DISCOVERING  RIENDSHIPS

Drew Wicke
Mihail Sharov

1. **Problem Description**

The goal of this project is to discover friendships among Facebook users based on features from their profiles. This clustering could be used for suggesting friends to a new user (this is an existing feature of Facebook which does not necessarily use the same method). It could also be used by third parties for advertising – they could improve the effect and reduce the cost of their advertisement by focusing on one representative from each cluster, relying on the users' tendency to share information with their friends. That way, the ad would reach more people with fewer resources. Discovering Facebook friends is a pattern recognition problem because it requires an unsupervised method for determining groups of similar users. The main challenge in this project is that we don't know for sure if it's even possible to discover the friendship circles based on the given features. Even if two users have a perfect correlation of some features, such as name or birthday, that doesn't necessarily imply that they are friends. Some other features, such as school and workplace, might be better indicators of whether two users are friends, but even they don't guarantee anything. Another major challenge is that the values of the features are binary, which limits our choices of clustering algorithms and may require us to modify them significantly. The sparseness of the data presents yet another challenge. There are a total of 224 features and 347 users. The average number of positive indicators (features with value of 1) per user is 9.56, which is only 4.27% of the available features. This is because most of the profile information on Facebook is optional. In our attempts to overcome these challenges and discover the friendship circles, we explored the performance of the following clustering algorithms: K-means, DBSCAN, SOM, and Spectral Clustering. The data that we used can be found here [2]. It contains 10 networks of users with multiple circles of friends within each network (some circles contain as few as 1 user, which is an additional challenge). The friendship circles are already known in the dataset but we ignored this information during training so that our learning can be completely unsupervised. We used the information about the actual friendship circles only for testing the validity of the clusters formed. We only focused on one of the ten networks provided, and attempted to discover the friendship circles within it. We used a combination of Matlab and R for this project with some C++. The performance of the algorithms was measured using the true friendship circles by calculating the percentage of correct group assignments.

2. Related Work

A nice introduction to clustering with graph data was introduced in [3]. Leskovec in [5] used the same Facebook data however they used both the feature data and the network data in order to identify the clusters. Mishra in [6] cluster social networks with the same network assumptions however they looked at overlapping clusters and provided an algorithm that provably could identify the clusters. Large graphs continued to be explored in terms of

identifying clusters in [10].

Other approaches to identifying clusters considered communities [7]. A density based approach to identifying community based clusters in social networks called Dengraph was implemented in [13]. Girvan and Newman considered both social and biological network communities in [15]. This work considered communities with dense connections between those in a community and sparse connections to those outside the community. Clauset and Newman continued the work of identifying community structure, but this time in large networks [16]. Rosval and Bergstrom created the Infomap algorithm to reveal community structure in scientific communications [17]. In [11] they found community based structures in 230 large real world social networks while defining a ground truth for their communities in order to evaluate their results. They also modified spectral clustering in order to identify community structures. [12] takes a different approach, rather than using a parameter free model they project the factors into a so called "social space" in order to cluster the nodes.

People have also focused on creating social networks or exploring how they evolve. Like in [4], who considers the Multiplicative Attribute Graph (MAG) where graphs were generated based off randomly generated features and probabilistic connections based off of those features. Another interesting work explored the evolution of four large social networks [8]. Also, email communication from the Enron corpus was considered under a social network perspective and the social dynamics to identify the underlying patterns in an organization's failure [14].

However, to our knowledge, clustering social networking data without using the underlying social network has not been explored. Interestingly, this seems like an important problem and the normal approach is to use the social network. There have been cases where social network extraction from a database was considered in [18].

3. Data Description and Examples

The data we will use in order to find friendships between users consists of 347 users and 224 anonymous binary features. By "anonymous" we mean that we don't have a clear understanding of what the features mean or how they were extracted. For example, the first 8 features have the following names:

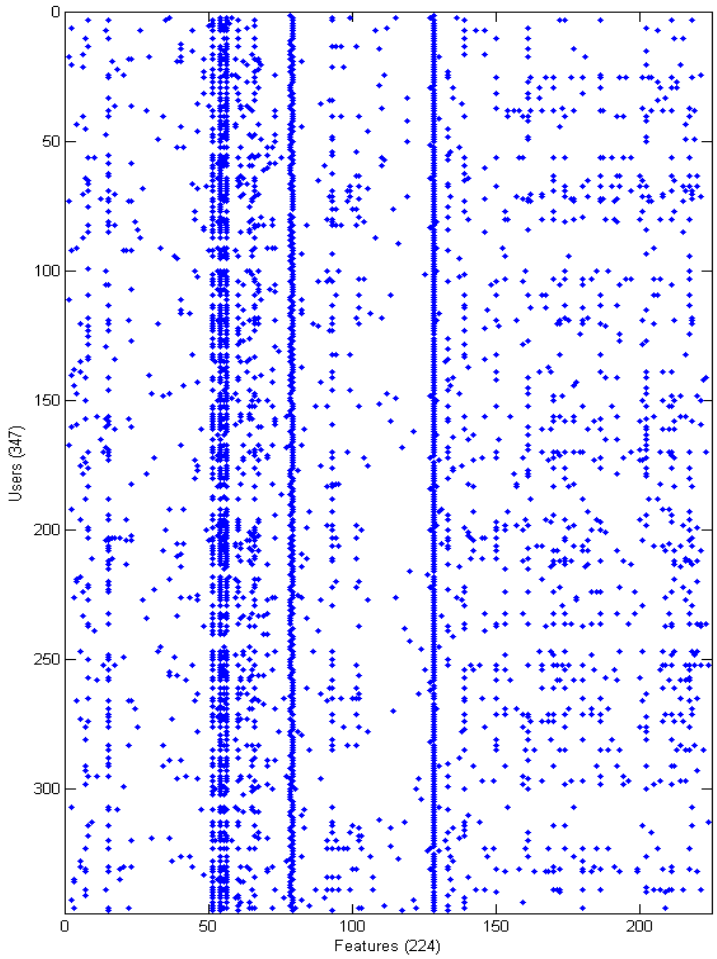
```
0 birthday;anonymized feature 0
1 birthday;anonymized feature 1
2 birthday;anonymized feature 2
3 birthday;anonymized feature 3
4 birthday;anonymized feature 4
5 birthday;anonymized feature 5
6 birthday;anonymized feature 6
7 birthday;anonymized feature 7
```

We know that these 8 features relate to the user's birthday, but we do not know the relationship between the user's day of birth and the binary values of these attributes. Therefore, we have only the binary patterns to work with. We are given various numbers of features for different categories (such as classes taken, degree acquired,

etc.). This is a complete list of the feature categories in this dataset:

```
0 birthday;anonymized
8 education;classes;id;anonymized
13 education;concentration;id;anonymized
20 education;degree;id;anonymized
24 education;school;id;anonymized
53 education;type;anonymized
57 education;year;id;anonymized
73 first_name;anonymized
77 gender;anonymized
79 hometown;id;anonymized
90 languages;id;anonymized
104 last_name;anonymized
125 locale;anonymized
128 location;id;anonymized
140 work;employer;id;anonymized
160 work;end_date;anonymized
176 work;location;id;anonymized
201 work;start_date;anonymized
```

Using Matlab's function "spy" we can get a good idea of what the data looks like. The x-axis is the feature ID, the y-axis is the user ID, and the dots correspond to value of 1 (no dot means 0).



4. Performance Evaluation

Let be the cluster that a method discovers and suppose the number of clusters is . Let be the actual circle of friends and suppose the number of circles is . For every cluster , we iterate through each of the actual circles and count the number of users from that circle that are also in the cluster:

We would like to know how far off our cluster is from an exact match with the actual circle. Because a discovered cluster may have either less or more people than the actual circle, we divide the number of users above by the cardinality of the larger set to get a measure of the similarity between cluster and circle :

We define the accuracy of a cluster as the maximum similarity with any of the actual

circles. Since there is no correspondence between the actual circles and the clusters that we discover, we match the discovered cluster with the circle that it resembles most:

Finally, we average those accuracies for all clusters that were discovered and we use that number to represent the success rate of the clustering method:

5. Preprocessing and Data Conversion

There were no missing values in this dataset and no data scaling needed to be done, since all the features are binary. However, since most clustering algorithms have been designed for numeric data, we feared that they wouldn't perform well on this binary data. We came to the realization that, even though PCA is usually used for dimensionality reduction, it can be used to transform our binary features into numeric (by taking linear combinations of the binary features). This way, we created a new dataset with 224 numeric features. From it, we created a 3rd dataset by taking only those principal components (as computed by the PCA) for which the variance is at least 0.01 (there were 100 such features). We tried using these datasets as well in hopes that the clustering methods will perform better on the numeric data.

Another way we converted the feature data was to create an adjacency matrix. The entries in our matrix correspond with the number of positive feature entries that the row user had in common with each of the column users. This is an approximation of a friendship network, where the connected vertices are friends. This graph is almost fully connected due to some of the features being common among most of the users. Therefore, the weight is useful since it characterizes how likely connected vertices are friends.

6. Clustering Methods

6.1. K-Means

In order to get some initial results and insights of the data, we performed the K-means algorithm with different values of K. The lowest number of K we used is 2 and the highest is 30. The reason behind this upper boundary is that the number of actual circles is 24, and we expect that any attempt to find significantly more than 24 clusters will result in low success rate. The method was applied to the three datasets

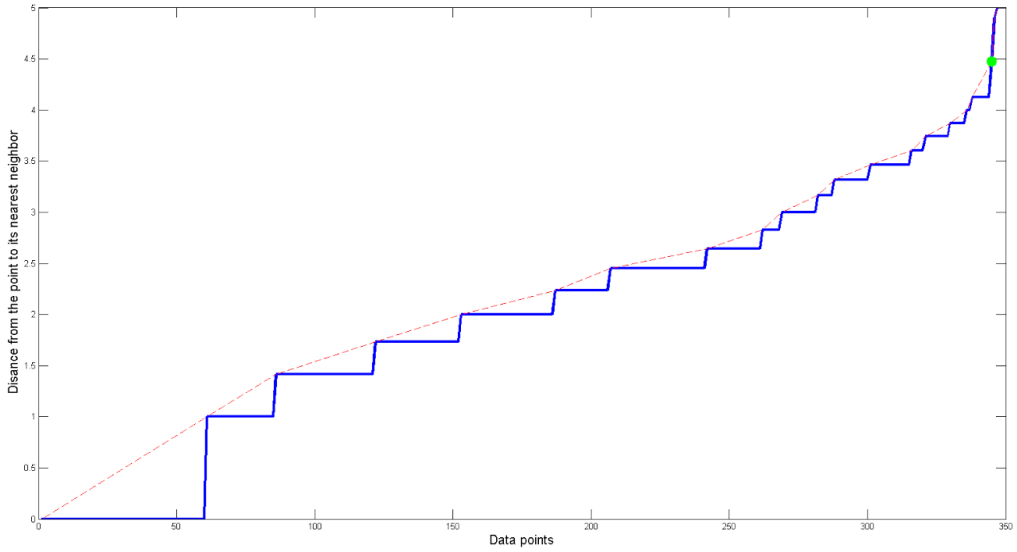
mentioned above, and the success rates are shown in the table below. In many cases, the conversion to numeric features improved the success rate, and the removal of the less significant principal components improved it even further.

K	Original Data	After PCA	100 Principal Comp's
2	0.4028	0.4028	0.4028
3	0.3066	0.3172	0.3220
4	0.2627	0.2617	0.2500
5	0.2084	0.2003	0.2365
6	0.1739	0.2104	0.1831
7	0.1809	0.1814	0.1928
8	0.1864	0.1880	0.1878
9	0.1636	0.2040	0.1641
10	0.1777	0.1719	0.1708
11	0.1533	0.1688	0.1808
12	0.1603	0.1640	0.1725
13	0.1546	0.1481	0.1706
14	0.1340	0.1551	0.1524
15	0.1493	0.1454	0.1452
16	0.1459	0.1327	0.1552
17	0.1635	0.1572	0.1245
18	0.1384	0.1416	0.1416
19	0.1371	0.1265	0.1353
20	0.1252	0.1443	0.1249
21	0.1195	0.1221	0.1410
22	0.1307	0.1221	0.1378
23	0.1264	0.1318	0.1175
24	0.1224	0.1171	0.1262
25	0.1293	0.1208	0.1197

26	0.1213	0.1410	0.1324
27	0.1198	0.1117	0.1321
28	0.1169	0.1080	0.1212
29	0.1309	0.1208	0.1167
30	0.1158	0.1200	0.1233

6.2. DBSCAN

This method requires that we specify two parameters: minimum number of points required to form a cluster and a distance threshold. The minimum number of points used here is 2 because a “circle” of friends could consist of just 2 friends. As we know from the actual circles, some of them contain just 1 user, but we discovered that this is not a good choice for the minimum number of points required to form a cluster using DBSCAN because most points ended up in clusters by themselves. In order to figure out the distance threshold, we plotted the sorted distances from each point to its nearest neighbor and found the largest change in distance (largest slope). The plot below shows the distances sorted in increasing order. The red dotted lines mark the changes in distance and the green dot marks the point of largest change. It occurs at the distance value of 4.47.



However, with this distance threshold, DBSCAN found only 1 cluster with 344 points, and disregarded the other 3 as noise. In order to be able to split the users into

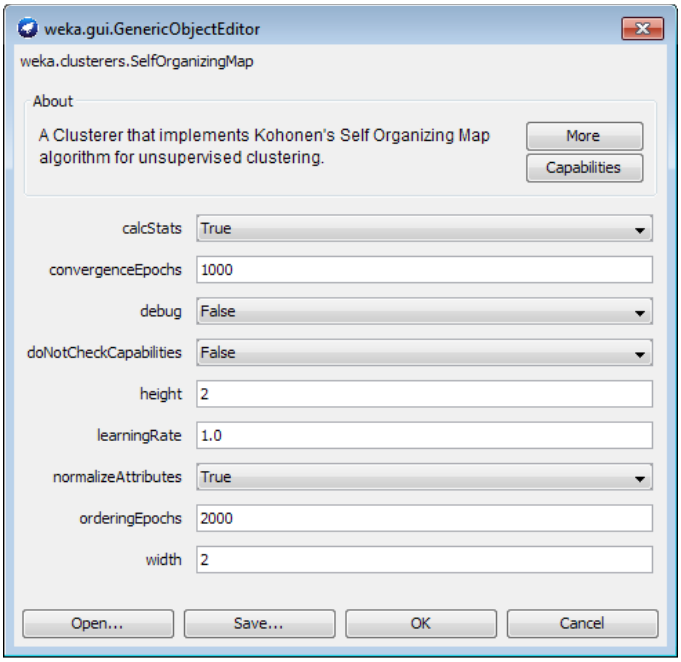
multiple clusters, we incrementally lowered this threshold and recorded the results:

Threshold	Cluster: # of members	Noisy points	Success rate
4.4	1: 344	3	0.3808
4.3	1: 344	3	0.3808
4.2	1: 344	3	0.3808
4.1	1: 337	10	0.3769
4.0	1: 335	12	0.3761
3.9	1: 335	12	0.3761
3.8	1: 329	18	0.3647
3.7	1: 320	27	0.3593
3.6	1: 313, 2: 2	32	0.2377
3.5	1: 313, 2: 2	32	0.2377
3.4	1: 300	47	0.3600
3.3	1: 287	60	0.3554
3.2	1: 287	60	0.3554
3.1	1: 281	66	0.3594
3.0	1: 268	79	0.3433
2.9	1: 268	79	0.3433
2.8	1: 261	86	0.3371
2.7	1: 261	86	0.3371
2.6	1: 238, 2: 3	106	0.1972
2.5	1: 238, 2: 3	106	0.1972

These results suggest that, instead of finding more clusters, DBSCAN maintains one cluster and labels the points that don't fall into that cluster as noise. The results obtained from the PCA-transformed dataset have the same nature. Thus, we conclude that this method will not be useful for clustering the users into friendship circles in this case.

6.3. Self-Organizing Maps

We also used Weka's SOM tool to discover clusters in the data. We used Weka's default parameters shown below, and we only changed the width and height in order to achieve different number of clusters. The table below shows the results we obtained.



This split of members seems reasonable, since we expect the different circles to have different number of members in them. For this reason, we wanted to find out what the results would be if we assume that we know how many clusters there are and used this information for the SOM. In a real-life scenario where the number of clusters is unknown, one can compute SOM with dimensions 1xN for an incrementally increasing integer N and then select the clustering that seems best according to some problem-specific evaluation function. Then, if the N chosen is not prime, one can compute 2D SOM's of different dimensions that would result in N clusters and decide which configuration is most desirable.

The table below summarizes our findings for SOM's of different sizes. The assumption that we know the number of actual circles (which is 24) is accounted for in the last 4 rows of the table in the manner described above.

Width X Height	# of clusters: (# of members in each)	Success rate
2 X 2	**4**: (103, 81, 95, 68)	0.25
3 X 3	**9**: (51, 28, 49, 4, 9, 8, 57, 82, 59)	0.1650
4 X 4	**16**: (27, 18, 7, 43, 21, 12, 3, 48, 21, 20, 20, 9, 25, 21, 16, 36)	0.1670
5 X 5	**25**: (19, 9, 9, 21, 26, 9, 8, 7, 11, 8, 21, 3, 3, 3, 40, 12, 5, 12, 3, 1, 18, 16, 25, 7, 51)	0.1488
6 X 6	**36**: (18, 7, 40, 1, 3, 46, 8, 3, 2, 2, 4, 5, 13, 13, 7, 3, 1, 9, 9, 4, 2, 16, 4, 20, 11, 3, 4, 5, 4, 12, 17, 9, 11, 14, 7, 10)	0.1234
24 X 1	**24:** (26, 12, 15, 13, 10, 4, 40, 1, 0, 47, 7, 9, 13, 21, 7, 19, 10, 16, 10, 10, 13, 9, 13, 22)	0.1374
12 X 2	**24**: (4, 1, 40, 4, 9, 31, 5, 11, 16, 15, 13, 14, 55, 0, 0, 8, 16, 11, 5, 32, 11, 12, 8, 26)	0.1216
8 X 3	**24**: (18, 14, 12, 29, 16, 15, 9, 42, 6, 5, 5, 4, 0, 3, 0, 1, 25, 13, 13, 15, 24, 18, 8, 52)	0.1295
6 X 4	**24:** (20, 9, 11, 31, 13, 40, 10, 13, 2, 13, 10, 0, 13, 6, 16, 8, 3, 1, 15, 19, 13, 22, 8, 51)	0.1543

As with K-means, the success rates seem better with less clusters. For example, the highest success rate was achieved by only 4 clusters. However, we know that this is far from the actual number of circles. It is interesting to note that SOM was unable to cluster the data into exactly 24 clusters because it always left at least one cluster with 0 members. It is also interesting to note that the success rate from the last 4 rows is highest at the 2 extremes – 24x1 and 6x4. On a 2D SOM these correspond to having the clusters arranged in one line and having the clusters arranged as close to a square as possible

4. Spectral Clustering

We chose this method due to its ability to cluster based on an adjacency matrix

representation of a graph. There are numerous methods for clustering graph data as discussed in [3]. However, due to the binary feature data the spectral clustering method seemed appropriate.

Using the full dataset converted to an adjacency matrix we performed spectral clustering and we obtained the success rate results for 2-30 clusters. With the spectral clustering algorithm we used we had two main knobs we could turn to improve the results: the method used to normalize the Laplacian and the number of neighbors to consider. The number of neighbors is used to construct the Laplacian using KNN. The Laplacian is then normalized using either the symmetric or the random walk methods. Based on the table below (method, number of neighbors) we notice that 12 neighbors maintains a higher success rate than the 25 neighbor version.

K	Symmetric, 12	Symmetric, 25	Random Walk, 12	Random Walk, 25
2	0.2458134	0.2525754	0.2415414	0.2455935
3	0.205372	0.2076393	0.1977527	0.2053203
4	0.1676628	0.2330068	0.1747765	0.2320836
5	0.1891448	0.2381408	0.1930008	0.2546226
6	0.208699	0.2186421	0.2247513	0.2342072
7	0.2027399	0.1947447	0.2062245	0.2110732
8	0.1914115	0.1896332	0.1888995	0.2096907
9	0.1949915	0.2060654	0.2089683	0.2269152
10	0.2007505	0.1970864	0.2210345	0.2035016
11	0.1889329	0.2111974	0.2084384	0.2111648
12	0.1913919	0.1994957	0.198643	0.1983819
13	0.1948989	0.1906025	0.1912602	0.1829898
14	0.1945231	0.1755856	0.1884097	0.1751954
15	0.205361	0.1839696	0.1946328	0.1894405
16	0.1916421	0.1834201	0.1951262	0.1932485
17	0.1943766	0.1941003	0.1916726	0.1829625
18	0.1948546	0.1813245	0.2006888	0.1743342
19	0.1761081	0.1779862	0.1797173	0.1826446

20	0.1871167	0.1759906	0.178602	0.1756145
21	0.1775598	0.1686443	0.1856293	0.1793395
22	0.1775594	0.1717379	0.1796933	0.17077
23	0.1655297	0.1659233	0.1702554	0.1699729
24	0.1702592	0.1619185	0.1693297	0.1543643
25	0.1698778	0.1617105	0.169766	0.1661586
26	0.1607767	0.1600171	0.1626444	0.1707694
27	0.1511264	0.1611408	0.1659894	0.1542603
28	0.1503132	0.1496132	0.1592523	0.1627561
29	0.1526648	0.143708	0.1534626	0.1471121
30	0.156183	0.1514456	0.149116	0.1500463

The success rates for mid-size number of clusters appears to achieve a better success rate than the previously discussed methods. This may suggest that even though the graph that is generated based off the data is not necessarily correct it is able to provide some useful information. Therefore, we decided to explore this method further and try to improve its results. We did so by removing extraneous features (ie. first name, gender, etc.) from the data since we believe that these features don't provide information about whether two users are Facebook friends. This feature removal left 143 features that correspond to nine categories (work employer, school attended, etc.) and we noticed that the results improved.

K	Symmetric, 12	Symmetric, 25	Random Walk, 12	Random Walk, 25
2	0.257205	0.2760249	0.2818408	0.2888258
3	0.306058	0.3063577	0.3185714	**0.3829154**
4	0.2955157	0.3484556	0.2810619	0.3114795
5	0.2668111	0.3206767	0.2546436	0.3211741
6	0.2390691	0.2665994	0.2571655	0.2881315
7	0.2163323	0.2554044	0.2341765	0.2596452
8	0.2177299	0.2642461	0.2271114	0.2570181
9	0.2134356	0.2385565	0.240128	0.2412271
10	0.1981741	0.2266557	0.2210587	0.2191494
11	0.202245	0.2181693	0.2273359	0.2147694
12	0.1943502	0.2163638	0.2252125	0.2056446
13	0.2136612	0.2067507	0.218713	0.2033435
14	0.2002827	0.1877179	0.2024464	0.2062839
15	0.1868183	0.2024481	0.1975195	0.1749841
16	0.191354	0.1717852	0.1786037	0.188573
17	0.1909483	0.174734	0.1849027	0.1703809
18	0.1717798	0.1685142	0.1859896	0.1828183
19	0.1759068	0.1706912	0.1688679	0.1681003
20	0.1827177	0.1668853	0.1718183	0.1648561
21	0.1780721	0.1775425	0.1718615	0.1586569
22	0.176439	0.1630868	0.1493036	0.1563862
23	0.1764935	0.1649405	0.1564474	0.1710266
24	0.1811889	0.1722273	0.157104	0.1545049
25	0.1629137	0.1636354	0.1442851	0.1473477
26	0.1610431	0.1536889	0.1374763	0.1425112
27	0.1669558	0.1578861	0.1409979	0.1483216
28	0.1623026	0.1599921	0.1290058	0.1373517
29	0.1506435	0.1710825	0.1390873	0.1469061
30	0.1477557	0.1646745	0.1373459	0.1397119

4.1. Spectral Clustering using MAG

We also tried creating the network in a manner inspired by how MAGs (Multiplicative Attribute Graph) are created. This type of graph was described in [4] and was used in a generative setting rather than a model fitting manner. Since we have the feature data already, we did not generate it like [4] did. Unlike [4] we determined the probability of an edge between two nodes by counting the number of mutually positive features between some node and the reference node and dividing by the number of features. This creates a probability matrix which we can cluster on.

Due to the large number of features we chose to only look at the 9 most relevant macro-features (143 binary features). For example, gender and first name were not included among the features. Also, rather than looking at the features with similar labels as different features we grouped them under one macro-feature. This helped to reduce the size of the feature space and allowed reasonable probability values to be created.

In order to determine the social network structure we then drew an undirected edge between the two nodes if p ~ uniform(0,1) < probability of an edge, otherwise no edge was drawn. This provided a reasonable replacement for the method that was presented in [4] as the method in [4] wouldn't work given the feature data. Again, this method did not generate a graph resembling the original social network, but is an approximation based off of the feature data. Finally, we used the generated probability matrix as input to spectral clustering:

K	Symmetric, 12	Symmetric, 25	Random Walk, 12	Random Walk, 25
2	0.2353248	0.3572603	0.2730392	0.2757532
3	0.2662233	0.3261051	0.2918918	0.269519
4	0.2481203	0.256203	0.233278	0.2214551
5	0.1984962	0.2086884	0.2214308	0.1971063
6	0.1699132	0.1972988	0.1996542	0.1891314
7	0.1632374	0.164866	0.195902	0.1831774
8	0.1580939	0.1569023	0.194251	0.1757258
9	0.1439108	0.1522612	0.1900611	0.1698815
10	0.1315325	0.1495865	0.1643012	0.1680233
11	0.1388441	0.1390384	0.1655278	0.1657698

12	0.1347596	0.1456067	0.1531434	0.1744905
13	0.1248792	0.1433784	0.152821	0.1581387
14	0.1333322	0.136684	0.1556457	0.1683128
15	0.1215663	0.1384401	0.1481982	0.1553286
16	0.1304283	0.1290763	0.1438682	0.1503728
17	0.1209201	0.1281632	0.1380965	0.1400799
18	0.1075407	0.1152608	0.1331056	0.1383646
19	0.1230382	0.1245566	0.146083	0.1309263
20	0.1168455	0.1244618	0.1470185	0.1351128
21	0.1148999	0.1166429	0.1432639	0.1417455
22	0.1150983	0.1186941	0.1483653	0.1369929
23	0.119438	0.1205008	0.1406461	0.1451478
24	0.1122303	0.1295714	0.1592075	0.1567276
25	0.1193488	0.1260772	0.1319399	0.1392603
26	0.1137655	0.113599	0.1303526	0.1427242
27	0.1197393	0.1285776	0.1366732	0.143252
28	0.117568	0.1188506	0.1304065	0.1538297
29	0.1144139	0.1230468	0.122024	0.1354272
30	0.1134981	0.1172226	0.1186042	0.147987

This method did not produce as good of a representation as the previous method.

Community Aside: We also utilized the C++ library found on the Snap website [2]. We provided our generated social networks based off of the feature data to the Community model using three different and highly cited community detection algorithms described in [15, 16, 17]. The results were not as good as our other methods. However, it should be noted that our generated social network is different than that of the true social network.

7. Conclusions and Future work

We have demonstrated the use of various pattern recognition algorithms applied to

the problem of determining the clusters in a social network. We approached this problem using only the feature data of the nodes. This is not a common approach since most related work assume a known network structure. We attempted to cluster based off of the entire feature set, the PCA-transformed features, and the top 100 PCA-transformed features. We also explored reconstructing the social network based off of the feature data alone and then cluster based of of this social network.

The pattern recognition task of identifying clusters is an important one and we believe more important is identifying these clusters with no social network structure. As our results show, an unsupervised approach is difficult, but an important one. Applications for this methodology could be expanded beyond Facebook and possibly into identifying clusters of criminal organizations in social networks.

We attempted to improve the clustering results by manually removing some features which we believe don't determine Facebook friendships. For example, it seems unreasonable to expect that users with the same name are friends. It does seem reasonable, however, to expect that people who work together or went to the same school are friends. We did find significant changes in the results. We evaluated the performance of various clustering methods such as K-means, DBSCAN, SOM, and Spectral clustering. We considered graph clustering algorithms to identify communities, however our inferred social network did not produce good enough results to report. It is important to keep in mind that there is no guarantee that the features in this dataset are even capable of identifying Facebook friendships. Based on our results, we can conclude that valuable information could be extracted from these features using appropriate feature selection and clustering methods. We discovered that it was important to remove features that don't seem to indicate friendships, such as name and gender. We also discovered that, from the methods we tried, SOM and Spectral Clustering performed the best. Thus, a future work could include improvements or modifications of these methods in order to obtain better results. This is worth pursuing because the nature of the dataset is very generic (anonymized binary features) and thus the successful methods could be applied to various other situations in which we try to cluster users into community circles.

REFERENCES

[1] Barbará, D., Chen, P. "Using the fractal dimension to cluster datasets". *Proceedings of the Sixth ACM SIGKDD International conference on Knowledge discovery and data mining.* 260-264. New York, 2000.

[2] Data source: http://snap.stanford.edu/data/egonets-Facebook.html

[3] Schaeffer, S. E. (2007). Graph clustering. *Computer Science Review*, *1*(1), 27–64.

[4] Kim, Myunghwan, and Jure Leskovec. "Multiplicative attribute graph model of real-world

networks." Internet Mathematics 8.1-2 (2012): 113-160.

[5] Leskovec, Jure, and Julian J. Mcauley. "Learning to discover social circles in ego networks." Advances in neural information processing systems. 2012.

[6] Mishra, Nina, et al. "Clustering social networks." Algorithms and Models for the Web-Graph. Springer Berlin Heidelberg, 2007. 56-67.

[7] Yang, Jaewon, and Jure Leskovec. "Community-affiliation graph model for overlapping network community detection." Data Mining (ICDM), 2012 IEEE 12th International Conference on. IEEE, 2012.

[8] Leskovec, Jure, et al. "Microscopic evolution of social networks." Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2008.

[9] Gionis, Aristides, et al. "Dimension induced clustering." Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining. ACM, 2005.

[10] Macropol, Kathy, and Ambuj Singh. "Scalable discovery of best clusters on large graphs." Proceedings of the VLDB Endowment 3.1-2 (2010): 693-702.

[11] Yang, Jaewon, and Jure Leskovec. "Defining and evaluating network communities based on ground-truth." Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics. ACM, 2012.

[12] Handcock, Mark S., Adrian E. Raftery, and Jeremy M. Tantrum. "Model-based clustering for social networks." Journal of the Royal Statistical Society: Series A (Statistics in Society) 170.2 (2007): 301-354.

[13] Falkowski, Tanja, Anja Barth, and Myra Spiliopoulou. "Dengraph: A density-based community detection algorithm." Web Intelligence, IEEE/WIC/ACM International Conference on. IEEE, 2007.

[14] Diesner, Jana, Terrill L. Frantz, and Kathleen M. Carley. "Communication networks from the Enron email corpus “It's always about the people. Enron is no different”." Computational & Mathematical Organization Theory 11.3 (2005): 201-228.

[15] Girvan, Michelle, and Mark EJ Newman. "Community structure in social and biological networks." Proceedings of the National Academy of Sciences 99.12 (2002): 7821-7826.

[16] Clauset, Aaron, Mark EJ Newman, and Cristopher Moore. "Finding community structure in very large networks." Physical review E 70.6 (2004): 066111.

[17] Rosvall, Martin, and Carl T. Bergstrom. "Maps of random walks on complex networks reveal community structure." Proceedings of the National Academy of Sciences 105.4 (2008): 1118-1123.

[18] Whitsitt, Sean, et al. "On the Extraction and Analysis of a Social Network with Partial Organizational Observation." Engineering of Computer Based Systems (ECBS), 2012 IEEE 19th International Conference and Workshops on. IEEE, 2012.

Drew Wicke: dwicke@masonlive.gmu.edu
Mihail Sharov: msharov@masonlive.gmu.edu