杭州電子科技大学

网络科学前沿专题研究

翻转课堂汇报
网络节点重要性排序专题

学　　院	网络空间安全学院
学　　号	16271110　18271126
学生姓名	陈中渊　潘彬民

演讲提纲

实现搜索引擎

- 爬虫
- 数据库
- 文字匹配引擎
- 重要性排序

网页结构

- 字符串节点（HTML）
- 质量参差不齐（极差大）：PageRank1.1
- 有向图（HTML <a></a>）

基本思路：谁更重要

- 被超链接多的更重要
- 被重要的网页超链接的更重要
- PageRank2.4
- dangling links

算法实现

- PageRank2.6
- Numpy
- Pandas

Query 类型

- —Specific queries. For example, “Does Netscape support the JDK 1.1 code-signing API?” （ Scarcity Problem）

- —Broad-topic queries. For example, “Find information about the Java programming language.”（disambiguation; Abundance Problem）
- — Similar-page queries. For example, “Find pages ‘similar’ to java.sun.com.”

What makes authority

- Lots of back-links?
- Links for navigation purpose
- Relevance and popularity

杂谈

- 聚类在搜索引擎的应用

HITS 算法

- 取子集 R 来工作， P6
- Authority 的性质，P8
- 赋予所有节点 x 属性和 y 属性 If p points to many pages with large x-values, then it should receive a large y-value; and if p is pointed to by manypages with large y-values, then it should receive a large x-value
- 迭代 收敛

代码实现

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import h5py

del_line = '\r\x1b[K'
SAMPLE_SIZE = 1000

def cleaned(df):
    """
    data clean process
    remove dangling links

    Parameters
    ----------
    df: pandas.dataframe
        raw dataframe of edges

    Returns
    -------
    cdf: pandas.dataframe
        cleaned dataframe

    danglings: list
        row indices of the removed
    """

    cdf = df.copy()
    lcdf = len(cdf)

    danglings = []
    ld = len(danglings)

    print('cleaning data frame')
    iteration_times = 1
    while True:
        for index, row in cdf.iterrows():
            if index in danglings:
                cdf = cdf.drop(index)
            elif not (cdf['from'] == row['to']).any():
                danglings.append(index)
                cdf = cdf.drop(index)
            if not index % 77:
                print(f'{del_line}{index / lcdf * 100:2.1f}% #{iteration_times}',
end='')
        iteration_times += 1

        # iterate until `danglings` does not change
        if len(danglings) == ld:
            break
        else:
            ld = len(danglings)

    print(f'{del_line}data cleaned with {iteration_times} iterations')

    return cdf, np.array(danglings)


def page_rank(df):
    """
    PageRank main function

    Parameters
    ----------
    df: pandas.dataframe
        cleaned dataframe of edges

    Returns
    -------
    ranks: dict
        PageRank result
    """
    froms = set(df['from'].unique())
    tos = set(df['to'].unique())
    pages = list(froms | tos)
    pages = sorted(pages)
    lp = len(pages)

    E = np.full(shape=lp, fill_value=1/lp)
    A = pd.DataFrame(
            data=np.zeros((lp, lp)),
            index=pages,
            columns=pages
        )
    for i, row in df.iterrows():
        A[row['from']][row['to']] = 1
    for i, row in A.iterrows():
        A.loc[i][row == 1] = 1 / row.sum()

    Ri = E
    delta = []
    while True:
        Ri_1 = Ri @ A.values
        d = Ri.sum() - Ri_1.sum()
        Ri_1 = Ri_1 + d * E
        delta.append((Ri_1 - Ri).sum())

        Ri = Ri_1

        if len(delta) > 2 and np.abs(delta[-1] - delta[-2]) < 1e-20:
            break


    return dict(zip(pages, Ri))



if __name__ == '__main__':
    hdf_path = 'data/datasets.h5'
    hdf = h5py.File(hdf_path)

    if 'cdf' not in hdf:
        df = pd.read_csv('data/web-Google.txt', sep='\t',
            skiprows=4, header=None, names=['from', 'to'])[:SAMPLE_SIZE]

        cdf, danglings = cleaned(df)
        cdf.to_hdf(hdf_path, 'cdf')
        np.save('data/danglings.npy', danglings)
    else:
        cdf = pd.read_hdf(hdf_path, 'cdf')
        danglings = np.load('data/danglings.npy')
        ranks = page_rank(cdf)
        plt.scatter(range(len(ranks)), ranks.values(), marker='d',
            facecolor='none', edgecolor='indianred')
        plt.show()

    hdf.close()
```