ANALYSIS OF SOCIAL MEDIA PAGES OF DIFFERENT LOCATIONS

Social Media Platform Chosen: Instagram

Team Members:

16BCE0082	**CHAITANYA BHOJWANI**
16BCE0277	**ANISH SAHA**
16BCE0722	**HRISHIKESH BHARADWAJ C**

Report submitted for the final Project Review of

Course Code: CSE3021
Course Title: Social and Information Networks

to

Professor: Vijayasherly V

Slot: A1 + TA1

DECLARATION

We hereby declare that the project entitled "**ANALYSIS OF SOCIAL MEDIA PAGES OF DIFFERENT LOCATIONS**" being submitted in partial fulfilment for the degree of Bachelor of Technology in Computer Science Engineering at Vellore Institute of Technology is the authentic record of our own work done under guidance of our guide Prof. Vijayasherly V.

Chaitanya Bhojwani

B. Tech. Computer Science

Vellore Institute of Technology

Vellore, Tamil Nadu

Anish Saha

B. Tech. Computer Science

Vellore Institute of Technology

Vellore, Tamil Nadu

Hrishikesh Bharadwaj C

B. Tech. Computer Science

Vellore Institute of Technology

Vellore, Tamil Nadu

TABLE OF CONTENTS

1. Abstract

Instagram Pages provide a key aspect of social media trends of locations all around the world. In recent times, social networking sites have provided a medium through which people can see and search and share photos and other things among themselves.

Most location have Instagram pages and regularly upload information about their latest photos from their line-up. This feed, on Search on Instagram, appear on our timeline.

On an Instagram Page we have the option to put hashtags to the photos and also add locations, react and also follow to the posts of some famous locations which means that whenever a new post comes up the user will be notified about the post as soon as the users logs in.

In most experimental approaches for social network analysis the study is based on social interconnection between the user's hashtags and other details of the social network. The engagement between locations and hashtags used on social media is something that is rarely studied upon. In our work, we are showing the interactions between posts and users, rather than interactions among users. We are modelling the network using a web parsing of Instagram using Instagram API and running clique percolation and Community Detection.

Keywords: Instagram, World Famous Locations, Social Networks, Posts, Hashtags, Web Crawling, Web Parsing, Clique Percolation, Community Detection.

2. Introduction

On a public Instagram page user have the ability to comment, like or share posts. This is the interaction between the users with the global page.

The four primary ways people engage with a post is:

- Comments
- Likes
- Shares
- Hashtags

Analysing these metrics will help in determining which posts resonate best with the audience.

Instagram location Pages play a key role in a tourism industry nowadays and hence the importance of the social network analysis of their Instagram Pages plays a key role. We will be performing Social Network Analytics on Instagram Pages of some of the famous and top world locations and perform a comparison between them based on posts published and hashtags used to identify the connectivity, centrality and other metrics in this network.

In the project parsing of the Instagram pages is done using Selenium tool and data about the posts is collected. The data collected includes posts likes reaction time when published and hashtags used according to the location. This data is collected over a period of time so we can get a better understanding of people's opinions on a particular location.

Social network analysis aspects such as parsing, clique percolation and community detection are used to produce useful and meaningful data. Once we get to know what posts and what hashtags people have liked over time, we can use the information for the success of tourism industries.

3. Literature Review

[1] Social Semiotic Aspects of Instagram Social Network

This paper addresses the semiotic aspects of Instagram social network. In this regard, it first discusses differences between the typical semiotic communication model and Instagram social network communication model. Then having discussed design aspects of an Instagram as well as its application user interface, it tries to show how social semiotic requirements can best fit in Instagram as a social networks media for sharing pictures and extending social relations among artists, citizens and business agents. In this paper it will be shown by recruiting a clear social semiotic model that Instagram can also be utilized as a successful platform for pictorial and multimodal sign production and distribution. In the end it discussed the collective semantic intelligence based on Instagram.

[2] Community detection in graphs

The modern science of networks has brought significant advances to our understanding of complex systems. One of the most relevant features of graphs representing real systems is community structure, or clustering, i.e. the organization of vertices in clusters, with many edges joining vertices of the same cluster and comparatively few edges joining vertices of different clusters. Such clusters, or communities, can be considered as fairly independent compartments of a graph, playing a similar role like, e.g., the tissues or the organs in the human body. Detecting communities is of great importance in sociology, biology and computer science, disciplines where systems are often represented as graphs. This problem is very hard and not yet satisfactorily solved, despite the huge effort of a large interdisciplinary community of scientists working on it over the past few years. We will attempt a thorough exposition of the topic, from the definition of the main elements of the problem, to the presentation of most methods developed, with a special focus on techniques designed by statistical physicists, from the discussion of crucial issues like the significance of clustering and how methods should be tested and compared against each other, to the description of applications to real networks.

[3] How unseen communities of Instagram users are revealed using the real-valued collocations of hashtags.

The growing popularity of Instagram social network and millions of daily uploaded photos, provides the possibility to analyse users' interests, emotions and opinions. In this social network that the relationship among users is based on following each other, Photo sharing, writing comments for other users' photos and tagging them are the core activity by which we access to valuable information of users. Monitoring and analysing these information lead to valuable results. An important task in analysing information of social networks is identifying communities. A community is a group which we believe is formed of people based on their relationships and attributes and famous accounts with most followers play the role of communities who attract so many followers based on their friends and common attributes. In this paper, we study the problem of detection the members of these famous accounts as a community detection problem. We develop an overlapping community detection method and our main contribution in this paper is considering continuous node attributes and handle multiplicity of dimensions and high running time properly.

4. Tools Used

Software's used:

- Instagram developer API
- Selenium for Web Parsing
- Java – For API calls and collection of Data
- Python 3.7 for Social Network Analysis

Libraries used:

- For Community Detection
  - NetworkX
  - Community API

Linux Based OS - Ubuntu.

Computer with minimum 4GB RAM and Intel™ i3 processor.

5. Implementation

At first, A Developer Instagram account is created. On-going through all the types of verifications by Instagram this account is made active. On getting an active account, a user can create an app within it. This app needs to get some permission from the Instagram App handling and verification department. The procedure is time taking.

That is why, a method which can function in the same manner is found, that is by parsing the respective pages and getting output in JSON format.

Web Crawling:

A web crawler (also known as a web spider or web robot) is a program or automated script which browses the World Wide Web in a methodical, automated manner.

This process is called Web crawling or spidering.

Many legitimate sites, in particular search engines, use spidering as a means of providing up-to-date data.

Web crawlers are mainly used to create a copy of all the visited pages for later processing by a search engine, that will index the downloaded pages to provide fast searches.

Crawlers can also be used for automating maintenance tasks on a Web site, such as checking links or validating HTML code.

Also, crawlers can be used to gather specific types of information from Web pages, such as harvesting e-mail addresses (usually for spam). It is a type of Web Scraping in which the data visible over the browser to a user can be saved on the user system for analysis and many other purposes. This data can be saved in many forms including SQL, JSON, CSV, Spreadsheet, etc.

Web scraping software automatically loads and extracts data from page based on user's requirement.

After the data is obtained with the help of this software in JSON format, it is converted to CSV in this case and then one by one, loaded to python code for social network analysis.

The data that this project uses is fetched in JSON format using the tool called **Selenium**. This JSON format is first converted to CSV Format for further processing.

In this project data from the online social media site Instagram about two hundred locations from around the world which are famous on Instagram are collected.

- From the pages of all the locations on Facebook, Data about most recent posts like

1. Number of Reactions on posts.
2. Number of Comments on posts.
3. Number of Shares that a post gets.
4. Hashtags Used in post.
5. Time when the post is uploaded.

is collected.

- In this project, the mentioned data is collected for the posts dated back up to the month of July.
- After this clique percolation and community detection is applied on the data to get meaningful information as output.

Clique Percolation: The clique percolation method is a popular approach for analysing the overlapping community structure of networks. The term network community (also called a module, cluster or cohesive group) has no widely accepted unique definition and it is usually defined as a group of nodes that are more densely connected to each other than to other nodes in the network. There are numerous alternative methods for detecting communities in networks, for example, the Girvan–Newman algorithm, hierarchical clustering and modularity maximization.

Community Detection: In the study of complex networks, a network is said to have community structure if the nodes of the network can be easily grouped into (potentially overlapping) sets of nodes such that each set of nodes is densely connected internally. In the particular case of non-overlapping community finding, this implies that the network divides naturally into groups of nodes with dense connections internally and sparser connections between groups. But overlapping communities are also allowed. The more general definition is based on the principle that pairs of nodes are more likely to be connected if they are both members of the same communities, and less likely to be connected if they do not share communities. A related but different problem is community search, here the goal is to find a community that a certain vertex belongs to.

- In this project we use NetworkX package in python to perform Clique Percolation and Community Detection. It uses the best partition method for community detection and uses Louvain Algorithm.

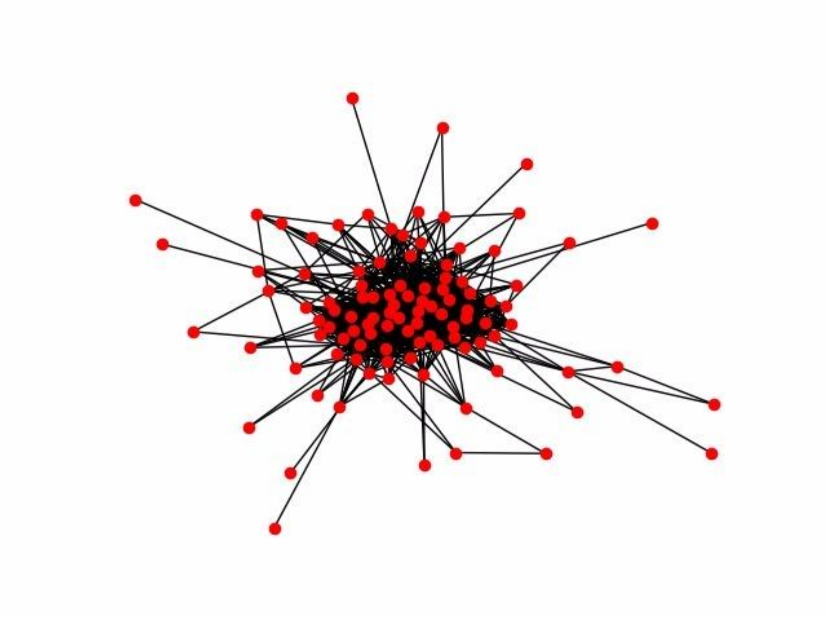
  Best Partition Method: Computes the partition of the graph nodes which maximises the modularity (or try...) using the Louvain heuristics. This is the partition of highest modularity, i.e. the highest partition of the dendrogram generated by the Louvain algorithm.

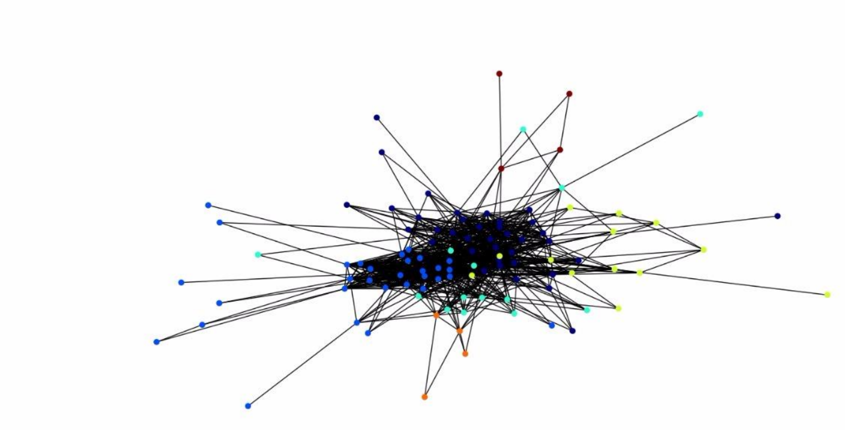
  Louvain Algorithm: The Louvain Method for community detection is a method to extract communities from large networks created by Blondel et al. from the University of Louvain (affiliation of authors has given the method its name). The method is a greedy optimization method that appears to run in time O (n log n).

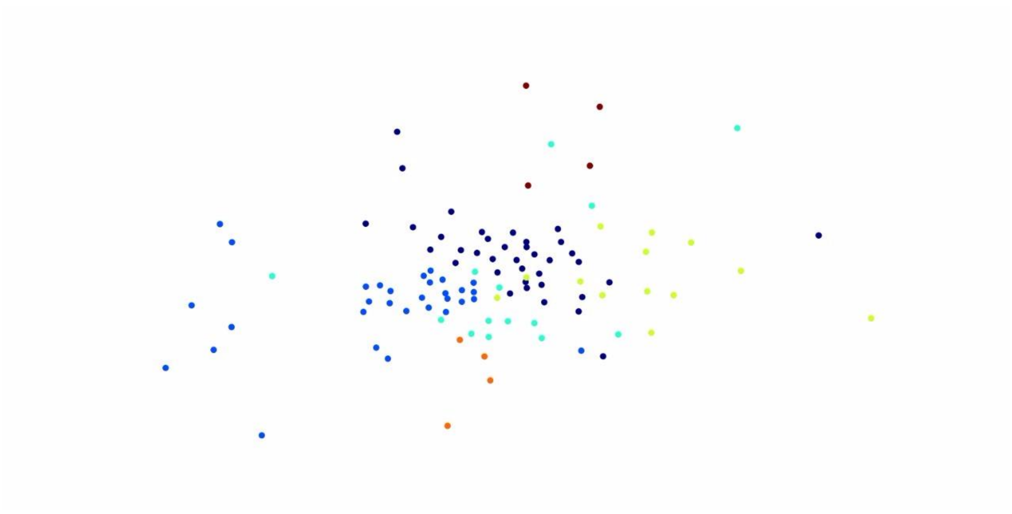
The Louvain method is a simple, efficient and easy-to-implement method for identifying communities in large networks. The method has been used with success for networks of many different type (see references below) and for sizes up to 100 million nodes and billions of links. The analysis of a typical network of 2 million nodes takes 2 minutes on a standard PC. The method unveils hierarchies of communities and allows to zoom within communities to discover sub-communities, sub-sub-communities, etc. It is today one of the most widely used method for detecting communities in large networks.

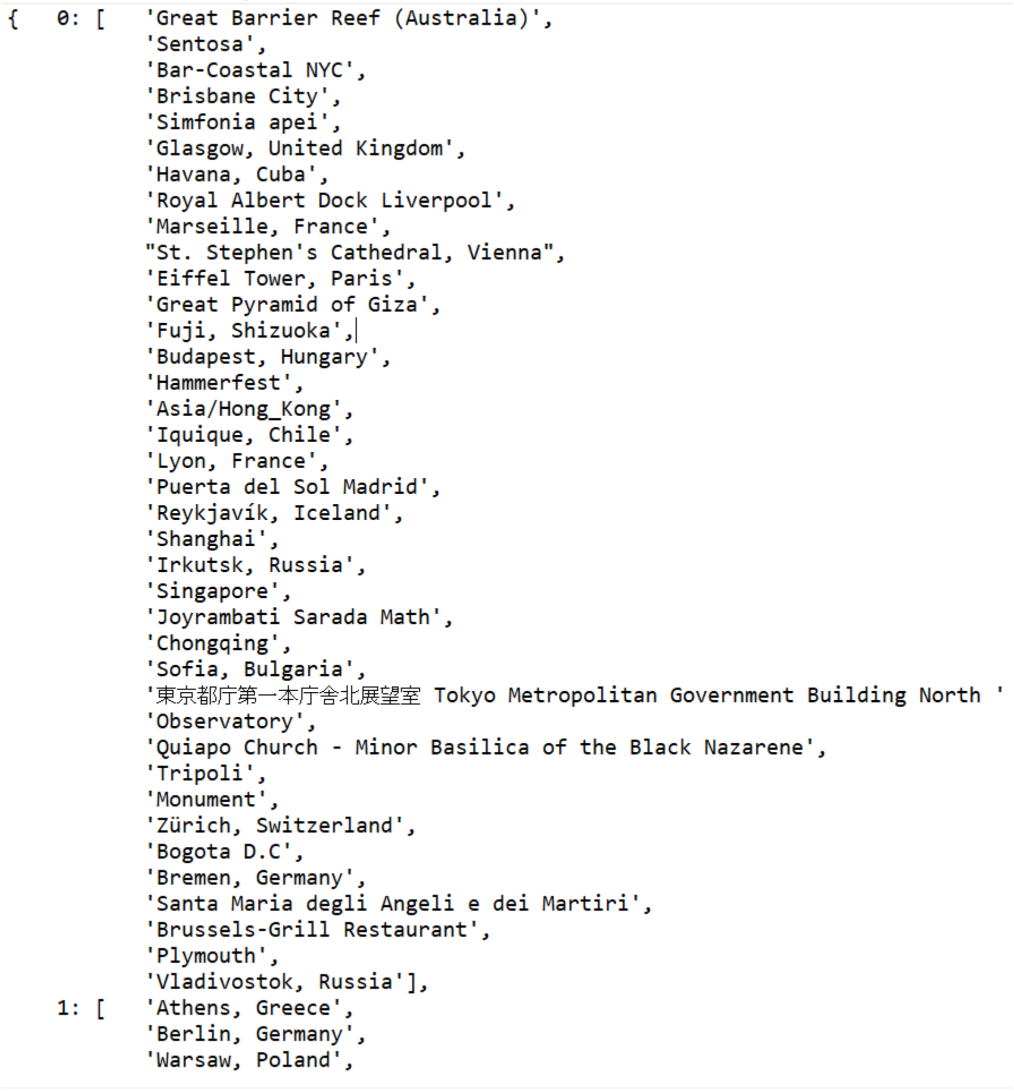
6. Results and Discussions

Communities Detected and Graphs with locations as nodes and Edges as edges

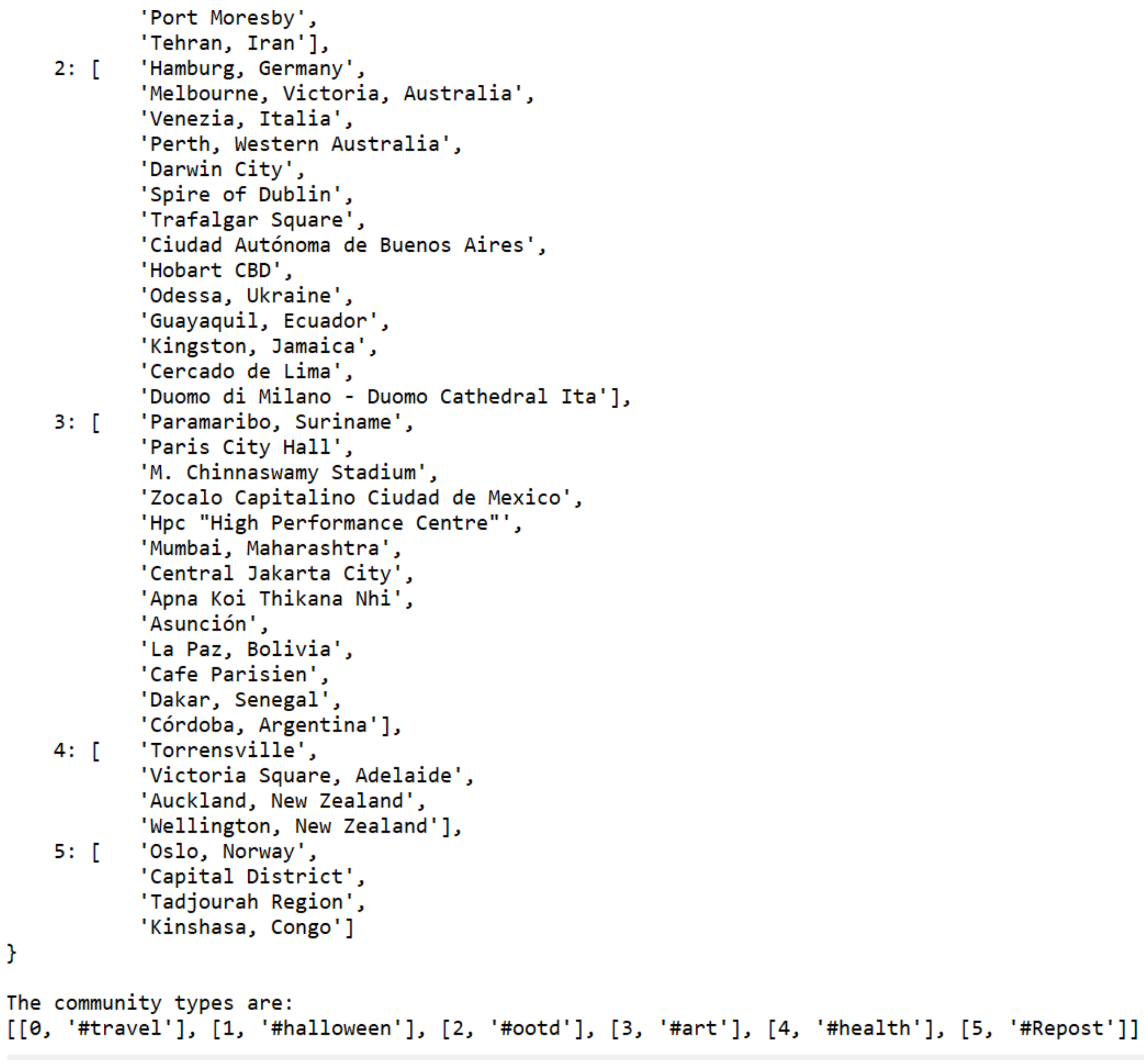








```
{   0: [   'Great Barrier Reef (Australia)',
           'Sentosa',
           'Bar-Coastal NYC',
           'Brisbane City',
           'Simfonia apei',
           'Glasgow, United Kingdom',
           'Havana, Cuba',
           'Royal Albert Dock Liverpool',
           'Marseille, France',
           "St. Stephen's Cathedral, Vienna",
           'Eiffel Tower, Paris',
           'Great Pyramid of Giza',
           'Fuji, Shizuoka',
           'Budapest, Hungary',
           'Hammerfest',
           'Asia/Hong_Kong',
           'Iquique, Chile',
           'Lyon, France',
           'Puerta del Sol Madrid',
           'Reykjavík, Iceland',
           'Shanghai',
           'Irkutsk, Russia',
           'Singapore',
           'Joyrambati Sarada Math',
           'Chongqing',
           'Sofia, Bulgaria',
           '東京都庁第一本庁舎北展望室 Tokyo Metropolitan Government Building North '
           'Observatory',
           'Quiapo Church - Minor Basilica of the Black Nazarene',
           'Tripoli',
           'Monument',
           'Zürich, Switzerland',
           'Bogota D.C',
           'Bremen, Germany',
           'Santa Maria degli Angeli e dei Martiri',
           'Brussels-Grill Restaurant',
           'Plymouth',
           'Vladivostok, Russia'],
    1: [   'Athens, Greece',
           'Berlin, Germany',
           'Warsaw, Poland',
```



```
            'Port Moresby',
            'Tehran, Iran'],
    2: [    'Hamburg, Germany',
            'Melbourne, Victoria, Australia',
            'Venezia, Italia',
            'Perth, Western Australia',
            'Darwin City',
            'Spire of Dublin',
            'Trafalgar Square',
            'Ciudad Autónoma de Buenos Aires',
            'Hobart CBD',
            'Odessa, Ukraine',
            'Guayaquil, Ecuador',
            'Kingston, Jamaica',
            'Cercado de Lima',
            'Duomo di Milano - Duomo Cathedral Ita'],
    3: [    'Paramaribo, Suriname',
            'Paris City Hall',
            'M. Chinnaswamy Stadium',
            'Zocalo Capitalino Ciudad de Mexico',
            'Hpc "High Performance Centre"',
            'Mumbai, Maharashtra',
            'Central Jakarta City',
            'Apna Koi Thikana Nhi',
            'Asunción',
            'La Paz, Bolivia',
            'Cafe Parisien',
            'Dakar, Senegal',
            'Córdoba, Argentina'],
    4: [    'Torrensville',
            'Victoria Square, Adelaide',
            'Auckland, New Zealand',
            'Wellington, New Zealand'],
    5: [    'Oslo, Norway',
            'Capital District',
            'Tadjourah Region',
            'Kinshasa, Congo']
}

The community types are:
[[0, '#travel'], [1, '#halloween'], [2, '#ootd'], [3, '#art'], [4, '#health'], [5, '#Repost']]
```

7. Conclusion

In the project, comparisons were made among locations using clique percolation and community detection on the data collected by scraping data from Instagram pages of locations and hashtags.

The comparisons were made and the resulting communities which were found in form of hashtags were:

1. #travel
2. #halloween
3. #ootd
4. #art
5. #health

6. #Repost

The graphs for these were also made using Python NetworkX library,

As the most significant data would be the hashtags and their number in this case, the whole procedure can have been done with Hashtags VS Hashtags graphical method if more than 2 locations are selected for similar type of comparison.

From this analysis, finally it can be said that the listed hashtags have better presence and reach in the online social media platform of Instagram for the given locations. This definitely creates an image in the minds of travel customers and can easily tell and check which place is famous for which thing and this can also benefit travel plans and agents.

References

[1] Mirsarraf, Mohammadreza, Hamidreza Shairi, and Abotorab Ahmadpanah. "Social semiotic aspects of instagram social network." *INnovations in Intelligent SysTems and Applications (INISTA), 2017 IEEE International Conference on*. IEEE, 2017.

[2] Fortunato, Santo. "Community detection in graphs." *Physics reports* 486.3-5 (2010): 75-174.

[3] Bejandi, Sarah Abdollahi, and Ali Katanforoush. "How unseen communities of instagram users are revealed using the real-valued collocations of hashtags." *Knowledge-Based Engineering and Innovation (KBEI), 2017 IEEE 4th International Conference on*. IEEE, 2017.

Appendix – Code

1. Code for scrapping the messages.

```
from selenium import webdriver
from selenium.common.exceptions import TimeoutException
from selenium.webdriver.common.by import By
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time
import json

class InstagramCrawler:
    BASE_URL = 'https://www.instagram.com/explore/locations/'
    def __init__(self):
        chrome_options = webdriver.ChromeOptions()
        prefs = {"profile.default_content_setting_values.notifications": 2}
        chrome_options.add_experimental_option("prefs", prefs)
        chrome_options.add_argument('--headless')
        chrome_options.add_argument('--disable-gpu')

        self.driver = webdriver.Chrome(chrome_options=chrome_options)
        self.wait = WebDriverWait(self.driver, 10)
    def open_location(self, id):
        self.driver.get(self.BASE_URL+str(id)+'/media/')
        # wait for the login page to load
        self.wait.until(EC.visibility_of_element_located((By.XPATH,
'//*[@id="react-root"]/section/main/article/header/div[1]/canvas')))

        location = self.driver.find_element_by_css_selector('#react-root >
section > main > article > header > div > h1').text
        posts = []
        for i in range(1,4):
            for j in range(1,4):
                try:
                    href =
self.driver.find_element_by_css_selector('#react-root > section > main >
article > div > div > div > div:nth-child('+str(i)+') > div:nth-
child('+str(j)+') > a').get_attribute("href")
                    # print('link:',href)
                    posts.append(href)
                except:
                    pass

        hashtags = []
        hashtags_list = []
        for post in posts:
            h = self.get_hashtags(post)
            if h:
                hashtags.append(h)
                hashtags_list += h['hashtags']

        print('Location ID:',id)
        print('Location:',location)
        print('Hashtags:',hashtags)

        obj = {}
        obj['location'] = location
        obj['hashtags'] = hashtags
        obj['hashtags_all'] = hashtags_list

        obj1 = {}
        with open('location_data.txt') as file:
            obj1 = json.load(file) # use `json.load` to do the reverse
        obj1[id] = obj

        with open('location_data.txt', 'w') as file:
            file.write(json.dumps(obj1)) # use `json.loads` to do the
reverse

    def get_hashtags(self, link):
        # navigate to "post" page
        self.driver.get(link)
        # wait for the main page to load
        # time.sleep(1)

        try:
            e = self.driver.find_elements_by_css_selector('#react-root >
section > main > div > div > article > div > div > ul > li:nth-child(1) >
div > div > div > span > a')
            e += self.driver.find_elements_by_css_selector('#react-root >
section > main > div > div > article > div > div > ul > li:nth-child(2) >
div > div > div > span > a')
            e += self.driver.find_elements_by_css_selector('#react-root >
section > main > div > div > article > div > div > ul > li:nth-child(3) >
div > div > div > span > a')
        except:
            pass

        hashtags = []

        for a in e:
            print
            if a.text.find("#")>=0:
                hashtags.append(a.text)

        timestamp = ''
        try:
            timestamp = self.driver.find_element_by_css_selector('#react-
root > section > main > div > div > article > div > div > a >
time').get_attribute("datetime")
        except:
            pass
        obj = {}
        obj['time']=timestamp
        obj['hashtags']=hashtags
        return obj

if __name__ == '__main__':
    crawler = InstagramCrawler()

    f = open('insta_loc_id.txt')
    lines = f.readlines()
    for line in lines:
        locid = line.strip()
        crawler.open_location(id=locid)
```

2. Code for Community detection and Graph drawing:

```
import json
import networkx as nx
import matplotlib.pyplot as plt

data = {}
with open('location_data.txt') as file:
    data = json.load(file)

graph = {}
edge_label = {}
G = nx.Graph()

loc_hash = {}

i=1
for locID1 in data:
    hashtags_all1 = set(data[locID1]['hashtags_all'])
    loc_hash[data[locID1]["location"]] = hashtags_all1
    for locID2 in data:
        if locID1 != locID2:
            hashtags_all2 = set(data[locID2]['hashtags_all'])
            inter = hashtags_all1.intersection(hashtags_all2)
            if inter:
                if (data[locID2]['location'],data[locID1]['location']) not
in graph.keys():

print((data[locID1]['location'],data[locID2]['location']))

graph[(data[locID1]['location'],data[locID2]['location'])]=i
                    edge_label[i] = inter
                    i+=1

G.add_edge(data[locID1]['location'],data[locID2]['location'])

#community detect - CPM
from networkx.algorithms.community import k_clique_communities
coms = list(k_clique_communities(G, 4))
list(coms[0])
#----------------

print(edge_label)
pos = nx.spring_layout(G)
plt.figure()
nx.draw(G,pos,edge_color='black',width=1,linewidths=1,node_size=500,node_co
lor='pink',alpha=0.9,labels={node:node for node in G.nodes()})
nx.draw_networkx_edge_labels(G,pos,edge_labels=graph,font_color='red')
plt.axis('off')
plt.show()

s_p = nx.spring_layout(G)
plt.axis("off")
nx.draw_networkx(G, pos = s_p, with_labels = False, node_size=35)
plt.show()

# communities
'''
Compute the partition of the graph nodes which maximises the modularity (or
try..)
```

```
using the Louvain heuristices. This is the partition of highest modularity,
i.e. the
highest partition of the dendrogram generated by the Louvain algorithm.
'''
import community
s_p = nx.spring_layout(G)
parts = community.best_partition(G)
values = [parts.get(node) for node in G.nodes()]

comm_loc = {}
comm_hashtags = {}
for v in values:
    l = []
    comm_hashtags[v]=[]
    for p in parts:
        if parts[p] == v:
            l.append(p)
            comm_hashtags[v].extend(loc_hash[p])
    comm_loc[v]=l
    comm_hashtags[v] = [comm_hashtags[v],
max(comm_hashtags[v],key=comm_hashtags[v].count)]

print("\n\n\n\n\n The communities Detected are: \n\n\n")
import pprint
pprint.PrettyPrinter(indent=4).pprint(comm_loc)

print("\n\n The community types are: \n")
print([ [k,comm_hashtags[k][1]] for k in comm_hashtags])


plt.axis("off")
nx.draw_networkx(G, pos = s_p, cmap = plt.get_cmap("jet"), node_color =
values, node_size = 35, with_labels = False, width = 0)
plt.show()
#-------------
```