Geographic Clustering and Visualization with Processing

Anastasios Noulas
Computer Laboratory, University of Cambridge

@ University of Namur, March 2014
What does Foursquare data look like?

a temporal window to our activities ...
What does Foursquare look like? (2)

new york noon

new york night

with geographical lens ...
A question ... and an idea!

Q: - How similar are two areas of a city or across cities?

I: - Isolate individual neighbourhoods and model activity at those
Modelling a neighbourhood ...

2 key modelling aspects:

Activity via place categories

Popularity via #checkins

Each square area a multi-dimensional vector
Area clustering (1)

\[ \text{sim}_{a,b} = \frac{\text{cs}_a \cdot \text{cs}_b}{||\text{cs}_a|| \cdot ||\text{cs}_b||} \]

\[ G : \text{graph} \]
(V:areas, E:similarity)
Area clustering (2)

\[ \text{sim}_{a,b} = \frac{\text{cS}_a \cdot \text{cS}_b}{\|\text{cS}_a\| \|\text{cS}_b\|} \]

\[ G : \text{graph} \]
\[ (V: \text{areas}, E: \text{similarity}) \]

\[ G \] is used to produce a degree \( D \) and a weight \( W \) matrix

Graph Laplacian

\[ L = I - D^{-1} - W \]

Spectral Clustering

- Dimensionality Reduction
- k means

a tutorial on spectral clustering by Ulrike von Luxburg

http://www.kyb.mpg.de/fileadmin/user_upload/files/publications/attachments/Luxburg07_tutorial_4488%5b0%5d.pdf
Clustering Geographical areas: results (1)
Clustering Geographical areas: results (2)

new york

![Map of New York City with clustering results]

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Parks (0.77)</td>
<td>Home (0.66)</td>
<td>Food(0.57)</td>
<td>Food(0.37)</td>
<td>Arts (0.55)</td>
<td>Nightlife(0.61)</td>
<td>Travel(0.36)</td>
<td>Food(0.42)</td>
</tr>
<tr>
<td>Home (0.05)</td>
<td>Parks (0.09)</td>
<td>Shops(0.13)</td>
<td>Shops(0.17)</td>
<td>Parks (0.17)</td>
<td>Food(0.07)</td>
<td>Shops(0.31)</td>
<td>Shops(0.09)</td>
</tr>
<tr>
<td>Nightlife(0.05)</td>
<td>College (0.06)</td>
<td>Home (0.12)</td>
<td>Home (0.12)</td>
<td>Food(0.07)</td>
<td>Home (0.09)</td>
<td>Food(0.12)</td>
<td>Nightlife(0.31)</td>
</tr>
<tr>
<td>Shops(0.04)</td>
<td>Travel(0.05)</td>
<td>Nightlife(0.1)</td>
<td>Travel(0.09)</td>
<td>Home (0.06)</td>
<td>Shops(0.07)</td>
<td>Home (0.07)</td>
<td>Food(0.12)</td>
</tr>
<tr>
<td>Food(0.03)</td>
<td>Food(0.04)</td>
<td>Parks (0.05)</td>
<td>Travel(0.09)</td>
<td>Home (0.06)</td>
<td>Shops(0.05)</td>
<td>Home (0.05)</td>
<td>Shops(0.09)</td>
</tr>
<tr>
<td>Other(0.06)</td>
<td>Other(0.1)</td>
<td>Other(0.09)</td>
<td>Other(0.15)</td>
<td>Other(0.08)</td>
<td>Other(0.05)</td>
<td>Other(0.1)</td>
<td>Other(0.08)</td>
</tr>
</tbody>
</table>

The map shows the clustering of geographical areas in New York City. Each cluster is represented by a different color, and the table below lists the distribution of various categories within each cluster.
- Discover similar neighbourhoods across cities
- And differences between them ...
## Clustering foursquare users

### New York

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Home (0.51)</th>
<th>Food (0.15)</th>
<th>Nightlife (0.1)</th>
<th>Shops (0.07)</th>
<th>Travel (0.05)</th>
<th>Other (0.12)</th>
</tr>
</thead>
</table>

### London

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Nightlife (0.61)</th>
<th>Food (0.18)</th>
<th>Home (0.06)</th>
<th>Travel (0.04)</th>
<th>Parks (0.03)</th>
<th>Other (0.08)</th>
</tr>
</thead>
</table>

### Notes

- **Food Fanatics and Shopping Freaks!**
- **Pub Lovers and Commuting Robots!**
Topic Modeling: Clustering Documents

**Seeking Life's Bare (Genetic) Necessities**

Cold Spring Harbor, New York—How many genes does an organism need to sustain life? Last week at the genetic institute here, two genome researchers with radically different approaches presented complementary views of the basic genes needed for life.

One research team, using comparative analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, these predictions:


http://videolectures.net/mlss09uk_blei_tm/

Topic Modeling: Clustering Area Documents

<table>
<thead>
<tr>
<th>London*</th>
<th>time</th>
<th>good</th>
<th>great</th>
<th>nice</th>
<th>waiting</th>
<th>place</th>
<th>back</th>
<th>love</th>
<th>free</th>
</tr>
</thead>
<tbody>
<tr>
<td>work</td>
<td>back</td>
<td>week</td>
<td>coffee</td>
<td>working</td>
<td>today</td>
<td>office</td>
<td>busy</td>
<td>flat</td>
<td>latte</td>
</tr>
<tr>
<td>dinner</td>
<td>drinks</td>
<td>night</td>
<td>food</td>
<td>birthday</td>
<td>evening</td>
<td>tonight</td>
<td>party</td>
<td>watching</td>
<td>drink</td>
</tr>
<tr>
<td>food</td>
<td>amazing</td>
<td>love</td>
<td>chicken</td>
<td>happy</td>
<td>street</td>
<td>team</td>
<td>favourite</td>
<td>birthday</td>
<td>lovely</td>
</tr>
<tr>
<td>pint</td>
<td>beer</td>
<td>lovely</td>
<td>pizza</td>
<td>chips</td>
<td>drink</td>
<td>england</td>
<td>fish</td>
<td>local</td>
<td>wine</td>
</tr>
<tr>
<td>meeting</td>
<td>event</td>
<td>office</td>
<td>media</td>
<td>beautiful</td>
<td>session</td>
<td>business</td>
<td>social</td>
<td>building</td>
<td>today</td>
</tr>
<tr>
<td>train</td>
<td>home</td>
<td>back</td>
<td>heading</td>
<td>london</td>
<td>waiting</td>
<td>bound</td>
<td>station</td>
<td>trains</td>
<td>homeward</td>
</tr>
<tr>
<td>lunch</td>
<td>coffee</td>
<td>afternoon</td>
<td>today</td>
<td>sunday</td>
<td>brunch</td>
<td>lunchtime</td>
<td>meetings</td>
<td>food</td>
<td>salad</td>
</tr>
<tr>
<td>morning</td>
<td>breakfast</td>
<td>coffee</td>
<td>today</td>
<td>early</td>
<td>start</td>
<td>friday</td>
<td>meetings</td>
<td>monday</td>
<td>bacon</td>
</tr>
<tr>
<td>shopping</td>
<td>shop</td>
<td>buying</td>
<td>place</td>
<td>checking</td>
<td>bought</td>
<td>store</td>
<td>stop</td>
<td>shoes</td>
<td>champagne</td>
</tr>
</tbody>
</table>

TABLE II: Top 10 topics (in rows) inferred in London. The background topic is marked with a star.

<table>
<thead>
<tr>
<th>Time*</th>
<th>love</th>
<th>back</th>
<th>great</th>
<th>good</th>
<th>finally</th>
<th>free</th>
<th>waiting</th>
<th>stop</th>
<th>place</th>
</tr>
</thead>
<tbody>
<tr>
<td>party</td>
<td>happy</td>
<td>birthday</td>
<td>bday</td>
<td>friends</td>
<td>place</td>
<td>drink</td>
<td>show</td>
<td>night</td>
<td>work</td>
</tr>
<tr>
<td>place</td>
<td>pizza</td>
<td>yummy</td>
<td>late</td>
<td>happy</td>
<td>favorite</td>
<td>cheese</td>
<td>delicious</td>
<td>chocolate</td>
<td>meal</td>
</tr>
<tr>
<td>work</td>
<td>coffee</td>
<td>back</td>
<td>today</td>
<td>good</td>
<td>iced</td>
<td>office</td>
<td>working</td>
<td>latte</td>
<td>busy</td>
</tr>
<tr>
<td>dinner</td>
<td>night</td>
<td>drinks</td>
<td>tonight</td>
<td>event</td>
<td>food</td>
<td>good</td>
<td>hour</td>
<td>home</td>
<td>sushi</td>
</tr>
<tr>
<td>working</td>
<td>meeting</td>
<td>today</td>
<td>weekend</td>
<td>friday</td>
<td>class</td>
<td>break</td>
<td>money</td>
<td>making</td>
<td>social</td>
</tr>
<tr>
<td>food</td>
<td>quick</td>
<td>picking</td>
<td>hungry</td>
<td>chicken</td>
<td>sushi</td>
<td>stuff</td>
<td>gettin</td>
<td>fresh</td>
<td>eating</td>
</tr>
<tr>
<td>event</td>
<td>meeting</td>
<td>week</td>
<td>york</td>
<td>great</td>
<td>hotel</td>
<td>team</td>
<td>media</td>
<td>room</td>
<td>checking</td>
</tr>
<tr>
<td>lunch</td>
<td>brunch</td>
<td>today</td>
<td>salad</td>
<td>sunday</td>
<td>sandwich</td>
<td>time</td>
<td>place</td>
<td>breakfast</td>
<td>burger</td>
</tr>
<tr>
<td>wine</td>
<td>drinks</td>
<td>amazing</td>
<td>restaurant</td>
<td>cocktails</td>
<td>friend</td>
<td>date</td>
<td>hanging</td>
<td>eating</td>
<td>dessert</td>
</tr>
</tbody>
</table>

TABLE III: Top 10 topics (in rows) inferred in New York. The background topic is marked with a star.
### Topic Modeling: Clustering Area Documents

<table>
<thead>
<tr>
<th>Category</th>
<th>Topic top words</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Hotel</strong></td>
<td>chicago hotel view nice floor river tour beautiful downtown work today meeting working coffee back office days business party night drinks stop drink tonight show late friends</td>
</tr>
<tr>
<td><strong>Train</strong></td>
<td>home back train heading headed waiting late line downtown work today meeting working coffee back office days business stop favorite late hair break shop quick afternoon cool</td>
</tr>
<tr>
<td><strong>Field</strong></td>
<td>softball game team tennis park soccer league playoffs playing work today meeting working coffee back office days business stop favorite late hair break shop quick afternoon cool</td>
</tr>
<tr>
<td><strong>Gym/Fitness</strong></td>
<td>workout time fitness working work cardio yoga sexy body work today meeting working coffee back office days business morning coffee breakfast work early today start week ready</td>
</tr>
<tr>
<td><strong>Fast Food</strong></td>
<td>food place love yummy good cheese delicious chicken hungry lunch salad lunchtime sandwich chicken burger soup eating pizza work today meeting working coffee back office days business</td>
</tr>
</tbody>
</table>

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Table VII: Most frequent place categories in Chicago along with the three most prominent topics (one per line) at places belonging to that category.</td>
<td></td>
</tr>
</tbody>
</table>
We have analysed the dataset using multiple topic model configurations, to reveal patterns of different temporal granularity. For example, if we use just the five time-of-day slots mentioned earlier, the topic counts per time slot can show the least entropic topic, whereas the most entropic one. Times Square stands out in New York as the least entropic topic, whereas in San Francisco the topic with extreme entropy values is dinner. We give some examples for London in Table VI: Most (top part with grey background) and least entropic Topics. We abbreviate Chicago (CHI), London (LDN), New York (NYC) and San Francisco (SF).

### Table VI: Most (top part with grey background) and least entropic Topics

<table>
<thead>
<tr>
<th>Location</th>
<th>Most Entropic Topic</th>
<th>Least Entropic Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHI</td>
<td>dinner</td>
<td>Times Square</td>
</tr>
<tr>
<td>LDN</td>
<td>Times Square</td>
<td>dinner</td>
</tr>
<tr>
<td>NYC</td>
<td>Times Square</td>
<td>dinner</td>
</tr>
<tr>
<td>SF</td>
<td>Times Square</td>
<td>dinner</td>
</tr>
</tbody>
</table>

### Table V: Top 10 topics (in rows) inferred in San Francisco. The background topic is marked with a star.

<table>
<thead>
<tr>
<th>Topic</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>birthday</td>
<td>5.8</td>
</tr>
<tr>
<td>giants</td>
<td>5.0</td>
</tr>
<tr>
<td>dinnner</td>
<td>4.9</td>
</tr>
<tr>
<td>happy</td>
<td>4.7</td>
</tr>
<tr>
<td>food</td>
<td>4.6</td>
</tr>
<tr>
<td>event</td>
<td>4.5</td>
</tr>
<tr>
<td>place</td>
<td>4.5</td>
</tr>
<tr>
<td>food</td>
<td>4.5</td>
</tr>
<tr>
<td>place</td>
<td>4.5</td>
</tr>
<tr>
<td>food</td>
<td>4.5</td>
</tr>
</tbody>
</table>

### Table IV: Top 10 topics (in rows) inferred in Chicago. The background topic is marked with a star.

<table>
<thead>
<tr>
<th>Topic</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>birthday</td>
<td>5.8</td>
</tr>
<tr>
<td>giants</td>
<td>5.0</td>
</tr>
<tr>
<td>dinnner</td>
<td>4.9</td>
</tr>
<tr>
<td>happy</td>
<td>4.7</td>
</tr>
<tr>
<td>food</td>
<td>4.6</td>
</tr>
<tr>
<td>event</td>
<td>4.5</td>
</tr>
<tr>
<td>place</td>
<td>4.5</td>
</tr>
<tr>
<td>food</td>
<td>4.5</td>
</tr>
<tr>
<td>place</td>
<td>4.5</td>
</tr>
<tr>
<td>food</td>
<td>4.5</td>
</tr>
</tbody>
</table>

Work intensity is high only during normal office hours, except during lunch break when we expect much fewer meetings to take place. Indeed the temporal evolution of prominent topics' intensities follows closely the rhythm of life of large cities in the industrialised world. We give some examples for London in Figure 5. For instance, work-related topics peak from Monday to Friday (cf. Figure 5(a)), while the values are much lower on weekends. There is an intra-day pattern as well: The topic's intensity peaks in the evenings when people are most active. The same applies to the topic (cf. Figure 5(b)), but will not be able to determine that day-of-the-week intensity is high only during normal office hours, except during lunch hour.
Clustering Areas: NLP Topics over Time

weekly trends of different topics
Clustering Areas: NLP Topics over Space

(a) London: Cinema topic
(b) Chicago: Lake topic
(c) New York: Brooklyn topic
(d) San Francisco: Beautiful topic

Talking Places: Modelling and Analysing Linguistic Content in Foursquare
Spectral Clustering: practical...
Unnormalized spectral clustering

Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number $k$ of clusters to construct.
- Construct a similarity graph by one of the ways described in Section 2. Let $W$ be its weighted adjacency matrix.
- Compute the unnormalized Laplacian $L$.
- Compute the first $k$ eigenvectors $u_1, \ldots, u_k$ of $L$.
- Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, \ldots, u_k$ as columns.
- For $i = 1, \ldots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the $i$-th row of $U$.
- Cluster the points $(y_i)_{i=1,\ldots,n}$ in $\mathbb{R}^k$ with the $k$-means algorithm into clusters $C_1, \ldots, C_k$.

Output: Clusters $A_1, \ldots, A_k$ with $A_i = \{j \mid y_j \in C_i\}$. 
Let’s Code it!
Spatial Clustering
A cluster, which is a subset of the points of the database, satisfies two properties:

1. All points within the cluster are mutually density-connected.
2. If a point is density-connected to any point of the cluster, it is part of the cluster as well.

Check out the wikipedia article of DBSCAN. Follow the link to the OPTICS algorithm which is an improved version that can deal with density heterogeneities in space.

Visit www.hoodsquare.org to see an example of how the algorithm has been applied to discover activity hotspots in Foursquare data.
Voronoi Diagrams

http://en.wikipedia.org/wiki/Voronoi_diagram
- A powerful visualization tool
- Designed initially for artists
- Supports animations and interactivity
- Based on the Java Programming Language
The Website

Processing 2

- Download Processing
- Play With Examples
- Browse Tutorials

Processing is a programming language, development environment, and online community. Since 2001, Processing has promoted software literacy within the visual arts and visual literacy within technology. Initially created to serve as a software sketchbook and to teach computer programming fundamentals within a visual context, Processing evolved into a development tool for professionals. Today, there are tens of thousands of students, artists, designers, researchers, and hobbyists who use Processing for learning, prototyping, and production.

- Free to download and open source
- Interactive programs with 2D, 3D or PDF output
- OpenGL integration for accelerated 3D
- For GNU/Linux, Mac OS X, and Windows
- Over 100 libraries extend the core software
- Well documented, with many books available

Exhibition

Fall in Love - Phantogram
by Timothy Saccenti and Joshua Davis

Keyflies
by Miles Peyton

Petting Zoo
by Minimaforms
Explore the Examples Library

```java
// Draws a triangle using low-level OpenGL calls.
import java.nio.*;

PGL pgl;
PShader flatShader;

int vertLoc;
int colorLoc;

float[] vertices;
float[] colors;

FloatBuffer vertData;
FloatBuffer colorData;

void setup() {
    size(640, 360, P3D);

    // Loads a shader to render geometry w/out
    // textures and lights.
    flatShader = loadShader("frag.glsl", "vert.glsl");

    vertices = new float[12];
    vertData = allocateDirectFloatBuffer(12);

    colors = new float[12];
    colorData = allocateDirectFloatBuffer(12);
}

void draw() {
    background(0);

    // The geometric transformations will be automatically passed
    // to the shader.
```
Many people have created great vizs...

... and they have shared their code!
Tutorials will help you get started!


Hello Processing
by Daniel Shiffman et al.

Short video lessons introduce coding exercises that lead to designing an interactive drawing program.

Level: Beginner

Getting Started
by Casey Reas and Ben Fry

Welcome to Processing! This introduction covers the basics of writing Processing code.

Level: Beginner

Overview
by Ben Fry and Casey Reas

A little more detailed introduction to the different features of Processing than the Getting Started tutorial.

Level: Beginner

Coordinate System and Shapes
by Daniel Shiffman

Color
by Daniel Shiffman

Objects
by Daniel Shiffman
MapProvider and Tiles
How to use another map style, switch between them, and how to create your own. Also gives a short introduction to map tiles.
Java vs Python ...

```java
public class Main {
    public static void main(String[] args) {
        System.out.println("hello world");
    }
}
```

```python
print "hello world"
```

Java vs Python ... string operations

```java
public static void main(String[] args) {
    String test = "compare Java with Python";
    for(String a : test.split(" "))
        System.out.print(a);
}
```

```python
a = "compare Python with Java";
print a.split();
```

Java vs Python ... control flow

```java
int condition=10;

//if
if(condition>10)
    System.out.println("> 10");
else
    System.out.println("<= 10");

//while
while(condition>1){
    System.out.println(condition);
    condition--;
}

//switch
switch(condition){
    case 1:
    System.out.println("is 1");
    break;
    case 2:
    System.out.println("is 2");
    break;
}

//for
for(int i=0; i<10; i++){
    System.out.println(i);
}
```

```python
condition=10;

# if
if condition > 10:
    print ">10";
elif condition == 10:
    print "=10";
else:
    print "<10";

#while
while condition > 1:
    print condition;
    condition = condition-1;

#switch
def f(x):
    return {
        1 : 1,
        2 : 2,
    }[x]
print f(condition);

#for
for x in range(1,10):
    print x;
```

Java vs Python ... objects and classes

```java
class Animal{
    private String name;
    public Animal(String name){
        this.name = name;
    }
    public void saySomething(){
        System.out.println("I am " + name);
    }
}

class Dog extends Animal{
    public Dog(String name) {
        super(name);
    }
    public void saySomething(){
        System.out.println("I can bark");
    }
}

public class Main {
    public static void main(String[] args) {
        Dog dog = new Dog("Chiwawa");
        dog.saySomething();
    }
}
```

```python
class Animal():
    def __init__(self, name):
        self.name = name

    def saySomething(self):
        print "I am " + self.name

class Dog(Animal):
    def saySomething(self):
        print "I am " + self.name + ", and I can bark"

dog = Dog("Chiwawa")
dog.saySomething()
```

- Make sure you import the correct libraries.
- The setup method is run once and first.
- Here you initialize your maps and other objects.
- The `draw` method runs as a loop in the background.
- You can control the refresh rate.
- Provides the key functionality for animation and dynamics.
- You can define your own methods, classes as you would do in any programming environment.