Cluster of Tweet Users Based on Optimal Set

Amit Paul
Department of Computer Science & Engineering
BTKIT, Dwarahat
Email: amitpaul06@gmail.com

Animesh Dutta
Department of Information Technology
National Institute of Technology Durgapur
Email: animeshnit@gmail.com

Frans Coenen
Department of Computer Science
University of Liverpool
Email: coenen@liverpool.ac.uk

***Abstract*—Over the years or even decades, researchers are dealing with the problem of duplicate clusters or overlapping clusters in a cluster set. Clusters overlap within each other just as in the case of social networking groups, or grouping movies by genre. In this paper, hierarchical form of clustering is used to cluster user based on interaction which creates numerous clusters with different sizes at different hierarchical level. In doing so, many overlapping clusters are generated but duplicates are not removed. Duplicity possesses a challenge for differentiation. Our work here is two fold. Firstly, to cluster users with different hierarchical levels to generate sets of clusters by level and secondly, to find among the different cluster sets the optimal one by simply using mean and standard deviation. The sense of optimality is different for different requirements. Our work shows that we can have a choice of picking the optimal set by requirement.**

1. Introduction

Clustering is an unsupervised learning where similar objects form groups in such a manner that the objects in a group are mostly similar within the group than in any other group. There are many examples of unsupervised learning. Take the case of online world. Some examples are: group movies by different parameters like genre, old and new, forming groups of users in a social network by their interaction. In all these case overlapping occurs. Social network such as Twitter is used to generate clusters based on only interaction among individuals. Here, interaction means that who has retweeted or replied to whom only. By doing so, many overlapping groups or clusters are generated. No consideration is given to real messages as a whole. There are many approaches proposed in the past regarding the generation of overlapping clusters [1], [2], [3], [4], [5].

Here, an algorithm is proposed for generating clusters sets which are overlapping, but our main focus is on finding the stability of these clusters sets and choosing the most optimal one. The clusters in an individual set are overlapping. A user may be found in more than one cluster and since clustering is done hierarchically somewhere cut has to be made to get the optimum cluster set. The generation of clusters in a set is given by level. Goldberg M K et al [6] compares two sets of clusters that are overlapping. Duplicity among clusters makes matter more complicated. Since, users interact with each other there will be hierarchy of users starting from any one user to form a cluster. The question arises is till which hierarchy level should be chosen. If no level is chosen, then there will be many clusters with numerous duplicates.

Finding duplicate cluster or the number of duplicate clusters within a cluster set is one challenge and finding optimal cluster set by different level, knowing that clusters within the cluster set might be overlapping, is another. The paper focuses on the later part. Heise A et al [7] came up with a method to detect duplicate clusters within a set.

2. Related Work

Complexity of online networks can be guessed by its structure where either the distance between community or group clusters is very short or overlapping. Some [1], [2], [5] are based on finding overlapping communities in an efficient manner and also finding outliers that does not belong to any community [2]. Moreover, [2] uses fuzzy technique to detect communities. Gregory S [1] proposed an improved CONGA algorithm based on 'local' form of betweenness. Palla et al[3] analyses statistical features of overlapping communities and introduces an approach for complex systems and found that overlaps are significant. Newman M E J [4] proposed a method to detect community structure and if there exists any natural cut into nonoverlapping communities. Wang X et al [5] proposed a co-clustering method to group communities using the tags information in messages. Goldberg M K et al [6] measures the distance between two overlapping cluster sets by using three different measures and assumed that each individual cluster contains no duplicates.

Duplicity in clusters is major area of concern for data quality. Detecting duplicates and cleaning is considered a part of preprocessing data in data mining before putting into different uses. Hassanzadeh et al [8] used several clustering techniques to detect duplicity and also used Stringer system for evaluating cluster quality. Banerjee et al [9] worked on overlapping clusters where, some entities are allowed to be member of more than one cluster. Our work is somewhat similar here as duplicates are allowed in the clusters and are not removed. Keeping interesting duplicates will enrich information for finding localness of tweet user in our study further. In a paper [10] stability of clusters is found out by principle component analysis. The work is done on gene but can be extended to any sort of data. Hennig C [11], [12] used Jaccard coefficient to find stability between two cluster set.

3. SCOPE OF THE WORK

In this paper, the work revolves around generating clusters and finding optimal set of clusters by level. The problem addressed is not only about generating overlapping clusters but comparing cluster sets by certain hierarchical level. In a single cluster set, at a particular level, there are numerous clusters which are compared with each other to find the percentages of similarity. At each level cluster size threshold is different. Goldberg et al [3] worked on finding similarity between two

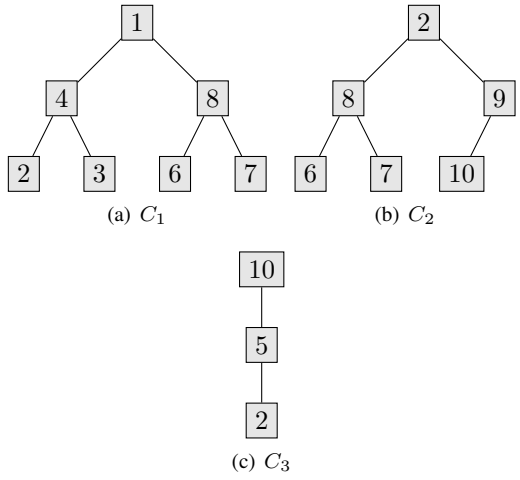
sets of overlapping clusters, but assumed that the individual clusters do not contain duplicates. The work is different in the sense, that in our single set there are numerous clusters that have been compared first and then at each level, to pick the optimal cluster set by level.Moreover, at each hierarchical level there are clusters generated those of which contain duplicates. In our case, clustering is simply based on hierarchy of users by interaction. The main focus of the paper is to find the optimal cluster set within many sets defined by hierarchical levels.

4. Problem Formulation

The goal is to find the most optimal cluster set of clusters with certain threshold like number of users per cluster and level. In doing so, overlapping clusters are first generated by hierarchical level. The cluster set will have application in location estimation of user group for social networking sites. A reasonable knowledge of location can provide advertisers their target users for advertisements. Furthermore, twitter or other social media conversations are the early warning system during any natural disaster, outbreak of disease or emergency related to crime or terrorist attacks. Thus, optimal cluster set will also help in linking or building graph of connected users and pin-pointing the affected user. Though, this part will be for further study.

Problem—: Given number of users, create clusters of users by hierarchy based on interaction among the users by finding who has retweeted to whom.

Let $U = \{U_1, U_2, U_3, .., U_m\}$ be the set of users where m is total number of users and U_i is an individual user. The goal is to create clusters of users. Each individual user is able to form a cluster provided some other user has retweeted. Cluster with only user is discarded. Suppose U={1,2,3,4,5,6,7,8,9,10} where m=10. The figure 1 shows some of the clusters C_1,C_2 and C_3 with level 1.



(a) C_1 (b) C_2 (c) C_3

Figure 1. C_1 , C_2 and C_3 by level 1 with target user 1,target user 2 and target user 10 .

From now on, target user means the user at the root level from where the process starts. Target user 1 1(a) gets retweet and replied messages from user 4 and user 8 only who are at base level and target user 2 1(b) gets from user 8 and user 9 whereas target user 10 1(c) gets from user 5. The base level users for cluster C_1 are users 4 and 8, and for C_2 are users 8 and 9 and for C_3 is 5. For the next level, level 1, users for cluster C_1, C_2 and C_3 are users $\{2, 3, 6, 7\}$, $\{6, 7, 10\}$ and $\{2\}$ respectively. The process is repeated till the desired hierarchical level is reached for the clusters or there are no more users to add to the clusters.

So, clusters formed are $C1 = \{1, 2, 3, 4, 6, 7, 8\}$, $C2 = \{2, 6, 7, 8, 9, 10\}$ and $C3 = \{10, 5, 2\}$. m number of clusters are generated from which few are discarded with only one user. This is called as a cluster set. For each level, a cluster set is generated.

Problem— Given a set of clusters C_m with level l and number of users per cluster greater than threshold τ, find out the most optimal set of clusters among all levels.

Given $P_l = \{C_1, C_2, C_3, C_4, .., C_m\}$ be a set of clusters where C_i is a cluster. Total number of clusters is C_m at level l , where l is any real number. Threshold τ is adjusted to top 5% of the clusters in a set P_l .

So, we get, $Q_l = \{D_1, D_2, D_3, .., D_x\}$ where x is the total number of clusters in the set Q_l and each cluster size is greater than threshold τ. D_i is an individual cluster. For all possible combination of clusters k, let D_i,D_j be a pair of clusters where $i = 1$ to $x - 1$ and $j = i + 1$ to x.

Let S_i be the set of users in cluster D_i and S_j be the set of users in cluster D_j. If $S_i \cap S_j = S_p$. Then, S_p is the set of duplicates or users common in both cluster Di and Dj.

Comparing (S_i, S_p) and (S_j, S_p). We get $d_{k_1} = Sim(S_i, S_p)$ and $d_{k_2} = Sim(S_j, S_p)$ where d_{k_1} and d_{k_2} is the percentage of duplicate users in the cluster pair of D_i and D_j respectively. For all possible combinations k, d_{k_1} and d_{k_2} is calculated. Thus mean is given by

$$m_z = \sum_{k=1}^{k} \frac{d_{k_z}}{k}$$

Standard deviation T_z

$$T_z = \sqrt{\frac{\sum_{k=1}^{k}(d_{k_z} - m_z)^2}{k}}$$

and Co-efficient of Standard Deviation $L_z = T_z/m_z$ where $z = 1, 2$.

Mean, standard deviation and co-efficient of standard deviation are calculated for all levels with threshold cluster size τ. These values are shown in the table II and table III.

5. The Proposed Algorithm

A. Collection of Users By Retweets

In the first part, algorithm 1 detects, separates and selects user identifier by target user in tweets. This will collect the users that a user has re-tweeted or replied to. In the example below $user(i)$ has re-tweeted to $user(x)$, $user(z)$ and more. Moreover, $user(k)$ has re-tweeted to user (i), so is $user(l)$ and $user(m)$.

Example 5.1: A user who has retweeted or replied to whom

$$user(i) \rightarrow user(x), user(z), ..user(..)$$
$$user(k) \rightarrow user(i), user(j), ..user(..)$$
$$user(l) \rightarrow user(i), .., user(..)$$
$$user(m) \rightarrow user(i), ..user(..)$$
$$user(n) \rightarrow user(k), .., user(..)$$
$$user(o) \rightarrow user(k), ..user(..)$$

Algorithm 1

INPUT: Data Corpus with user identifiers
OUTPUT: Collection of Re-Tweet users by target user

1: **while** Not end of line **do**
2: **while** for each target user **do**
3: read each line
4: split each line
5: find pattern 'RT' and '@'
6: collect each ReTweet user by target user
7: **end while**
8: **end while**
9: Return Collection of Re-Tweet 'users' by target user.

B. Directly Connected Users Through Retweets

From the output of the algorithm 1, all the user identifiers are collected by target user. For each target user, the algorithm 2 will search for the users who have re-tweeted or replied to this target user. In this way another corpus is generated that gives the idea about the users who have directly connected to the target user.

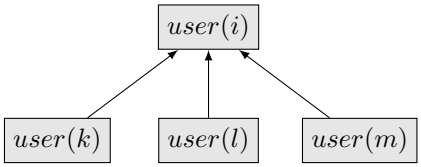


Figure 2. Target $user(i)$ with retweet user

Algorithm 2

INPUT: Collection of Re-Tweet user by target user
OUTPUT: Collection of Re-tweet user id directly linked by target user

1: **while** next line is not null in the file **do**
2: **while** for each user **do**
3: search for those users who have Re-Tweeted the target user
4: **end while**
5: **end while**

Starting from the first target user, the algorithm searches the output file from the algorithm 1, the user who has re-tweeted the target user, then next target user till we get a corpus where we collected users who have re-tweeted a target user directly, considering this level as base level. In figure 2, user(i) is the target user and user(k), user(l) and user(m) has retweeted or replied to user(i). So, all the three users here are directly connected to user(i) and is at base level. Target user is at the root level.

C. Collections of Users by Level and Calculating Stability

In the algorithm 3, level of users is introduced. In the base level, the users who have re-tweeted to the target users are collected only. Figure 3 shows the target user(i) forming a cluster of size seven at level one. Line $1:12$ in the algorithm 3 can be explained as follows. Starting from the base level and

Algorithm 3

INPUT: Target users with base level user i.e directly connected user
OUTPUT:: Cluster set for different levels with mean, standard deviation and co-efficient of standard deviation.

1: put target users with users in a Hash Set
2: **while** read Hash set **do**
3: set level $= n$
4: create hash set for each cluster
5: recursively collect all user till specified level is reached or there is no more user connected search each user
6: **if** user (in level) match retweet user **then**
7: pick the user who has retweeted and add to hash set
8: $count = count + 1$
9: **end if**
10: **end while**
11: level $= n$
12: pick k clusters by count $>$ threshold
13: **if** cluster size $> \tau$ **then**
14: **for** all possible combination of clusters **do**
15: find duplicate users
16: find percentage of duplicates in first and second cluster of the combination pair
17: **end for**
18: **if** cluster size $> \tau$ **then**
19: find within overall corpus and combinations
20: standard deviation, mean, co-efficient of standard deviation
21: **end if**
22: **end if**

target user, users are grouped together who directly interact with the level above it, forming a hierarchy of users. The algorithm creates clusters by target user and when a given level is reached for a particular target user or there are no more users to collect, it switches to next target user, creating another cluster till no more target user are there to create cluster. Line $13:22$ in the algorithm 3 says that for each given level, cluster size is chosen above certain threshold. Threshold is adjusted approximately such that top five percent of the clusters in the set are selected. Each cluster is compared with every other cluster in the set giving the number of duplicates in percentage for the first and second cluster in the pair respectively. For each level, mean, standard deviation and coefficient of standard deviation is calculated from these percentages.

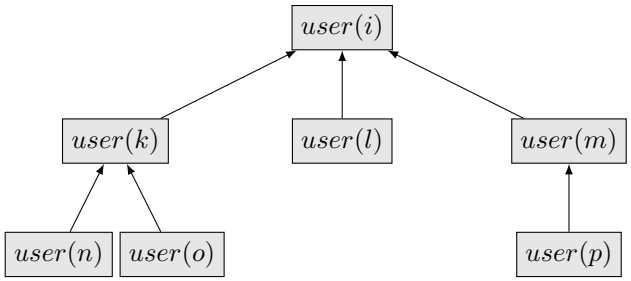


Figure 3. Target $user(i)$ forms a cluster of seven user at level 1.

6. The Experiment

Data is collected from online data repository[1]. The tweets are provided with latitude and longitude by user and separated by tab. In the tweets the users identifiers are amalgamated with the tweets. If some user has re-tweeted then 'RT' is specified in the tweet with identifiers. There are many tweets which are provided with user identifiers starting with '@'. The dataset covers each of the 48 states of US and the District of Columbia [13].

Generally, it is assumed that a user will reply to another user if they are close or within certain home distance range. It is also understood that not all user are in close range who have replied. The data is of 56 MB with 9477 users. With the target users, only the other user identifiers are filtered out from the tweets. The user identifiers are collected by target user.

As discussed in the algorithms, users are grouped by target user at the root level. Each user, a target user at root level, forms a cluster, by adding users, till the specified level is reached or there are no more users to add to the cluster. Single user clusters are discarded. At the starting levels, clusters do not have much duplicity as cluster size is small but moving up the level, as more users are added, chances of getting duplicates increases. As level increases, more and more users are added to the root target user. It is possible, if no limitation of level is introduced, there will be more clusters which are duplicates of each other. So, lower level provide us with numerous small cluster set with very few duplicates and the higher level provide us with many duplicates. Since, level is there, it is challenging to find out the most optimal cluster set within different levels. The table I below shows some fragment of the data generated at level 11. Total number of clusters selected at this level is 350 (Table II and Table III). Take for example, from table I, the zeroth and the first cluster in the pair, the size of zeroth cluster is 1298 and first cluster is 1345. Number of common users in both the cluster is 998. This comes out to be 76.88% and 74.200% similarity with the zeroth and first cluster in the pair respectively.

7. Analysis and Observation

Total number of clusters generated is around 7123 for each level after discarding the single user clusters. Each user generates a cluster at each level. For example, say for level 5, starting from a user, a target user, a cluster is formed. For the cluster, maximum level reached would be level 5 here. Clusters are generated for level 5 till 11. For each level, only top 5% of the clusters are considered. Threshold size of the clusters in the cluster set is set accordingly. Standard deviation is a measure of each dimension, independent of other dimensions, in a data set. Since, data set generated is of percentages similar of all possible combinations of threshold clusters, standard deviation would provide the measure of how data is spread out for each set independently.

In the graph below in figure 4 and table II, at level 5, mean is 0.38 with standard deviation of 3.73 and co-efficient of standard deviation is 9.8. Since the original values are all percentages, mean of 0.38 means that there are only few duplicates in each cluster. In fact, most of the clusters do not have any duplicates and mostly the percentages are all zeros. So, it gives the mean less one. This is explained by the coefficient of standard deviation where deviation is 980%. Generally, if standard deviation and co-efficient of standard deviation is low then the data is stable and also the cluster set. At level 11, table II, where cluster size is greater than 1150, mean is 24.972 and standard deviation is of 10.868 and coefficient of standard deviation is 0.4352. Since mean is around 25, it means that most of the clusters will have around this percentage of duplicity. In other way, 25% of duplicity holds for almost all the pair of clusters. Standard deviation is 10.8685 which is very close to 10.497 for level 10 in table II. From level 10 to 11 standard deviation does not increases sharply. Moreover, till level 8, table II and III, co-efficient of standard deviation is very high. Thus, level 10 is the best solution or optimal cluster set. Level 10 is the most optimal cluster, which can be used further for finding the localness of tweet user. Since the values of different measures in the table II and table II do not vary much, it can be guessed that the cluster sizes are not too far from each other. Figure 4 and figure 5 shows the chart of different measures by levels.

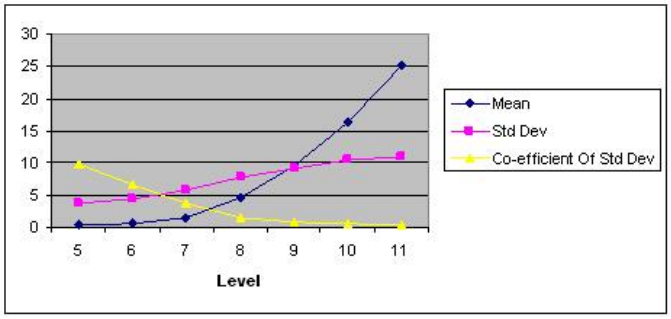


Figure 4. Measures by level for 1st one in cluster pair combination set

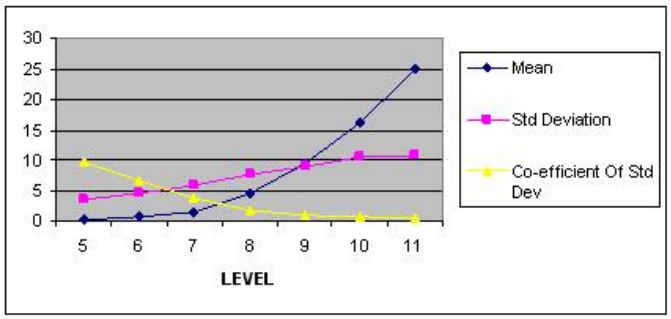


Figure 5. Measure by level for 2nd one in cluster pair combination set

[1] http://www.ark.cs.cmu.edu/GeoTwitter

Cluster Number(i)	Cluster Number (j)	Size (ith Cluster)	Size (jth Cluster)	Number of Common Users	Percentage similar(ith Cluster)	Percentage similar(jth Cluster)
0	1	1298	1345	998	76.8875	74.2007
0	2	1298	1204	688	53.0046	57.1429
0	3	1298	1181	621	47.8428	52.5826
0	4	1298	1253	593	45.6857	47.3264
0	5	1298	1193	592	45.6086	49.6228
…	…	…	...	…	…	…
0	349	1298	1318	184	14.1757	13.9605
1	2	1345	1204	790	58.7361	65.6146
…	…	….	…	…	…	…
…	…	…	…	…	…	…
348	349	1476	1318	1076	72.8997	81.6389

Table I. CLUSTERS AT LEVEL 11 SHOWING CLUSTER SIZE OF DIFFERENT CLUSTERS(0,1,2, 3, 4, 5, .., 348, 349) AND PERCENTAGES SIMILAR, BY COMPARING WITH EACH ITH CLUSTER AND JTH CLUSTER RESPECTIVELY.

	Level 5. cluster size >190 Number of cluster=344	Level 6. cluster size > Number of cluster=355	Level 7. cluster size >400 Number of cluster=355	Level 8. cluster size >560 Number of cluster=355	Level 9. cluster size >745 Number of cluster=354	Level 10. cluster size >950 Number of cluster=344	Level 11. cluster size >1150 Number of cluster=350
Mean	0.38	0.68932	1.55	4.62	9.63	16.308	24.972
Standard deviation	3.73	4.58446	5.79	7.75	9.096	10.497	10.8685
Coefficient of standard deviation	9.8	6.65	3.73	1.67	0.94439	0.64	0.4352

Table II. MEAN, STANDARD DEVIATION AND CO-EFFICIENT OF STANDARD DEVIATION BY DIFFERENT LEVELS, CLUSTER SIZE AND TOTAL NUMBER OF CLUSTER IN A SET FOR THE FIRST ONE IN THE CLUSTER PAIR.

	Level 5. cluster size >190 Number of cluster=344	Level 6. cluster size > Number of cluster=355	Level 7. cluster size >400 Number of cluster=355	Level 8. cluster size >560 Number of cluster=355	Level 9. cluster size >745 Number of cluster=354	Level 10. cluster size >950 Number of cluster=344	Level 11. cluster size >1150 Number of cluster=350
Mean	0.376	0.686	1.546	4.6	9.564	16.2514	24.89
Standard deviation	3.698	4.544	5.77	7.71	9.047	10.461	10.83
Coefficient of standard deviation	9.79	6.62	3.732	1.67	0.945	0.6437	0.4351

Table III. MEAN, STANDARD DEVIATION AND CO-EFFICIENT OF STANDARD DEVIATION BY DIFFERENT LEVELS, CLUSTER SIZE AND TOTAL NUMBER OF CLUSTER IN A SET FOR THE SECOND ONE IN THE CLUSTER PAIR.

8. CONCLUSION

The optimal cluster set will be used to cluster the tweets by users and target user to find the localness of tweet by user. Optimality of cluster set provide us with clusters which can be considered as set having duplicity not to many or not to scarce. Too many duplicity will have information very similar to the other clusters and differentiation between the two will be difficult and on the other hand small clusters having very less duplicity will have very little information, since the cluster of users can be used to cluster the tweets too. Moreover, for further study, if duplicity is there among a pair of clusters, users can be provided with different weights for appearing at different level among the pair. It will also be interesting to find among the cluster set a user who is present among how many clusters and at which level and how it is linked.

REFERENCES

[1] S. Gregory, "A fast algorithm to find overlapping communities in networks," in *Machine learning and knowledge discovery in databases*. Springer, 2008, pp. 408–423.

[2] T. Nepusz, A. Petróczi, L. Négyessy, and F. Bazsó, "Fuzzy communities and the concept of bridgeness in complex networks," *Physical Review E*, vol. 77, no. 1, p. 016107, 2008.

[3] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, vol. 435, no. 7043, pp. 814–818, 2005.

[4] M. E. Newman, "Modularity and community structure in networks," *Proceedings of the national academy of sciences*, vol. 103, no. 23, pp. 8577–8582, 2006.

[5] X. Wang, L. Tang, H. Gao, and H. Liu, "Discovering overlapping groups in social media," in *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 2010, pp. 569–578.

[6] M. K. Goldberg, M. Hayvanovych, and M. Magdon-Ismail, "Measuring similarity between sets of overlapping clusters," in *Social Computing (SocialCom), 2010 IEEE Second International Conference on*. IEEE, 2010, pp. 303–308.

[7] A. Heise, G. Kasneci, and F. Naumann, "Estimating the number and sizes of fuzzy-duplicate clusters," in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, 2014, pp. 959–968.

[8] O. Hassanzadeh, F. Chiang, H. C. Lee, and R. J. Miller, "Framework for evaluating clustering algorithms in duplicate detection," *Proceedings of the VLDB Endowment*, vol. 2, no. 1, pp. 1282–1293, 2009.

[9] A. Banerjee, C. Krumpelman, J. Ghosh, S. Basu, and R. J. Mooney, "Model-based overlapping clustering," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 2005, pp. 532–537.

[10] A. Ben-Hur and I. Guyon, "Detecting stable clusters using principal component analysis," *Functional Genomics: Methods and Protocols*, pp. 159–182, 2003.

[11] C. Hennig, "Cluster-wise assessment of cluster stability," *Computational Statistics & Data Analysis*, vol. 52, no. 1, pp. 258–271, 2007.

[12] ——, "Dissolution point and isolation robustness: robustness criteria for general cluster analysis methods," *Journal of multivariate analysis*, vol. 99, no. 6, pp. 1154–1176, 2008.

[13] J. Eisenstein, B. O'Connor, N. A. Smith, and E. P. Xing, "A latent variable model for geographic lexical variation," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2010, pp. 1277–1287.