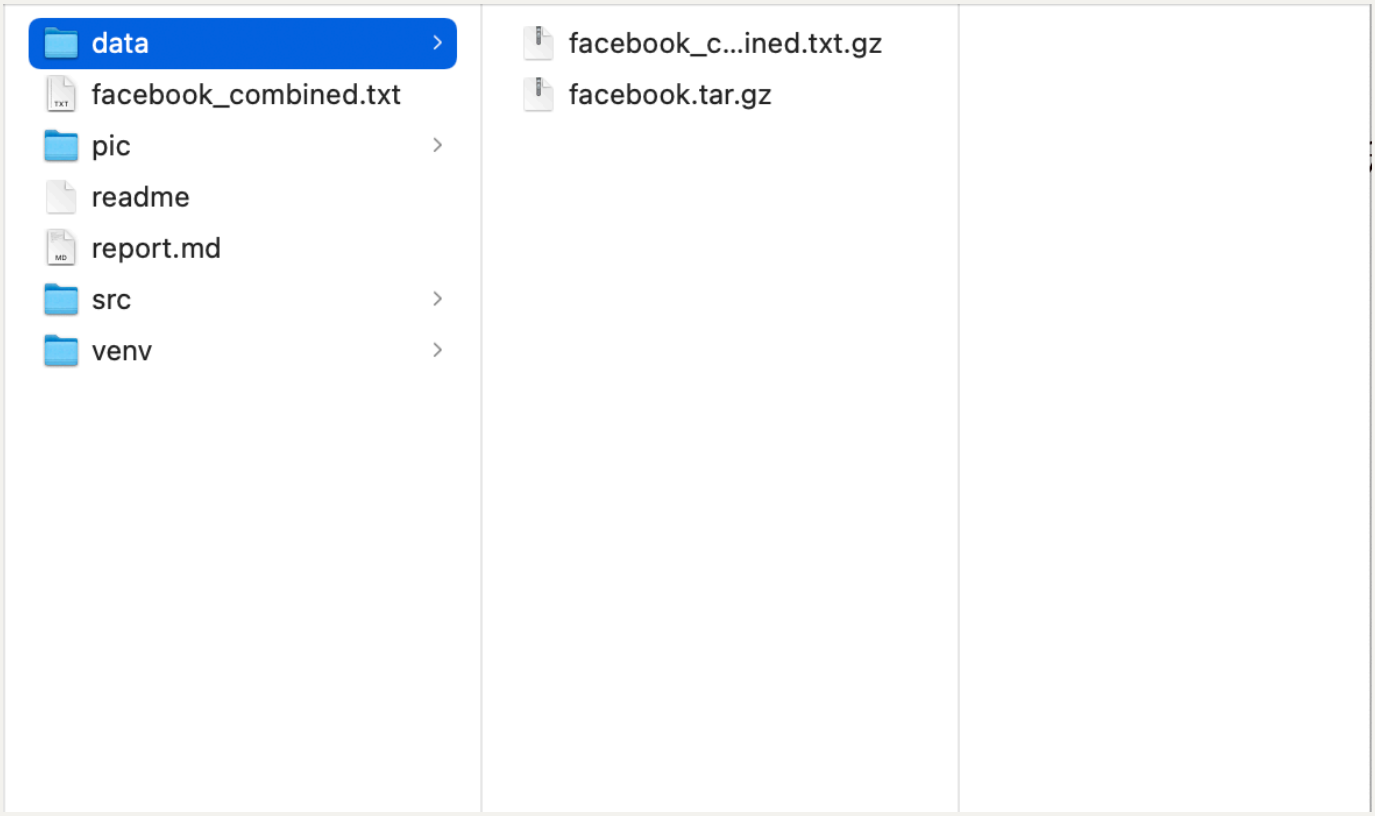

图3.准备工作

2. 对网络进行分析

2.1. 准备工作

准备工作的源码如下：

```
import community.community_louvain
import networkx as nx
import pandas as pd
import numpy as np
```

```
import pylab
import matplotlib.pyplot as plt
from networkx.algorithms.community import k_clique_communities
from community import community_louvain
from random import randint

# 从数据文件夹中加载数据
fb = pd.read_csv(
    "data/facebook_combined.txt.gz",
    compression="gzip",
    sep=" ",
    names=["start_node", "end_node"],
)

G = nx.from_pandas_edgelist(fb, "start_node", "end_node")
```

主要功能是import相应的库，并且解压并读入data文件，形成图结构。

2.2. Visualize the whole graph connectivity

代码部分如下：

```
plot_options = {"node_size": 10, "with_labels": False, "width": 0.15}

# 使用spring_layout函数 考虑节点和边来计算节点位置
pos = nx.spring_layout(G, iterations=15, seed=1721)
fig, ax = plt.subplots(figsize=(15, 9))
ax.axis("off")
nx.draw_networkx(G, pos=pos, ax=ax, **plot_options)
pylab.show()
```

使用`spring_layout`函数，进行绘图并输出。输出的结果如图4所示。

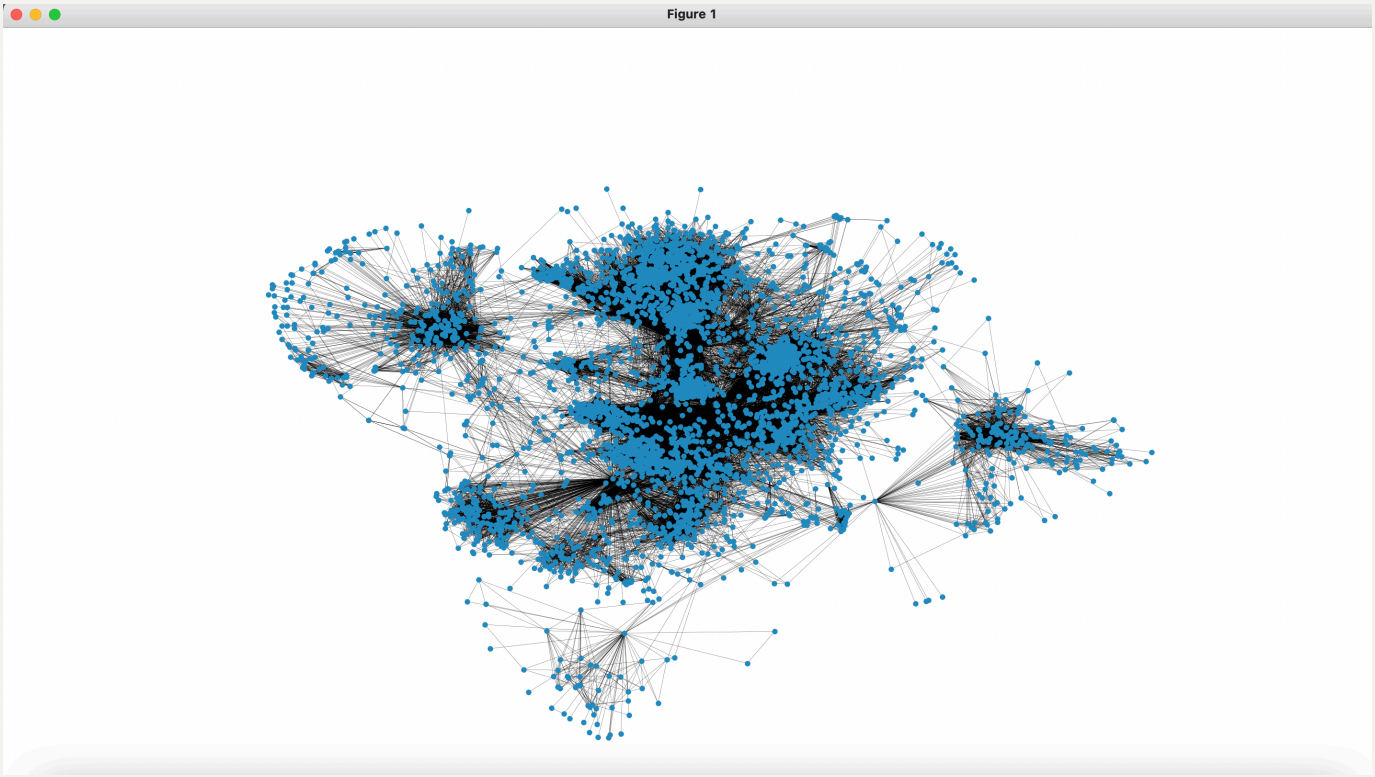

图4.网络图像

2.3. Average path length

```
print("Average path length:")
print(nx.average_shortest_path_length(G))  # average_path_length
```

平均路径长度只需要使用networkx即可得到。

2.4. Clustering Coefficient

代码部分如下:

```
plt.figure(figsize=(15, 8))
plt.hist(nx.clustering(G).values(), bins=50)
plt.title("Clustering Coefficient Histogram ", fontdict={"size": 35},
loc="center")
plt.xlabel("Clustering Coefficient", fontdict={"size": 20})
plt.ylabel("Counts", fontdict={"size": 20})
pylab.show()
```

由此可以得到Clustering Coefficient的图像，如图5所示：

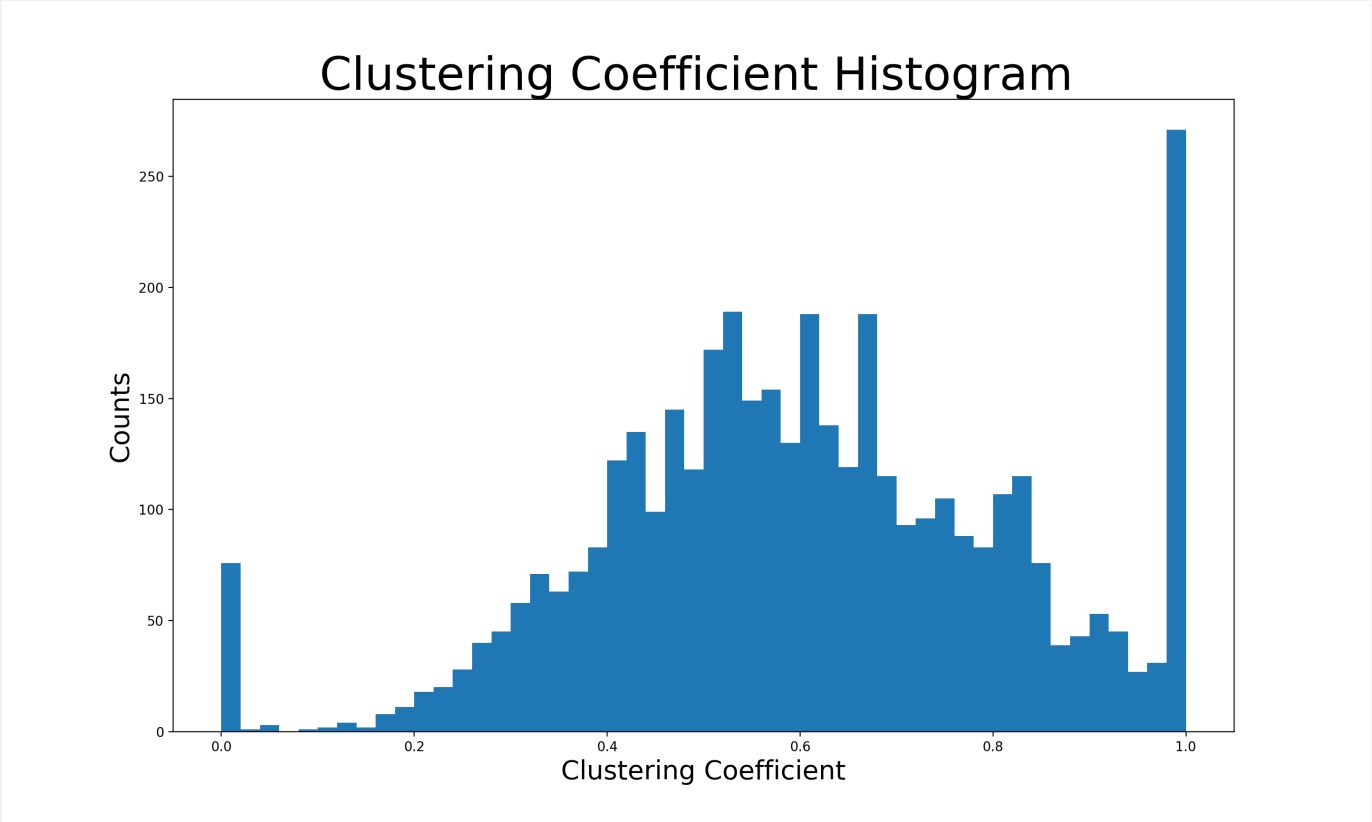

图5.Clustering Coefficient

2.5. Degree distribution

首先计算平均度：

```
# 平均度
d = dict(nx.degree(G))
print(d)
print("平均度为：", sum(d.values()) / len(G.nodes))
```

打印度分布：

```
# 度分布
print(nx.degree_histogram(G))  # 返回所有位于区间[0, dmax]的度值的频数列表
```

绘制度分布图：

```
x = list(range(max(d.values()) + 1))
# y = [i/len(G.nodes) for i in nx.degree_histogram(G)]
y = [i / sum(nx.degree_histogram(G)) for i in nx.degree_histogram(G)]
print(x)
print(y)

plt.bar(x, y, width=0.5, color="blue")
plt.xlabel("$k$")
plt.ylabel("$p_k$")
plt.xlim([0, 350])
pylab.show()
```

结果如图6所示

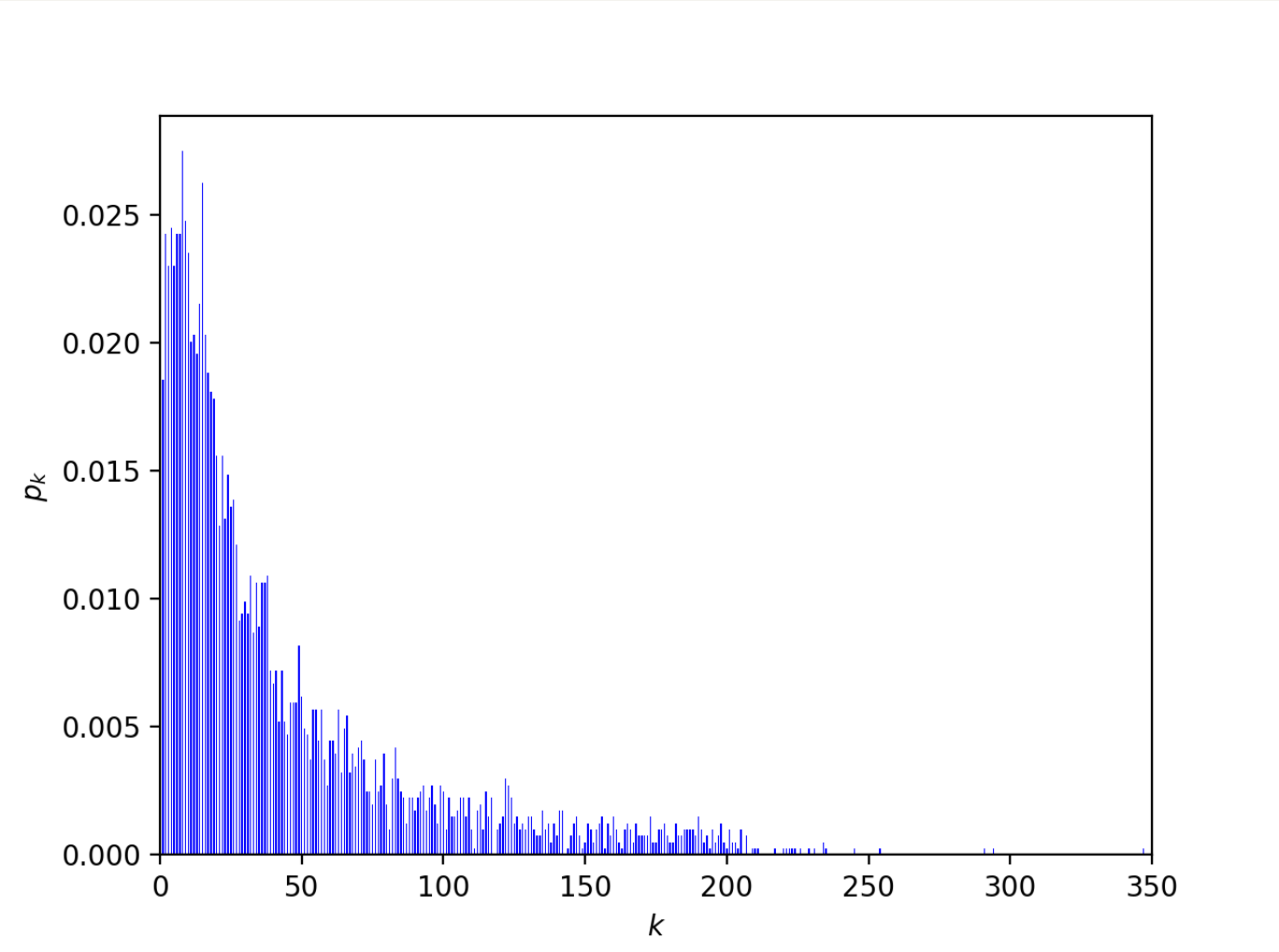

图6.Degree distribution

2.6. Assortativity

直接print即可

```
print("Assortativity:")
print(nx.degree_assortativity_coefficient(G))
```

2.7. Network community

对于此段的分析，有过不太成功的尝试。代码如下：

```
part = community.community_louvain.best_partition(G)
values = [part.get(node) for node in G.nodes()]
nx.draw_spring(
    G, cmap=plt.get_cmap("jet"), node_color=values, node_size=30,
with_labels=False
)
pylab.show()
```

具体的效果如图7所示

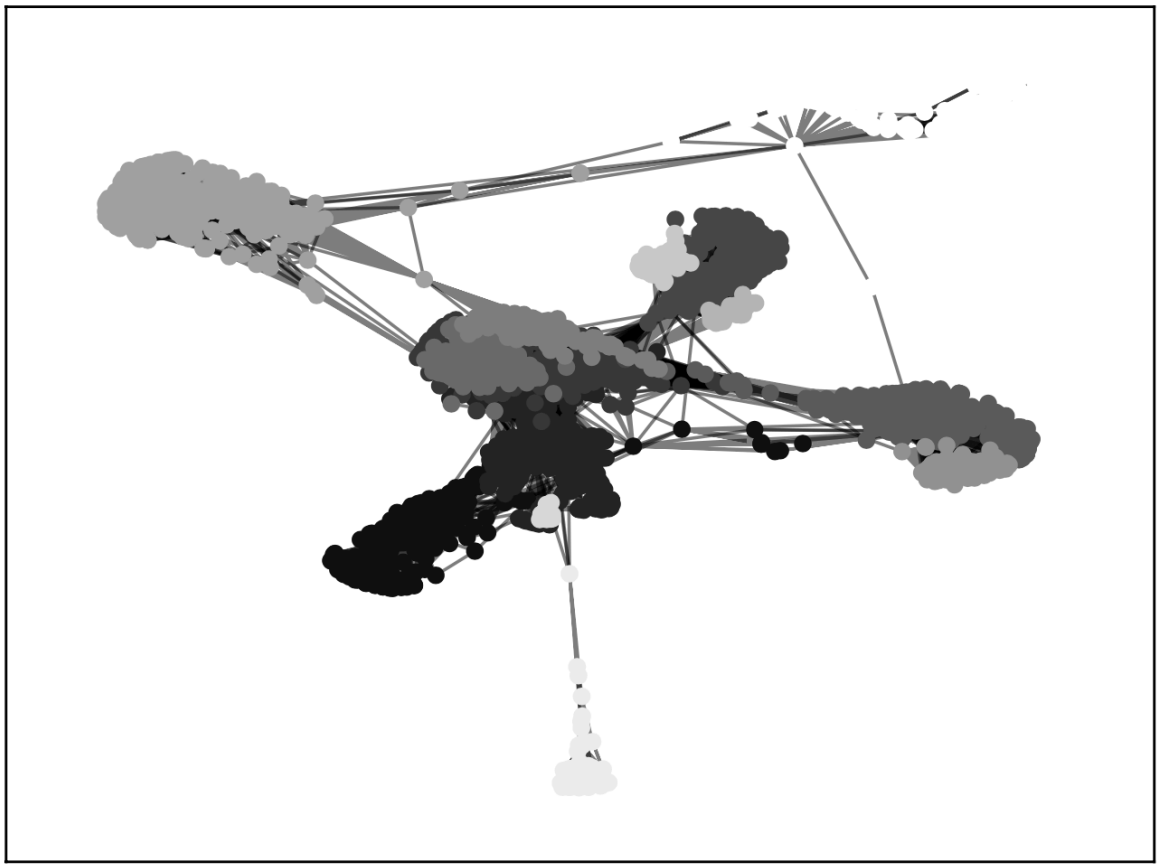

图7.一次失败的尝试

由图可见，由于没有染色，并且网络社区设置的节点数太小，导致图难以识别。经过优化得到如下代码

```
# network community

# first compute the best partition

partition = community_louvain.best_partition(G)
# drawing
size = float(len(set(partition.values())))
pos = nx.spring_layout(G)
count = 0.0
for com in set(partition.values()):
    count = count + 1.0
```

```
    list_nodes = [nodes for nodes in partition.keys() if
partition[nodes] == com]
    nx.draw_networkx_nodes(
        G, pos, list_nodes, node_size=20, node_color=str(count /
size)
    )
nx.draw_networkx_edges(G, pos, alpha=0.5)
plt.show()
pylab.show()
```

得到如图8所示的网络社区

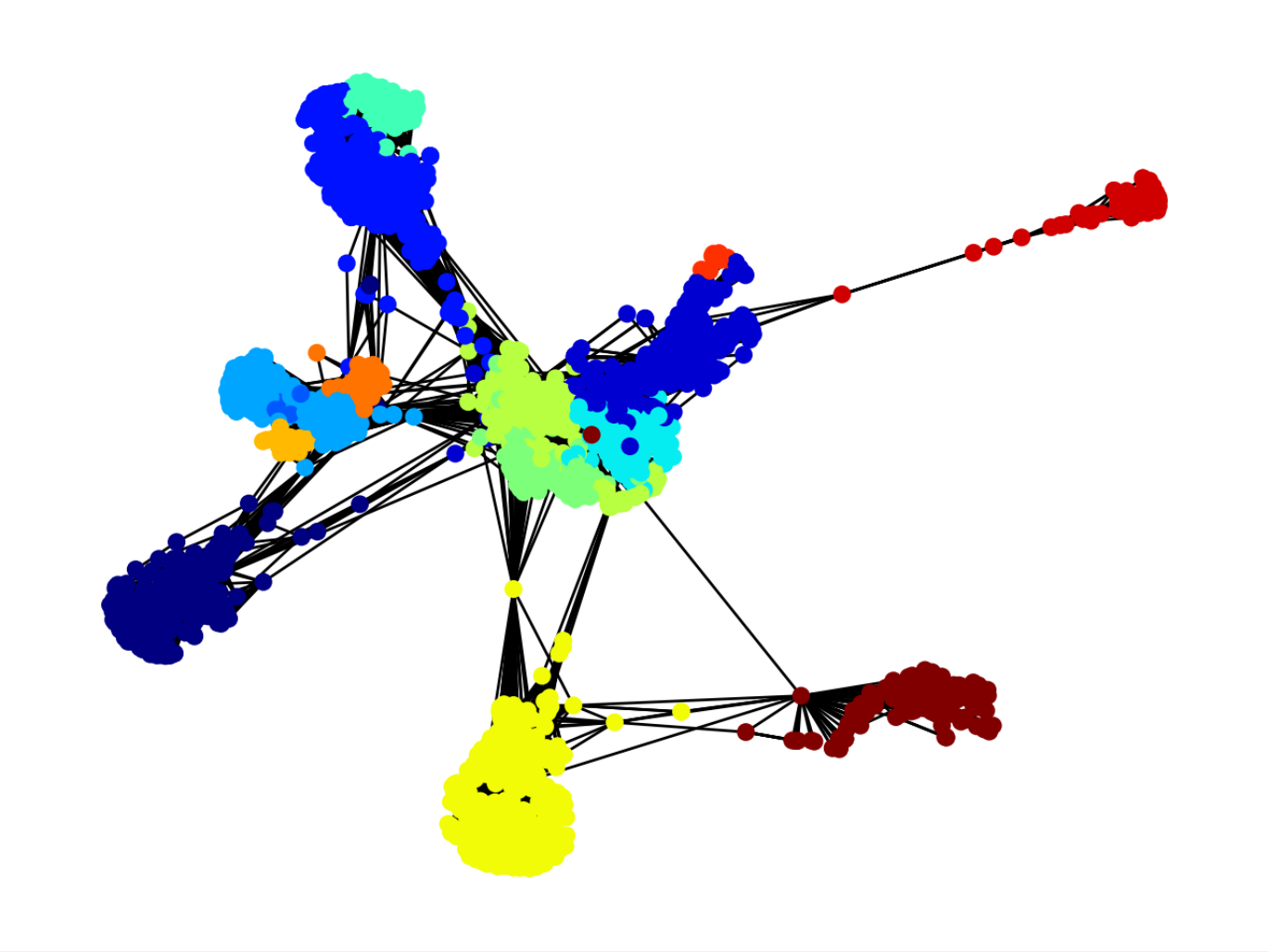

图8.network community

2.8. shrinking diameter

验证思路是：首先读取文件的节点和边，每次向图中加入500个节点，依次计算diameter，代码如下：

```
import community.community_louvain
import networkx as nx
import pandas as pd
import numpy as np
import pylab
import matplotlib.pyplot as plt
from networkx.algorithms.community import k_clique_communities
from community import community_louvain
from random import randint

G = nx.Graph()

path = 'facebook_combined.txt'
edge_list = []
node_set = set()
with open(path, 'r') as f:
    for line in f:
        cols = line.strip().split(' ')
        y1 = int(cols[0])
        y2 = int(cols[1])
        node_set.add(y1)
        node_set.add(y2)
        edge = (y1, y2)  # 元组代表一条边
        edge_list.append(edge)

list_1 = []
list_2 = []


for i in range(0, len(edge_list), 500):
    ad_edges = edge_list[i:i+500]
    G.add_edges_from(ad_edges)
    dia = nx.diameter(G)
    list_1.append(i)
```

```
    list_1.append(dia)

plt.plot(list_1, list_2)
pylab.show()
```

得到如图9所示的图片，验证shrinking diameter

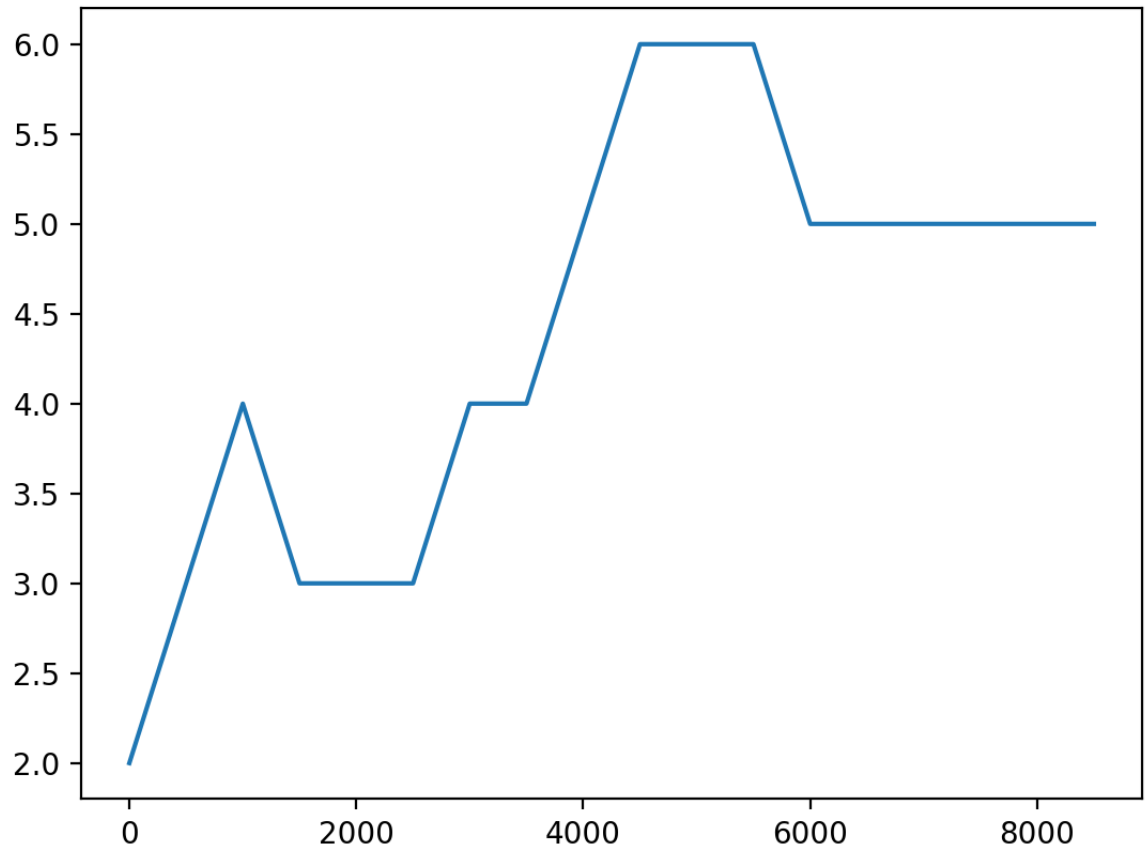

图9.shrinking diameter

3. 实验总结

通过本次实验复习了python的使用技巧，并且熟悉了networkx库。复习了复杂网络的相关知识，初步了解了如何对复杂网络进行分析。