Structural and Dynamic Analysis of Foursquare Network

Huimin Li, Yang Zhao, Ningxia Zhang (Group ID: 50)

Abstract

The soaring location-based social services not only make it possible to take advantage of one additional source of information: the places people visit, but also present graph structure consisting of individuals and places in certain relationship. In this paper we investigated the interactions between venues in Foursquare network. Provided with the check-in number for each venue as well as the transitions between them, we studied the basic characteristics of the network, detected and analyzed community structures, and modeled the graph evolution with various techniques. Our work has strong application uses and potential business values, such as using community structures to recommend venues for users based on tastes, and to predict future popularity of venues, which can be essential for business owners.

1 Introduction

The dimension of location bridges the gap between the physical world and online services, bringing social networks back to reality. Thanks to the widespread adoption of location-sensing enabled devices, location detection and sharing is embraced by almost all large social networks, and understanding location-based social networks(LBSN) empowers us to perform tasks varying from recommending interesting activities to discovering one's life pattern.

This paper examines the dataset from Foursquare, one of the largest location-based social networking websites for mobile devices. We are interested in the location-location analysis of the given dataset. In particular, we focus on three primary domains: basic graph structure analysis, community analysis and graph evolution modeling. The basic graph structure analysis presents the high-level overview of the place graph, such as distribution of different attributes and their relationships. In community analysis, we experiment multiple techniques for community detection, compare the outcomes, and attempt to interpret the community formation with map visualizations and related information of the city. Finally, given the dynamic nature of social network, we explore methods to model graph evolution in terms of the check-in number attribute for each node.

2 Previous Work

The subject of analysis in this project is the transition graph constructed from the Foursquare transition data, with venues as its nodes, and edges recording the number of transitions between venues. It will be a directed, weighted graph with attributes in its nodes and edges. Thus we have examined a number of literatures regarding weighted networks, especially in the domain of community detection. Lu et al [9] has proposed an algorithm for community detection in weighted networks, so do Jin et al. in [4]. Marrama and Low[5] have applied multiple community detection algorithms in the weighted Github network. To accomplish check-in prediction in our evolutionary analysis, we are inspired by the classic PageRank algorithm proposed by Page et al [8].

3 Data Collection and Graph Construction

For this project we used Foursquare Dataset. Every day millions of people check-in to the places they go on Foursquare and in the process create vast amounts of data about how places are connected to each other. As a sample of such "venue" interconnections for 5 major US cities, the dataset contains meta data about 160k popular public venues and 21 million anonymous check-in transitions which represents the trips between venues. There are two types of information we obtain from the dataset: (1) the detailed information about the venues, such as their geographical location, category and checkin numbers. (2) the transition between venues, with all transi-

tion time shorter than 3 hours. In addition, noticing the 3 hours transition time limit, we realize it would be hard to have a lot of edges between venues in different states, such as from San Francisco to New York. Therefore in our current analysis, we used a processed subset of the data where all the venues are in the city of San Francisco.

Insofar, we have constructed three sets of graphs: transaction graph (TG), venue graph(VG), and venue snapshot graph(VSG). TG is a simple directed graph, with venues as its nodes, and transition routes as its edges. Whenever there exists a transition log between two venues, a corresponding edge would be added into the graph. This graph is aimed to help us understand how venues and transitions are distributed in a major city (i.e. San Francisco). VG is a directed graph with node and edge attributes (i.e. TNEANet). Based on TG, node attributes have been added into VG: vid(venue id), ckn(checkin number insofar), sts(start checkin timestamp), ets(end checkin timestamp), lat(latitude), lng(longitude), category, pcategory(parent category); and also the edge attributes: trsn_cnt(transition count insofar), duration(average transition duration insofar). Because VG covers most of the information we need, it has been the graph we mainly work on. VSGs are a set of snapshot graphs of VG, which are used to analyze the evolution of VG. One-year foursquare checkin information are divided into 12 months, and for each month, a VSG is created. By looking into the dynamic change of graph characteristics, we are able to learn more about the evolution of VG.

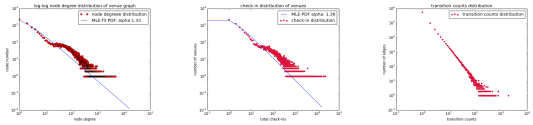
4 Methods and Analysis

4.1 Basic Structure Analysis

Given a graph, in order to get a good sense of the graph, we first conduct a series of systematic graph analysis, including distribution analysis, bow-tie analysis, etc. The algorithms we used are mostly described in class, so here redundant method descriptions are omitted. Instead, some interesting results and tentative explanations are provided.

4.1.1 Result and Analysis

- Distribution Analysis:
  We first analyzed the static Foursquare VG, and found the node distribution and checkin distribution roughly follow the power-law distribution, with $\alpha = 1.33$ (Figure 1a). However, an interesting point about these two graphs is that unlike the classical power-law distribution, there is a "plateau" in the middle (i.e. around degree 500 and 100 check-in respectively). We guess this seemingly bizarre distribution may come from the mixture of gaussian and power-law distribution. Power-law represents the real nature of node degree distribution, while the gaussian reflects the possible random check-in behavior of user.



(a) Node Degree (b) Ckin Count (c) Transition

Figure 1: Venue Graph Distribution Analysis

- Bow-tie Analysis: According to Andreil et. al.[2], graph structure in the Web often presents a bow-tie shape. A similar analysis was done to the Foursquare VG. But interestingly, the "bow-tie" we obtained seems to be an extremely skewed bow-tie shape (Table 1). Unlike the bow-tie proposed by Andreil, our skewed bow-tie has much larger out-component and largest-SCC ratio. This means, most of the venues live in a strongly-connected environment. And also there are quite a number of venues always become the "end point" of customers' moving pattern. A reasonable guess is that living community or working locations may constitute this portion of "end point".

Table 1: Table of Venue Graph bow-tie statics

NO.	Names	Value
(1)	Total Node Number	16218
(2)	Total Edge Number	771831
(3)	The largest SCC	53.10%
(4)	The in-component of the largest SCC	0.16%
(5)	The out-component of the largest SCC	46.65%
(6)	The disconnected components	0.09%

- Transition Count Analysis:
  The maximum transition count is 1935, which means the commute between that pair of nodes are extremely frequent. But most transition counts fall in between 1 and 30. This "long-tail" phenomenon causes the oscillation in our Pagerank-based graph evolution analysis, which will be covered in the next section. Again, the transition count between each pair of venue nodes also fit into one power-law distribution perfectly. But the interesting point is that the $\alpha = 4.12$(Figure 1c) is much higher than typical $2 < \alpha < 3$ range(e.g. internet graph, or human language graph). The high α means the graph has higher-degree of finite moments.

4.2 Community Analysis

Community structure has been an active research area in network analysis, as many networks of interest in natural science, social science etc. are naturally divided into communities or clusters. It'd be interesting to know whether any community structures also exist in our Foursquare transition graph, where we have venues as nodes, and transitions between venues as edges. If two venues happen to belong to the same community, it means people often go from one to another, indicating these venues are popular among people with some similarities, possibly in taste, culture, occupation or lifestyle. Investigating these community structures would not only surface interesting observations of relations between people and venues, but also provide opportunities for better recommendations of places to go, and smarter strategies of advertisements targeting at specific demographic groups. In this part, we would attempt community detection algorithms progressively and evaluate their performance.

4.2.1 Algorithms

1. **Modified Girvan-Newman Algorithm**

The first algorithm we tried is modified Girvan-Newman network, which is a variant of the most popular Girvan-Newman (GN) [7] community detection algorithm. Considering our VG graph is a directed weighted graph, we borrowed the idea from Marrama [5], to incorporate the "scaling-back" effect of the edge weight. The new betweenness formulation becomes:

$$\delta_v(v,w) = \frac{\delta_{sv}}{\delta_{sw}}(1+\sum_x \delta_s(w,x))\frac{1}{\sqrt{W(v,w)}} \qquad (1)$$

where $W(v,w)$ is the edge weight of the edge (v,w). Moreover, in order to make faster calculation of betweenness, we sacrifice the accuracy by calculating the approximate betweenness centrality instead of the above accurate one. The idea is to repeatedly sample a random vertice $v_i \in V$, and perform BFS from v_i to get the dependency score $\delta_{v_i}(e)$ of every edge e using the above modified formula. Sample until the accumulated betwenness of each edge Δ_e is greater than cn for some constant $c \geq 2$. Let the total number of samples be k, then the estimated betweenness is given by $\frac{n}{k}\Delta_e$.

2. **Clauset-Newman-Moore Algorithm**

To analyze community structures of very large networks we often see today, a group of algorithms that maximize the quantity modularity are proposed. One commonly used one among them is Clauset-Newman-Moore (CNM) algorithm, which greedily approximates the optimal modularity and proves to work substantially faster than GN algorithm ($O((m+n)n)$). It is demonstrated by Clauset et al. [1] that it's possible to handle networks with up to 400, 000 nodes and two million edges, with this algorithm and some optimization techniques.

The original CNM algorithm is a modularity-based algorithm. The modularity Q is defined as

$$Q = \sum_i (e_{ii} - a_i^2) \qquad (2)$$

where e_{ij} is the fraction of edges in the network that connect vertices in group i to those in group j, and $a_i = \sum_j e_{ij}$. Intuitively, it is the fraction of edges that fall within communities, minus the expected values of the same quantity based on random graph null model. A high value of Q generally represents a good community division. The algorithm starts with a state in which each vertex is an independent community, and we repeatedly join communities in pairs, which at each step results in the greets increase (or smallest decrease) in Q. The change in Q upon joining two communities is given by $\Delta Q = e_{ij} + eji - 2a_i a_j = 2(e_{ij} - a_i a_j)$.

Furthermore, considering the weighted nature of our VG graph, we again adapted the CNM algorithm into a weighted version. Based on the suggestion proposed by Newman, we change the e_{ij} in the modularity formula to the fraction of total edge weight in the network that connect vertices in group i to those in group j. With this slight change, we incorporate the weight factor in our analysis.

3. **Conductance-based Algorithm**

Recently, there has been research specifically on community detection algorithms that work on weighted networks. Lu et al. [9] have proposed an algorithm for weighted networks, which can also detect overlapping communities. This algorithm greedily maximizes the conductance measure, and chooses the next candidate node with maximum belonging degree.

The conductance measure of a community C is defined as:

$$\Phi(C) = \frac{cut(C, G \setminus C)}{w_c} \qquad (3)$$

If we define cut edges of community C as edges that connect nodes in the community with the rest of the network, $cut(C, G \setminus C)$ denotes the weights of the cut edges of C and w_c denotes the weights of all edges within community C including cut edges. It's obvious smaller conductance means better community detected.

The belonging degree $B(u,C)$ between node u and com-

munity C is defined as:

$$B(u,C) = \frac{\sum_{v \in C} w_{uv}}{k_u} \quad (4)$$

where w_{uv} denotes the weight of the edge between u and v and k_u is the degree of node u, which is the sum of the weights of all edges connecting node u. Apparently as belonging degree approaches to 1, node u is more likely to belong to community C.

For more details of the algorithm including the pseudo-code, please reference the paper.

4.2.2 Results and Discussions

1. **Modified Girvan-Newman Complexity Analysis**

We applied the modified GN algorithm to our test data – a subset of our transition graph of San Francisco, which consists of 8291 nodes and 87605 edges. We have reduced the graph to a weighted undirected graph. However, the algorithm couldn't output any useful data within reasonable timeframes, as Newman and Girvan algorithm is very computationally expensive. For networks with m nodes and n edges, it runs in worst case $O(m^2 n)$ time. This proves that with typical computer resources available, betweenness calculation based community detection method is not scalable, which usually can only accommodate at most a few thousands of vertices.

2. **Network Community Profile Analysis**

To make the original CNM algorithm applicable to our graph, we first reduced our VG graph to a simple undirected, unweighted graph with around 8, 000 nodes and 70, 000 edges. The algorithm terminates very fast, usually within a few minutes. Then, we also investigated the modified CNM algorithm, and experimented on the weighted VG graph. Modularity and Network Community Profile(NCP) plot are used in our evaluation.

First, we noticed that the modularity obtained by both original CNM algorithm (i.e. around 2.7) and our modified CNM algorithm (i.e. around 0.6) is surprisingly low. According to Newman, only communities with modularity higher than 0.3 can be viewed as significant community structure. However, as we will show later in the visualization part, communities detected in our data can be interpreted with some other information about San Francisco. We believe this is because the modularity proposed by Newman may not be a suitable objective function in our case. In other words, the null model in the modularity formula may not be a good metric, and therefore the "magic" number 0.3 means little in our case.

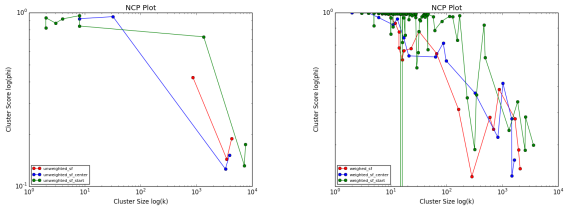
Second, we used NCP plot to compare the modified and original CNM algorithm (Fiugure 2). The y-axis of NCP plot is the NCP score, defined as

$$\Phi(k) = \min_{S \subset V, |S|=k} \phi(S) \quad (5)$$

where $\phi(S) = \frac{|\{(i,j) \in E; i \in S, j \notin S\}|}{\sum_{s} \in S d_s}$. From Figure 2, we can see original NCM tentatively divide the graph into few large communities, which may result in the loss of details inside each large community. Also, the conductance drops monotonously along the increase of cluster size, which is an unattractive trivial result, since too large communities incorporate most of the edges inside it, and only have few edges stretching out.

However, unlike the original CNM algorithm, our modified CNM algorithm yields an interesting 'V-shape' result, which echoes the discovery mentioned by Leskovec et. al. [6]. We have applied the unweighted and weighted versions of CNM to three graphs: transition graph of downtown San Francisco (transition starts and ends within a grid area centered at latitude and longitude (37.76956, -122.42280), radius 0.04), transition graph within San Francisco (transition starts and ends within the city) and transition graph starting from San Francisco (transition starts in San Francisco and ends elsewhere). These three graphs are represented by red, blue, and green lines, respectively in Figure 2b. All three curves show the similar "V-shape" pattern, indicating the size of 400 nodes per community achieves the best cluster score.

Another interesting observation is that in the Figure 2b modified CNM, there is a big drop around cluster score the size of 15. In fact, the cluster score is 0. After more thorough investigation, we found this is an isolated community of 15 nodes that have no outgoing edges.



(a) CNM (b) Modified CNM

Figure 2: Network Community Profile

3. **Visualization and Interpretation**

To interpret the community detection results, we have proposed three hypotheses: 1) Venues of the same community cluster geographically; 2) Patrons of venues

within one community share ethnical or racial similarities; 3) Different communities have different category distributions.

To test the first two hypotheses, we visualized the venues on the map, and color coded different detected communities. Note that the communities here represent a cluster of venues that people often check-in sequentially. In Figure 3, we can clearly see that venues of the same community tend to cluster geographically, e.g. there's a dense cluster around the downtown area. This observation matches the intuition that people usually check in at one place, and then check in at a nearby place. Meanwhile, note that the biggest community, labeled red here, is scattered around the downtown and spreads throughout the sunset area. We'll explain this observation with our second hypothesis.

To test the second hypothesis, we have compared our community map with the racial segregation map (Figure 4), created by Dustin Cable at University of Virginia's Weldon Cooper Center for Public Service with data from 2010 U.S. Census and found them surprisingly similar to each other. The community labeled red corresponds to the Asian community, while the community labeled orange corresponds to the Latino / Hispanic community. This explains that people from certain cultures tend to be active around certain areas in San Francisco.

If we compare the community map obtained via unweighted CNM algorithm and that via weighted CNM algorithm, we can see that the latter has more, smaller communities, i.e. more detailed community structure. This observation confirms our initial intuition that utilizing edge weights yields more sophisticated results. Most notably, the downtown community has more internal structure in the weighted version, which cannot be explained away just by racial segregation.

To inspect the category distribution within communities, we found out the top 20 sub-categories that have the most occurrences in the community. Partial results of the largest communities in the weighted version can be found in table ref. The high number of Asian restaurants in the red community, and Mexican restaurants in the orange community again confirm our racial segregation hypothesis. Moreover, the light blue community represents the prominent tech startup scene and the financial area, the pink community has the theme of shopping and leisure, the green community features hotels and event venues around Union Square and finally the purple community represents the touristy Fisherman's Wharf area.

It's interesting that such rich dynamics and diversity of a city can be inferred purely from these digital traces people left everyday.

(a) CNM (b) Modified CNM

Figure 3: SF Venue Graph Community Visualization

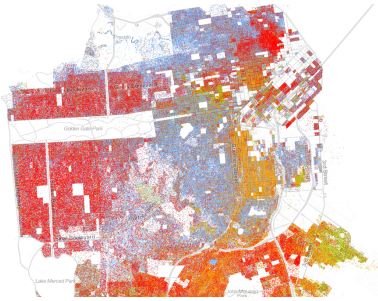


Figure 4: SF Racial Segregation

4. **Overlapping Community Detection**

The conductance-based community detection algorithm turns out to have a high time complexity, especially when one community gets really large. Thus we haven't got the complete result of our transition graph of San Francisco, but we have some partial results that already show some potentials of the algorithm. It has detected the Latino / Hispanic community, and several smaller overlapping ones along the northeastern coast, with conductance measure around 0.5. Detecting overlapping communities and yielding finer structures promise to be the advantages of this algorithm.

4.3 Graph Evolution Analysis

Graph evolution is an interesting problem in network analysis, and once modeled correctly, can be a powerful tool with business value. In particular, Foursquare data

Community color	Top 10 sub-categories
Red	Chinese Restaurants: 70, Coffee Shops: 66, Sushi Restaurants: 61, Bars: 58, Cafs: 56, Japanese Restaurants: 51, Grocery Stores: 48, Parks: 44, Pizza Places: 41, Vietnamese Restaurants: 38
Orange	Bars: 66, Coffee Shops: 65, Cafs: 54, Mexican Restaurants: 52, Sandwich Places: 42, American Restaurants: 40, Art Galleries: 40, Grocery Stores: 37, Sushi Restaurants: 36, Pizza Places: 35
Light Blue	Offices: 137, Coffee Shops: 72, Sandwich Places: 67, Chinese Restaurants: 57, Tech Startups: 52, Food Trucks: 48, Cafs: 34, American Restaurants: 32, Banks: 32, Bakeries: 28
Pink	Coffee Shops: 26, Parks: 23, Cafs: 22, Clothing Stores: 21, Boutiques: 18, Bars: 17, Pizza Places: 14, Light Rails: 13, Sushi Restaurants: 13, Breakfast Spots: 12
Green	Hotels: 57, Clothing Stores: 42, Coffee Shops: 26, Cafs: 21, Thai Restaurants: 20, American Restaurants: 17, Pizza Places: 17, Bars: 16, Sandwich Places: 16, Event Spaces: 16
Purple	Italian Restaurants: 39, Seafood Restaurants: 28, Hotels: 24, Boats or Ferries: 22, Bars: 20, Gift Shops: 19, Harbors / Marinas: 15, Pizza Places: 13, Parks: 13, Cafs: 12

Table 2: Top 10 Sub-category for Each community

can be a precious asset for business owners as it helps them to understand how popular their places have been in the past, however, is there a way to learn about the future given all the information at presented In this section, we present two approaches to model the evolution of popularity of venues and to predict future popularity by using check-in numbers for each venue as a proxy.

At a high level overview, we form our problem as following: at any point of time, given a subset of venues(e.g. venues in a community, or even venues in a city), the check-in numbers for each of them, and the transitions between them so far, what would be the number of new check-ins for each venue at a particular future time? In other words, given a snapshot of the graph(with check-in numbers as node attribute and transition frequency between nodes as edge attribute) and the total number of new check-in numbers that will be added to the graph as a whole, the goal is to predict the snapshot of the entire graph at a future time.

We experienced two different approaches for modeling: the first one is based on previous check-in number, and the other one is based on PageRank [8].

4.3.1 Algorithms

1. **Distribution-based Approach**

This approach is based on the distribution of historical data(check-in number of each venue), as it assigns the new check-ins to venues according to such distribution. The detailed algorithm is defined as following: suppose at time t, node v has check-in number c_v^t, and we know between t and $t+1$, we have M new check-ins, then

$$c_v^{(t+1)} = M \cdot \frac{c_v^t}{\sum_v c_v^t} \tag{6}$$

The intuition is that venues are likely to continue their past performance and therefore the graph will maintain similar popularity structure. As we will see later, this straightforward method produces relatively good prediction for the whole graph in general, yet it is limited in nature since it cannot model potential changes to graph structure.

2. **PageRank-based Approach**

Notice that the graph has this "flow" structure between nodes(i.e. the transition information between venues), by intuition this fits well in the PageRank model - only now users instead of jump from one website to another in browser, travel from one place to another in real life. Like the Internet graph, in Foursquare place graph each node(i.e. venue) has a number of "in-edges"(i.e. transitions from other venues) and a number of "out-edges"(i.e. transitions to other venues), and intuitively nodes with larger "in-degree" are likely to get more check-ins whereas nodes with smaller "in-degree" are likely to receive fewer.

Define the total new check-in number as M, the number of nodes as N, the value of node i (i.e. the proposed check-in number for node i) as r_i, the transition frequency between node i and j (i.e. weight of the directed edge) as f_{ij}, we redistribute the value of r_i to all the nodes it can transit into, based on the frequency distribution among all destinations. Implementation wise, given the large number to nodes, we used Power Iteration, which is more plausible than solving a large dimension matrix.

```
Set r_j = M / N for each node j
1. for each j:
```

$$r'_j = \beta \sum_{i \to j} r_j \cdot \frac{f_{ij}}{\sum_k f_{ik}} + (1-\beta)\frac{M}{N}$$

```
2.  r = r'
3.  if meet exit criteria: return 1;
    otherwise go to 1 and repeat the process
```

We started off with the standard PageRank(with teleporting, see pseudo-code), and made a series of modification and improvement to reflect characteristics of this graph.

3. **Modified PageRank with teleporting**

The above algorithm assumes that the transition graph has no dead-ends that do not have out-going edges. Unfortunately this is not the case in Foursquare dataset - in fact, analysis shows that on average up to $\frac{1}{3}$ of nodes are dead-ends. To handle this, we added in logic to explicitly follow random teleport links with probability 1.0 for dead-ends[3]. Furthermore, we ran an initial experiment with different value and found it models the graph best when = 1. Our modified step 1 of above algorithm is as below.

$$r'_j = \sum_{i \to j} r_j \cdot \frac{f_{ij}}{(\sum_k f_{ik})} + \sum_{deadend\ i} \frac{r_i}{N} \tag{7}$$

4. **PageRank with Enhanced Graph**

We use the transition graph with the total number of transitions from one venue to another, up to a given time, as our directed edge weight. Yet due to its time sensitive nature, in real life recent check-ins(thus recent transitions) can be more valuable information than those happened earlier. The intuition is that the popularity of venues is likely to follow the "trend" from last time period. Inspired by this, we decide to use the

enhanced graph, where the edge puts more weight into transitions in most recent time period(i.e. one month). Formally, define new transition graph G' at time t, and the number of transitions happened from node i to j between time $t-2$ and $t-1$ as $transition_{ij}^{t-1}$, then the edge weight $f_{ij}^{'t}$ for edge (ij) is updated as

$$f_{ij}^{'t} \leftarrow f_{ij}^{t-1} + \alpha \cdot transition_{ij}^{t-1} \quad (8)$$

Where α is a parameter we can adjust based on by how much we want to favor recent transitions. Our experiment shows the enhanced graph performs relatively good when we use $\alpha = 2$.

5. Add Distribution Information

So far our PageRank algorithm has only used information of transitions between venues within this subset. One key observation to make is that the real check-in number of a venue can be larger than the number of transitions in our transition graph snapshot, since a place can get visits that come from another place out of our subset selections, or get visits without transition from other places. To capture this nature, we propose a modified PageRank score distribution, where we not only use the transition frequency of an edge to determine how to distribute the score, but also take the previous check-in number of the destination node into consideration.

$$r'_j = \sum_{i \to j} r_j \cdot \frac{(\lambda + dist_j) \cdot f'_{ij}}{\sum_k ((\lambda + dist_k) \cdot f'_{ik})} + \sum_{deadend\ i} \frac{r_i}{N} \quad (9)$$

Here λ is the smoothing constant to prevent the score distribution from being distorted too much. Intuitively, if a node has higher check-in numbers at the snapshot time, this node is more likely is receive visits in our PageRank algorithm.

6. Learning from Last Prediction

In this step, we incorporate learning in our algorithm by adjusting the PageRank score distribution dynamically based on previous prediction results. In particular, after we perform a prediction, we compare our prediction result to the real data, and calculate learning result L_j for each node j. Initially we use $L_j = \frac{real\_value\_j}{prediciton_j}$, however it reveals such calculation will adjust to the opposite direction too much, and therefore we propose a milder way of calculation by setting L_j to 1.1 if $real\_value\_j$ is larger than $prediction_j$, and setting L_j to 0.9 if $real\_value\_j$ is smaller than $prediction_j$. Then in the next round of prediction, we reuse the learning result in PageRank.

$$r'_j = \sum_{i \to j} r_j \cdot \frac{(\lambda + dist_j) \cdot f_{ij} \cdot L_j}{\sum_k ((\lambda + dist_k) f_{ik} \cdot L_k)} + \sum_{deadend\ i} \frac{r_i}{N} \quad (10)$$

Therefore, if our model underestimate a node's check-in number in previous prediction, the algorithm adjusts itself to give this node more score in the next round.

4.3.2 Result and Discussion

We evaluate our models by using monthly snapshots (from month 0 to month 11) of the graph. Specifically, as we use the first snapshot as base graph for prediction, and the last snapshot only contains partial data, we apply different models to predict check-in numbers for month 1 to 10 using previous month's snapshot, and compare our prediction with the real graph. We are interested in learning which model yields a closer outcome to the real one, with the cost function defined as below:

$$F(s, s') = \sum_v \frac{(r^{s_v} - r^{s'_v})^2}{M} \quad (11)$$

The result measured by the above cost function is shown in Figure 5. It turns out the distribution-based approach gives the best performance for the overall graph, yet, the PageRank algorithm indeed achieves better results as we add in modifications gradually.

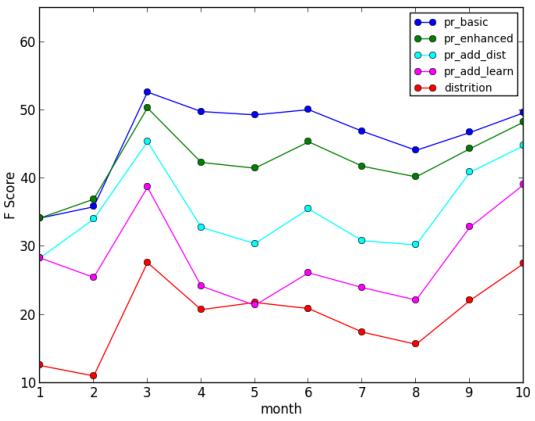


Figure 5: Error Score for Monthly Prediction

Nevertheless, both approaches appear to capture the basic pattern for next time stamp successfully - as we can see from the example in Figure 6, the predicted check-in number of each venue follows the trend of the real data in general. One interesting observation is that the distribution-based approach tends to have larger difference from correct prediction for venues with high visits

Table 3: F Score (number of successful prediction) for 10 nodes with Highest New Check-ins

Method / Month.	1	2	3	4	5	6	7	8	9	10
PR-M.	127.7(5)	141.7(6)	349.8(5)	133.2(6)	34.7(8)	91.6(7)	80.3(7)	20.1(9)	236.7(7)	271.2
Dist.	85.0(8)	54.2(8)	320.3(6)	198.5(5)	135.1(6)	138.2(5)	114.8(5)	70.8(6)	228.5(6)	296.6(5)

(i.e. nodes with high check-in numbers), whereas the modified PageRank approach has larger variance at the tail.

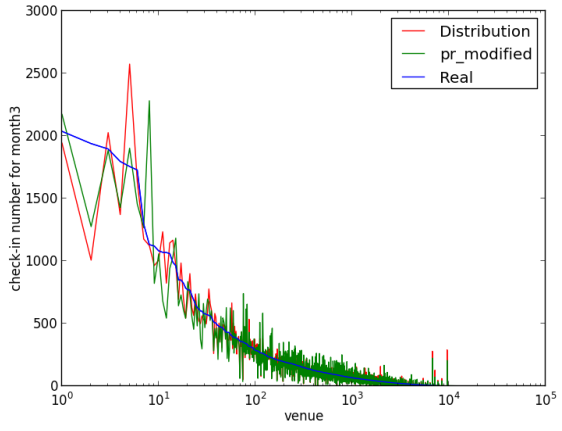


Figure 6: Prediction for distribution-based and modified pagerank approaches

Inspired by such observation, we further investigated prediction for high visit nodes. We found that modified PageRank actually performs better (especially on later predictions due to the learning over steps) on those particularly popular nodes both in terms of F score and the number of nodes we predict "correctly" - we define a node to be predicted correctly if the absolute difference between our prediction and real value is within 20% of the real value (Figure 7 and Table 3). Here we expected a higher F score on average since now we only consider those 10 nodes, each of which has more than 1000 new check-ins. Also the sequential graphs in Figure 7 shows the improvement in PageRank algorithm over time steps.

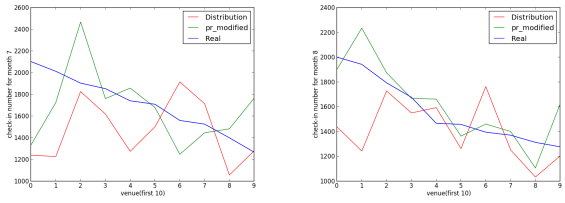


Figure 7: Prediction Comparison for 10 nodes with Highest New Check-ins

5 Conclusion and Future Work

As shown above, we have successfully adapted the CNM algorithm to accommodate large-scale weighted networks, and used it to uncover community structures in our transition graph. By means of visualization and category analysis, we have shown that applying community detection on networks of venues, can surface interesting dynamics and diversity features of a city. Our attempt with the conductance-based algorithm has shown potentials of efficient algorithms for overlapping community detection in weighted networks.

To advance our work, one of the directions is to find a better null model to measure modularity of networks like our transition graph, which is different from the usual social network. The detected community structures can also be valuable in various location-based applications, such as friend / activity recommendation, overview of the city for tourists, advertising strategy optimization etc. As the conductance-based algorithm suffers from complexity problems, there might be ways to optimize the algorithm to make it more practical in the real world.

To model graph evolution in terms of check-in number, we present two approaches: one using previous check-in number distribution, the other using PageRank algorithm with various modifications. Our result demonstrates that both approaches mimic the behavior of check-in number evolution in the future graph. In particular, straightforward distribution-based method gives lower error value in terms of the entire graph, whereas the modified PageRank algorithm is better at predicting future check-in for venues with large traffic(i.e. nodes with high check-in number attribute).

One of the future improvement is to use "personalized" PageRank by taking the other attributes of a venue into consideration, such as category and community information, and therefore defining a "teleport" set for each node where they have large chance to transit to such set. Other directions include to use a combined approach of the two algorithms, with the assumption that the future check-in is partially from keep receiving similar portion of new check-ins(for the visits that cannot be captured by transitioning), and meanwhile partially dependent on how it gets visits by transitioning from other venues.

References

[1] AARON CLAUSET, M. E. J. NEWMAN, C. M. Finding community structure in very large networks. *Phys. Rev. E* (2004).

[2] ANDREI BRODER, RAVI KUMAR, F. M. Graph structure in the web. *Computer Networks* (2000), 309–320.

[3] BERKHIN, P. *A Survey on PageRank Computing*. Internet Mathematics, 2005.

[4] JIE JIN, L. P. *A Center-Based Community Detection Method in Weighted Networks*. IEEE International Conference on, 2011.

[5] JOSEPH MARRAMA, T. L. Social coding: Evaluating github's network using weighted community detection. *CS224W Project Report* (2012).

[6] JURE LESKOVEC, K. J. L. Statistical properties of community structure in large social and information networks. *Social Networks and Web 2.0* (2008).

[7] M. GIRVAN, M. E. J. N. Community structure in social and biological networks. *PNAS 99* (2002).

[8] PAGE, L. The pagerank citation ranking: Bringing order to the web. *Technical Report* (1999).

[9] ZONGQING LU, Y. W. *Community Detection in Weighted Networks: Algorithms and Applications]*.