2013 17th International Conference on Information Visualisation

A Semi-Supervised Approach to Visualizing and Manipulating Overlapping Communities

Patrick M. Dudas
School of Information Sciences
University of Pittsburgh
Pittsburgh, USA
pmd18@pitt.edu

Martijn de Jongh
School of Information Sciences
University of Pittsburgh
Pittsburgh, USA
mad159@pitt.edu

Peter Brusilovsky
School of Information Sciences
University of Pittsburgh
Pittsburgh, USA
peterb@pitt.edu

***Abstract*— When evaluating a network topology, occasionally data structures cannot be segmented into absolute, heterogeneous groups. There may be a spectrum to the dataset that does not allow for this hard clustering approach and may need to segment using fuzzy/overlapping communities or cliques. Even to this degree, when group members can belong to multiple cliques, there leaves an ever present layer of doubt, noise, and outliers caused by the overlapping clustering algorithms. These imperfections can either be corrected by an expert user to enhance the clustering algorithm or to preserve their own mental models of the communities. Presented is a visualization that models overlapping community membership and provides an interactive interface to facilitate a quick and efficient means of both sorting through large network topologies and preserving the user's mental model of the structure.**

***Keywords*-visualization; overlapping communities; semi-supervised clustering; user-defined cliques**

I. INTRODUCTION

Visualizations are tools used to express both the structure of the data and cognitive mapping of the user observing and interacting with this data [1]. In a traditional sense, graph based data is handled using simple vertex and edge based representations and in some cases, cliques are derived from these graphs (networks or sub-networks) and presented to the user in a means that help segregate and dissect closely associated vertices from one another. This technique works in cases where the data is segmented into heterogeneous groups and vertices are easily discernible from one another; in the cases that data points (vertices) fall into multiple different groupings, there is a need to allow for cliques or communities to overlap. These overlapping communities require new representation in that the visualization needs to be created to highlight areas where the vertices fall into single inclusion communities and multi-communities.

Building upon the cognitive mapping and mental models these network visualizations with overlapping communities communicate, we consider that when there is a gray area for vertices to fall into (in terms of community identity) the line between absolute solution and best guess solution fades, and human involvement and interaction provide a critical improvement in acceptable grouping. In this paper we introduce user-defined cliques as an approach to provide human feedback in the process of mixed-initiative community visualization. Optimally, there is mixture of both the machine learning algorithm and the user generated cliques. Presented is an approach to combine both semi-supervised clustering and cluster visualization, rich in its interaction and aesthetics to provide the user with both an understanding of the machine learning clustering and the ability to apply their own structure based on their mental model of the network.

In section 2 we present the previous work related to these topics. Section 3 goes into detail of the visualization, the machine learning clustering algorithm (Label Propagation), and the novel approach of using a semi-supervised interface that allows and adapts to the user-defined and machine-learning cliques. Section 4 provides an example in which we apply a dataset, created from UMAP (User Modeling, Adaptation and Personalization) publications, to showcase our design. Sections 5 and 6 conclude with our future work.

II. BACKGROUND

A. Visualization

There are a plethora of different, mainstream types of visualization software packages available for network data including Gephi [2], Pajek [3], Ucinet [4]. These are all limited to non-overlapping clustering approaches and do not allow the user the ability to put vertices into new groups without having to change meta-data about the vertex itself. Our approach allows for the inclusion and visualization of overlapping communities and an interactive, semi-supervised clustering approach.

Using biclusters in gene-expression, BicOverlapper [5] incorporates a hull based visualization using a singular color to display higher levels of overlap. An interesting approach to overlapping communities is Overlapper [6], where they improved upon their design in BicOverlapper by highlighting vertices that overlapped by using a pie chart to show the number of overlaps. Vertices that have overlap are easily identified and to what degree. We expand on this by including a variety of different colors (ten colors) with similar hull opacity to showcase multiple cliques definitively and areas of high overlap. Using multiple colors also allows for salient understanding of which clique each vertex comes from and where they

1550-6037 2013

DOI 10.1109/IV.2013.23

overlap. Additionally, we provide an improved pie chart structure that allows for both the identification of overlapping nodes, and to what degree they overlap.

B. Clustering

Since the work by Girvan & Newman [7], we have seen many different approaches to clustering network structures. See Fortunato [8] for a comprehensive overview. Among other things, approaches differ by their quality function [7], the ability to allow overlapping communities [9], and how cluster assignments are propagated through the network [10]. For our approach we applied an extended version of the Label Propagation Algorithm (LPA), which was proposed by Raghavan, Albert, and Kumara [11]. LPA iteratively determines the final cluster assignments. It initializes all vertices with a unique label, and then proceeds to update vertex labels by checking the labels of their neighbors. The most frequently occurring label among a vertex's neighbors is chosen as its new label. Ties are broken randomly. During all iterations, all vertices are (possibly) assigned a new label asynchronously. The process continues until it converges on a stable set of label assignments, which usually occurs within a few iterations.

C. Interactive Visualization

When it comes to this sensemaking and user-defined approach in a visualization, Apolo [12] provides an interface incorporating both user interactions and machine learning in large datasets. They accomplish this by building on a single vertex and as users provide a paper of interest to interface, the network-like visualization builds by providing cited work and visual clues based on number of citations and relevance. Building on this visualization, TourViz [13] divides sub-domains of user interests into convex hulls to help segment multiple topics of interest. In contrast to this work, our approach takes into account the entire network structure, providing overall topology. Also, we provide not only the utilization of color changes (to distinguish group assignment), but also shape distinction by user-defined clusters and opacity changes to discern vertices already selected and grouped.

III. METHODS

A. Visualization Approach

We opted to use D3.js for our force-directed framework, which utilizes a Verlet integration with simple graph constraints [14]. Our prototype is a community enhanced force-based diagram, with the modification that we provide centralization vertices for all clique centroids. The reasoning for this was two-fold: 1) produce a more efficient, interactive visualization and 2) maximization of convex hull coverage. These are both covered in detail below.

First, addressing efficiency of the visualization, one of the major rendering issues found in most force-directed visualization of networks is not always the number of vertices and edges, but the edge overlap. This is due to that fact that when it comes to constructing the graph, edge overlap can be the more disorienting aesthetic properties [15] to deal with, and when faced with this problem without altering the edge shape can lead to an NP-hard problem [16]. We reduced overlap and the number of connections by creating a single vertex for each clique as a marker for the entire clique. Highly dense cliques have the potential to have $O(n^2)$ number of connections between each vertex. With our approach, it is guaranteed that each clique will have a maximal number of connections of $O(\mathrm{n})$. While this may appear to reduce the natural connectivity and topology of the graph, clustering is completed utilizing the original network layout so as to preserve these inner-cluster relationships but minimize their overall effect on the rending of the visualization. Fig. 1 (left) illustrates a traditional arrangement of a network topology. Instead of this layout we create a single vertex as a central point for the entire clique (Fig. 1 – Centroid Node). The centroid vertex has only one connection to all vertices in the clique, Fig. 1 (right).

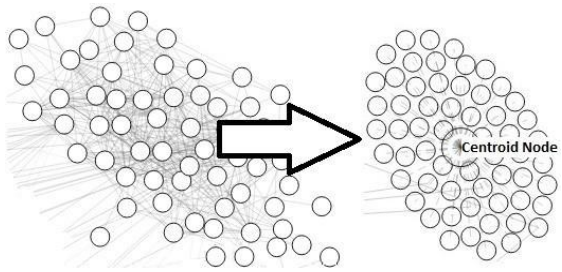


Figure 1. Removing Edge Crossing and Reducing Layout Computations by Using Centroid Vertex

To examine this approach, we completed measures for two separate datasets obtained from UMAP. The datasets include the original topology (Original) and our centroid approach (Centroid). We compared the number of vertices (v), number of edges (e), average cluster coefficient (CC), average path length (PL), and the total time (TT) to converge. The total time to converge was a measurement designated by our group built upon the tick() function in D3.js. To create the force-based diagram, d3.js implemented a tick() function to move the vertices and edges to their final arrangement based on link strength, link distance, gravity, and charge. These constraints are balanced using Dwyer's layout constraints [17], where as shown in Fig. 2 the distance d must be minimized.

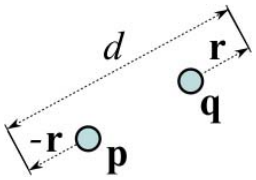


Figure 2. Representing Graph Constraints [17]

This is done by finding the smallest r value to satisfy this constraint, where r [17] is defined as:

$$r = \frac{(d - |\boldsymbol{p}-\boldsymbol{q}|\boldsymbol{pq})}{2|\boldsymbol{p}-\boldsymbol{q}|} \quad (1)$$

All constraints were kept constant throughout both trails. We defined the total time as the amount of time it took to finally hit equilibrium with the force-based criteria. For Table 1 the centroid topology took less time to come to equilibrium for both trials. Both factors that lead to better times for the centroid topology was the number of edges and connectivity of the network itself.

TABLE I. COMPARING TIME TO CONVERGE, ORIGINAL VERSUS CENTROID TOPOLOGIES

Dataset	V	e	CC	PL	TT
1-Original	766	8038	0.876	1.97	02:18.1
1-Centroid	766	2262	-	-	01:21.9
2-Original	766	2506	0.812	4.596	01:56.5
2-Centroid	766	1714	-	-	01:32.6

To maximize awareness of the overlapping communities a convex hull [18] was applied to showcase the cliques and the overlap. The convex hull benefits by using this centroid vertex by not forcing the visualization graph constraints to focus strictly on the distances between vertices V, but using the repulsive force k provided in the Barnes-Hut quad-tree [19]. The repulsion is approximated using a charge value q_c determined from the quad-tree resulting in:

$$d_{u,v} = \sum_{(u,v)\in VxV} \frac{1}{\sqrt{(p_u - q_u)^2 + (p_v - q_v)^2}} \quad (2)$$

$$k_{u,v} = q_c\left(d_{u,v}\right)^2 \quad (3)$$

In the case where the $q_c < 0$, the graph layout is computed. This provides an even distribution of vertex distances and makes a more aesthetically pleasing interface by removing overlap of the vertices. Also, community overlap is easier to identify, because in the scenario where a vertex overlaps with two communities. It will have edges to both centroids or multiple centroids in the case of more than two overlaps and will pull evenly based on the force-based parameters.

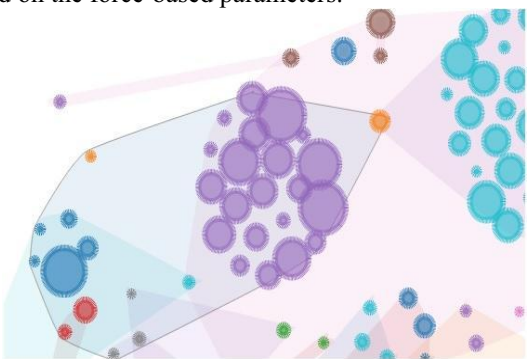


Figure 3. Example of Overlapping Community Visualization

B. Extending Label Propagation

We have extended the original LPA to allow it to be an active part in our interactive clustering visualization. We added to ability to run the algorithm on weighted graphs, changing the label picking criterion from the most frequent label of the neighbors to a version where the frequencies are modified by the edge weights. Furthermore, the labeling process was extended to allow for discovery of overlapping clusters. The calculated cluster label weights are further weighted by the proportions they appear in the label distribution of the adjacent vertices. The proportion weights for the new set of cluster labels of a vertex are calculated after selecting the top n labels. The calculated weights of these n labels are normalized and stored with the new cluster labels of the vertex. A weight w_j for cluster label j is calculated using edge weights e_i of adjacent vertex i and cluster label proportion p_{ij} of vertex i for label j:

$$w_j = \sum_{i=1}^{n} e_i \frac{p_{ij}}{\sum_{j=1}^{c} p_{ij}} \quad (4)$$

To facilitate an interactive cluster visualization experience, we have made several modifications to LPA to make this possible. The original algorithm paper proposed an approach to seed a few vertices with cluster labels and leaving the rest unlabeled, allowing for clusters to form around those seeds. We have taken a slightly different approach, allowing vertices to be fixed. Once a vertex is fixed, it will no longer update its label. This allows clusters to grow around vertices in a similar way, with the added benefit that we can choose to fix vertices in an already existing complete set of cluster assignments and rerun the algorithm on this cluster assignment to update it. Second, featured prominently in our visualization, is granting the user the ability to group vertices. Effectively this allows grouped vertices to behave as a single vertex. When any of the vertices of the group updates its (set of) label(s), all others follow suit.

Groups can also be fixed; if one of its members is a fixed vertex, the whole group becomes fixed.

This feature and the ability to recover overlapping communities have made it necessary to adjust the algorithm's stopping criterion. Originally, the algorithm terminates after all vertices belong to the cluster to which the majority of its neighbors belong. Allowing vertices to be groups can cause the groups to flip between vertex clusters labels, which could cause the algorithm to get stuck in an infinite loop.

To support overlapping clusters, the stopping criterion needed to be generalized. Before, the newly chosen cluster label was compared with the current assignment. If they differed, the vertex was assigned the new label and the global cluster assignment was deemed to be unstable; this required the algorithm to continue to iterate until none of the vertices would have to update their label assignment anymore.

To guarantee algorithm termination with the new features in place, two adjustments were necessary:

1) A 2-phase stopping heuristic was put in place. After 500 iterations, cluster assignment distributions are recorded. After a customizable number of iterations are reached, the algorithm assigns the most frequently occurring (set of) cluster label(s) of this group to the group as the final assignment. After all groups are assigned their cluster(s), the algorithm terminates.

2) We started using cosine similarity [20] to measure how close the new (set of) cluster(s) is to the old assignment. Once a preset minimum cluster similarity is reached, the algorithm terminates. Cosine similarity between cluster sets *C* of vertices *a* and *b* is defined as follows:

$$\cos_{ab}(\alpha) = \frac{\sum_{i=1}^{n} C_{ai} \times C_{bi}}{\sqrt{\sum_{i=1}^{n} (C_{ai})^2} \times \sqrt{\sum_{i=1}^{n} (C_{bi})^2}} \quad (5)$$

C. Interaction Workflow

As part of an effort to provide a more interactive experience, work has been done to allow for a more dynamic workflow for clustering and visualizing data. The goal of this dynamic workflow is to allow users to execute clustering algorithms from within the visualization display and to interact with the cluster results.

The current prototype also combines a few affordances to help guide the user in the interactions themselves [21]. Edges on the graph reflect similarity between vertices (see *Section 4 – UMAP Example* for details on similarity measure used) and represented by the thickness of the line equating higher similarity. Vertices and text boxes provide a pointer cursor to make them known as selectable objects. Vertex size is determined by calculating the degree centrality. Also, each vertex is represented using a pie diagram which represents the amount of weighting provided by LPA to showcase to what degree each vertex belongs to this community. Fig. 4 shows an example of this weighting.

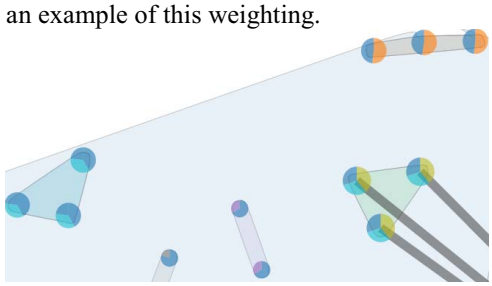


Figure 4. Pie Chart Aesthetics to Highlight LPA Values

Once a vertex is selected, the user has the option (and is encouraged) to select multiple vertices to group. This is completed using a control-click, which is comparable to traditional multi-file select [22].

Objects that are selected are then temporarily grouped into a user-defined sub-group and labeled using different glyphs, differentiating themselves from the predefined, machine learned groups that are labeled using color. The groups are then automatically passed to a modified version of the LPA. The information provided by the user is then incorporated in the clustering process and after completion, the cluster assignments of the current dataset are updated in the visualization display. For the final affordance, the user-defined group is differentiated by changing the opacity to be completely opaque.

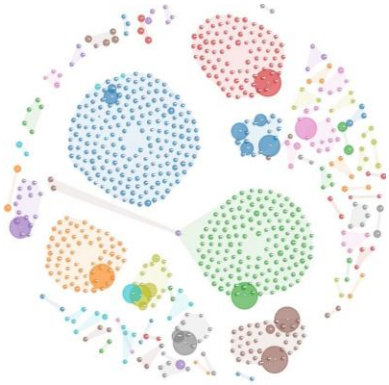
IV. UMAP Example

A. UMAP Dataset

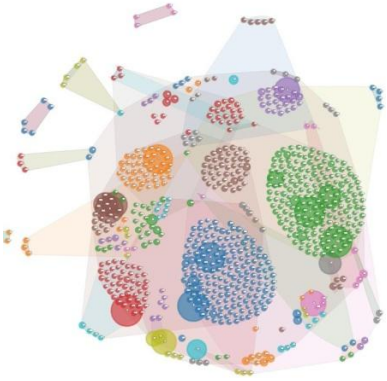
To run an information evaluation of our approach, we explored it using a real-world dataset that represents a similarity measure among a group of authors that have published in the UMAP (User Modeling, Adaptation and Personalization conference series. To create this dataset, we have extracted data from the DBLP [23] database, which created a 766 vertex and 8038 edge network. DBLP is a computer science bibliography database. It has indexed over 2 million articles from many conferences, books and journals. We have parsed a dump of DBLP database to extract the relevant publications and other information for the group of UMAP authors we are examining. This data was used to build similarity relationships based on the Jaccard index [24], author similarity depends on the number of papers co-authored together, taking into account how many papers each of the authors authored in total.

B. Visualization Example

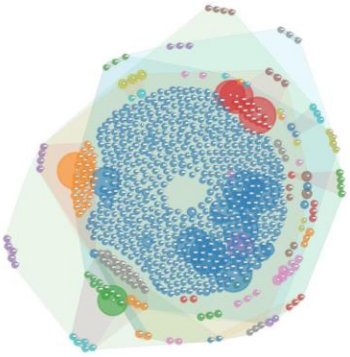
We visualized the UMAP data with three different settings for overlapping cluster: 1 (no overlap), 3, and 5. The results are shown in Fig. 5.



(a)



(b)



(c)

Figure 5. Visualizing UMAP data with no, 3, and 5 overlapping clusters respectively

For this example we will be using the 3 overlapping clusters (Fig. 5 - b). To proceed, the user would select multiple vertices to group them together (using control-clicks). Selections will be based on overlapped areas with high correlations, depicted using the pie chart in Fig. 6.

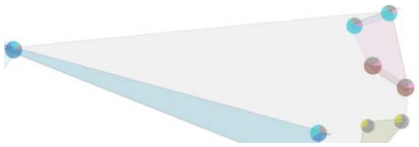


Figure 6. Pie Charts Showcasing LPA correlation

Once selected, the outline of the vertex changes to provide affordance that the vertex was selected. After the user has stopped selecting vertices, the visualization updates to showcase the user-defined group by using a different glyph, shown in Fig. 7. This triggers the visualization to update values in the database and activates the LPA to rerun with updated ground truth. Results are then sent back to visualization via an AJAX callback and the visualization shows the updated graph.

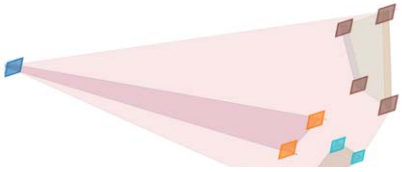


Figure 7. User-Defined Clusters

The final aesthetic is the color change to signify group inclusion and opacity change to illuminate already clustered nodes. Fig. 8 shows the updated community, the user-defined community (using an alternate glyph), and the processed vertices (changing the opacity to complete opaque). This process continues and the user-defined groups will be differentiated from one another by using a new glyph.

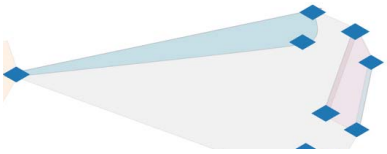


Figure 8. Final Aesthetic to Highlight Completed Nodes

V. FUTURE WORK

To continue this work we want to validate first the claim that our approach allows for easy understanding of community identity using convex hulls and optimization proposed in this paper. To do this, we will need to provide a user-study of this mechanism and show that a mixture between user-defined and machine learned clustering can build an optimal and accurate model of the network topology and the user's mental model.

There is also a novel and yet strikingly obvious need to understand what it means to belong to multiple overlapping communities. Much work has been done in social capital that illustrates the strength of weak ties in bridging multiple communities [25]. We are interested in

studying these networks more in depth to see if these vertices that fall between multiple communities can be defined more precisely using arguments from social capital. Gilbert [26] supports the claims made in this paper in regards to a spectrum to vertex-to-vertex variability and we believe that social capital and overlapping communes go hand in hand.

VI. Conclusion

Presented is a semi-supervised clustering implementation using convex hulls to illustrate overlapping community structures. We believe that when it comes to community identification, data points cannot be restricted to a single clique inclusion, that data points can have a gray area that can mask them into multiple communities. To that degree, even when clustering algorithms provide the ability to compute overlapping communities there is necessity to involve the experts in those networks to obtain noiseless, error-free networks and community detection.

To handle the user interactions involved with this visualization, great care was taken in making sure both the visualization itself and the LPA could handle the size of the network and the interactions. Modifications were made to the visualization to showcase the communities in better quality and minimize edge crossing. The LPA was adjusted to allow for fixed vertices within the algorithm, allowing it to obtain an optimal solution in a relatively small amount of time. We believe that as networks are examined more in-depth, that platforms like this that take both visual information and user involvement can balance out both the human mental model of the network and the machine learning techniques used for efficiency.

References

[1] A. Tversky and I. Simonson, "Context-dependent preferences," *Management science,* vol. 39, pp. 1179-1189, 1993.

[2] M. Bastian*, et al.*, "Gephi: An open source software for exploring and manipulating networks," 2009.

[3] V. Batagelj and A. Mrvar, "Pajek-program for large network analysis," *Connections,* vol. 21, pp. 47-57, 1998.

[4] S. P. Borgatti*, et al.*, "Ucinet for Windows: Software for social network analysis," 2002.

[5] R. Santamaría*, et al.*, "BicOverlapper: a tool for bicluster visualization," *Bioinformatics,* vol. 24, pp. 1212-1213, 2008.

[6] R. Theron*, et al.*, "Overlapper: movie analyzer," *Infovis Confererence Compendium,* pp. 140-141, 2007.

[7] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical review E,* vol. 69, p. 026113, 2004.

[8] S. Fortunato, "Community detection in graphs," *Physics Reports,* vol. 486, pp. 75-174, 2010.

[9] G. Palla*, et al.*, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature,* vol. 435, pp. 814-818, 2005.

[10] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *science,* vol. 315, pp. 972-976, 2007.

[11] U. N. Raghavan*, et al.*, "Near linear time algorithm to detect community structures in large-scale networks," *Physical Review E,* vol. 76, p. 036106, 2007.

[12] D. H. Chau*, et al.*, "Apolo: making sense of large network data by combining rich user interaction and machine learning," 2011, pp. 167-176.

[13] D. H. Chau*, et al.*, "TourViz: interactive visualization of connection pathways in large graphs," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2012, pp. 1516-1519.

[14] M. Bostock*, et al.*, "D³ Data-Driven Documents," *Visualization and Computer Graphics, IEEE Transactions on,* vol. 17, pp. 2301-2309, 2011.

[15] H. Purchase, "Which aesthetic has the greatest effect on human understanding?," in *Graph Drawing*, 1997, pp. 248-261.

[16] M. R. Garey and D. S. Johnson, "Crossing number is NP-complete," *SIAM Journal on Algebraic Discrete Methods,* vol. 4, pp. 312-316, 1983.

[17] T. Dwyer, "Scalable, versatile and simple constrained graph layout," in *Computer Graphics Forum*, 2009, pp. 991-998.

[18] F. P. Preparata and M. I. Shamos, "Introduction," *Computational Geometry,* pp. 1-35, 1985.

[19] J. Barnes and P. Hut, "A hierarchical O (N log N) force-calculation algorithm," 1986.

[20] G. Salton, "Automatic text processing. 1989," ed: Addison Wesley, 1989.

[21] A. Dieberger*, et al.*, "Social navigation: techniques for building more usable systems," *interactions,* vol. 7, pp. 36-45, 2000.

[22] A. Ritter and S. Basu, "Learning to generalize for complex selection tasks," in *Proceedings of the 14th international conference on Intelligent user interfaces*, 2009, pp. 167-176.

[23] M. Ley, "The DBLP computer science bibliography: Evolution, research issues, perspectives," in *String Processing and Information Retrieval*, 2002, pp. 481-486.

[24] P. Jaccard, *Etude comparative de la distribution florale dans une portion des Alpes et du Jura*: Impr. Corbaz, 1901.

[25] R. S. Burt, "The social capital of structural holes," *The new economic sociology: Developments in an emerging field,* pp. 148-190, 2002.

[26] E. Gilbert and K. Karahalios, "Predicting tie strength with social media," in *Proceedings of the 27th international conference on Human factors in computing systems*, 2009, pp. 211-220.