Journal of the Association for Information Systems (2021) **22**(6), 1625-1658
doi: 10.17705/1jais.00711

RESEARCH ARTICLE

ISSN 1536-9323

Mining Online Social Networks: Deriving User Preferences through Node Embedding

Mahyar Sharif Vaghefi[1], Derek L. Nazareth[2]

[1]College of Business Administration, University of Texas at Arlington, USA, mahyar.sharifvaghefi@uta.edu
[2]Lubar School of Business, University of Wisconsin-Milwaukee, USA, derek@uwm.edu

Abstract

In the last decade, online social networks have become an integral part of life. These networks play an important role in the dissemination of news, individual communication, disclosure of information, and business operations. Understanding the structure and implications of these networks is of great interest to both academia and industry. However, the unstructured nature of the graphs and the complexity of existing network analysis methods limit the effective analysis of these networks, particularly on a large scale. In this research, we propose a simple but effective node embedding method for the analysis of graphs with a focus on its application in online social networks. Our proposed method not only quantifies social graphs in a structured format but also enables user preference identification, community detection, and link prediction in online social networks. We demonstrate the effectiveness of our approach using a network of Twitter users. The results of this research provide valuable insights for marketing professionals seeking to target personalized content and advertising to individual users as well as social network administrators seeking to improve their platform through recommendation systems and the detection of outliers and anomalies.

Keywords: Communities of Interest, Individual Preference, Recommendation Systems, Social Network Analysis, User Embedding, Visualization

Sudha Ram was the accepting senior editor. This research article was submitted on August 28, 2020 and underwent one revision.

1 Introduction

Online social networks as IT-based artifacts have introduced many changes to modern life and play increasingly significant roles in disseminating information during global events—for example, the 2016 and 2020 US presidential elections and the ongoing coronavirus pandemic. Social networking platforms have also revolutionized online marketing. Targeted advertising, also known as "behavioral targeting," is one area that has benefited greatly from this new paradigm. Targeted advertising, which was pioneered by search engines such as Google, uses cookies and other tracking tools to derive individuals' interests based on browsing behavior and uses the derived information to choose appropriate ads to be displayed to users. Research has shown that targeted advertising can significantly increase both the click-through rate and the purchase intentions of individuals (Yan et al., 2009, Goldfarb & Tucker, 2011). It also improves the competitive environment among advertisers by enabling them to target different segments of users, ultimately leading to lower advertising costs for businesses compared to traditional forms of advertising (Chen & Stallaert, 2014). The fine granularity of information collected/shared on social media platforms makes this type of media especially attractive to marketers interested in targeted advertising. In many cases, what users share on social media platforms can reveal much more about them than is immediately apparent. For example, a simple 140-

character tweet posted on Twitter provides information that goes far beyond those few characters. New analytical tools make it possible to extract complicated patterns from large volumes of data, which can reveal individual demographic information, where one lives, personality traits (Arnoux et al., 2017), and even brand perceptions (Culotta et al., 2016). Such information can be utilized by marketers to pitch products and services directly to users on social platforms.

However, behavioral targeting is not without its challenges. Advertisers must allocate resources to control and improve the quality of targeting. The quality of targeting can be measured using two metrics: accuracy and recognition (Gal-Or et al., 2006).[1] Accuracy specifies the percentage of individuals in the predicted target group that are actually in the target group, and recognition measures how well the predicted target group represents all individuals in the actual target group. It is not easy to find a balance between the two metrics. In order to achieve high accuracy, marketers must limit their predicted target group to smaller samples encompassing more user information and greater confidence about their behavior. However, companies using this approach may lose potential customers, as the available user-level information is generally limited. In contrast, achieving a high recognition rate requires broadening the predicted target group and reaching out to higher numbers of users. This approach may lead to targeting the wrong audience, which wastes company resources and individuals' time. Targeting the wrong audience is generally expected to have an adverse effect on companies in the long run by making customer retention and customer relationship management more difficult (Rollins et al., 2014). In addition, targeting the wrong audience can foster negative word of mouth in social networks. Thus, the role of profiling methods in identifying user attributes is crucial because it can improve the quality of predicted target groups. A good profiling method can improve both accuracy and recognition.

By definition, a user's online profile is "a summary of a user's interests and preferences revealed through the user's online activity" (Trusov et al., 2016). However, compiling and analyzing online activities, especially for platforms other than online social networks, are not simple tasks. For example, search engines need to continuously track user search and browsing behaviors, and e-commerce websites need to keep track of transactions, product views, and individual shopping carts. This continuous data collection raises privacy concerns, as many people do not like to be continuously tracked (Aguirre et al., 2015) even though they perceive personalized ads to be useful (Bleier & Eisenbeiss, 2015). Analysis of such large data sets is also challenging. While a number of techniques have been proposed for efficient user profiling (Trusov et al., 2016), they still require considerable computational power for real-time data analysis. Other possible challenges include the lack of data accessibility for marketers, reliance on third-party publishers for user analysis and targeting, and the provision of partial views to user attributes. In this study, we argue that many of the above-mentioned difficulties can be mitigated in the context of online social networks. We focus on the structure of social networks and argue that it contains information about individuals and groups that is not overtly apparent when examining the network. This information is useful for inferring user characteristics, making recommendations, and predicting new relationships within the social network.

The structure of a social network is typically represented as a graph and may lack rich descriptions of individual users. Given the large number of users in a social network, matrix representations of the network result in extremely large and very sparse matrices, rendering most machine learning techniques ineffective. Reducing the size of the data set and eliminating some of the sparsity is necessary for effective application of machine learning techniques. Graph embedding, where a portion of the social network graph is transformed into a vector, offers promise in this context (Goyal & Ferrara, 2018). Different forms of embedding are available, including node embedding, edge embedding, and whole graph embedding. The selection of the embedding option depends on the objectives of machine learning techniques. In this research, we are interested in inferring the characteristics of users and making link predictions based on those inferences. Accordingly, we employ node embedding, wherein a node in the graph (a specific user) is converted to a vector representation that can be processed by machine learning algorithms. There are several techniques for node embedding. However, they tend to be static in nature and need to be recomputed whenever the social network structure changes, i.e., when users join, leave, or alter their relationships in the social network, and are ineffective for generating interpretable embedding factors.

The goal of this study is to address this gap by introducing a new algorithmic approach to extract user interests and preferences from the social network structure, which is robust and able to tolerate incremental changes in the network. Using the concept of homophily from social science, and a graph-based representation of the social network, we extract actionable user preferences that can be used for targeted advertising. Our proposed algorithm is

[1] Accuracy and recognition metrics in data analytics and computer science literature are equivalent to precision and recall, respectively.

consistent with recent node-embedding research (Perozzi et al., 2014: Grover & Leskovec, 2016) and aims to transform nodes, edges, and graph features into a low-dimensional vector space that allows the application of traditional machine learning approaches to graph data. The main advantage of our work is that it is capable of extracting meaningful dimensions from the structure of online social networks in a manner that facilitates application in preference-based recommendation systems. To demonstrate the effectiveness of our method, we empirically analyze a social network of more than 32,000 Twitter users. We then show that our proposed algorithm outperforms other node embedding approaches in providing friendship recommendations. We characterize this approach as homophily-based user embedding (HUE).

The rest of the paper is organized as follows: The next section provides an overview of the literature on node embedding approaches and highlights the limitations of existing methods. Next, we introduce our homophily-based user embedding approach, outlining different ways in which it can be applied to recommendation systems. Empirical application of the HUE approach is presented in the following section. We then demonstrate the application of our user embedding method in a link prediction task for a real data set of users. A discussion of the findings and an assessment of managerial and research implications conclude the paper.

2 Literature Review

Extracting underlying meaning from a graph typically relies on dimensional reduction techniques. The main goal of dimensional reduction is to reduce the size of the data by eliminating noise and less concise information, thereby preserving the salient information in the network. Early forms of dimensional reduction are manifest in techniques like multidimensional scaling, which typically seeks to coalesce attributes to enable comparison among competing alternatives. A number of methods have been devised, including IsoMap (Tenenbaum et al., 2000), Laplacian eigenmap (Belkin and Niyogi 2001), and LLE (Roweis & Saul, 2000). These techniques rely on features of the observations, e.g., geometric distance or k-nearest neighbors to produce relational graphs that can be mapped to lower dimensional spaces. These techniques often rely on eigenvector computing for dimension reduction. While these approaches are appropriate for structured data sets, social networks are different and require other techniques for reducing complexity. Matrix factorization (Ahmed et al., 2013) and nonnegative matrix factorization (Lee & Seung, 2001) are other types of dimension reduction approaches that can be applied to the adjacency matrix of the graphs. Nonnegative matrix factorization has also been used in literature for the extraction of homophilic features in online social networks (Shi & Whinston, 2013). However, matrix factorization approaches cannot preserve the global structure of the graphs, as they only model the dyadic relation between nodes in the adjacency matrix. In addition, they are also susceptible to computational complexity issues, particularly for large graphs. With the emergence of deep learning models in recent years, new methods have arisen in the field of graph embedding that are designed to retain graph features. We classify these approaches into three areas: (1) node embedding, (2) edge embedding, (3) entire graph embedding. In this study, our main focus is on node embedding methods. See Cai et al. (2018) for a comprehensive review of the literature.

The main objective of node embedding is to embed graph nodes in a way that preserves the similarity of the nodes in the form of first-order and second-order proximities. First-order proximity captures the closeness between a pair of nodes, e.g., the existence of a direct link between nodes, or the strength of this relationship, if available. Second-order proximity seeks to capture similarity on the basis of common neighbors. Both measures rely on the structure of the network for their computation. Research has shown that higher-order proximities have a positive impact in the computation of node embeddings (Cao et al., 2015).

Node embedding strategies can be divided into two categories based on their search strategies. DeepWalk (Perozzi et al., 2014) and Node2Vec (Grover & Leskovec, 2016) are two prominent algorithms that use random walk techniques. When employing a random walk technique, the idea is to generate a number of node sequences that represent alternative paths between a pair of nodes. These sequences are generated using a random walker that traverses the graph from different starting points.

DeepWalk (Perozzi et al., 2014) is one of the most popular algorithms in the field and uses the hierarchical Softmax technique to estimate the embedding vectors from the random walk node sequences. The algorithm, however, does not provide control over the generation of random walk sequences. Node2Vec (Grover & Leskovec, 2016) covers this deficiency and uses a generalized version of random walk that gives control over the generation of node sequences.[2] Additionally, it uses a more efficient approach called negative sampling to estimate embedding vectors.

[2] Node2Vec uses two hyperparameters *p* and *q* that control how the graph is traversed by a random walker. A value of $q>1$ leads to breadth-first sampling (BFS) and a value of $q<1$ leads to depth-first sampling (DFS). Parameter *p* controls whether sampling occurs locally around the target node.

Models without random walk apply deep learning techniques to the matrix representing the graph in order to preserve proximity among nodes (Niepert et al., 2016, Wang et al., 2016). There are several options available for deep learning applications. LINE (Tang et al., 2015) is one of the most prominent algorithms in this collection. It specifies two conditional and empirical distribution functions for context nodes and applies the KL-divergence difference between the two distributions to compute the loss function in the deep learning strategy. Separate loss functions are defined for first-order and second-order proximities.

Despite the strengths of previous models, we argue that there are weaknesses that limit their application to online social networks. First, embedded features are latent and noninterpretable variables, meaning that while they can be used to determine a user's structural similarity to other users or to community membership, they do not provide interpretable values such as preferences to analyze user behavior. Second, they suffer from a practical issue regarding the accommodation of new users joining the network. This requires recomputing the embeddings for the entire network and making the recommendation of new links infeasible. In this study, we plan to address both of these shortcomings using the concept of homophily.

The homophily concept suggests that individuals have a strong tendency to interact with people who have similar attributes rather than people with different attributes (McPherson et al., 2001). Homophily has roots in a variety of demographic and psychographic attributes (Gu et al., 2014). Some of these attributes are fixed and immutable, e.g., race, while others may change over time, e.g., attitudes and preferences (Li et al., 2013). There are several factors that shape the formation of homophilic relationships: (1) it increases the chance of being liked by others, (2) it makes it easier for individuals to get confirmation from other similar individuals, (3) the ongoing cost of maintaining relationships with similar others is lower than with dissimilar others because of the ease of developing trust and solidarity with them, and (4) individual relationship choices are frequently constrained by factors such as geographical locations, neighborhoods, work locations, and schools. These constraints lead to homogenous choices of relationships by individuals in a social network (Kossinets & Watts, 2009; Gu et al., 2014). As a result, the structure of a social network potentially contains a large number of latently embedded attributes of members within that network. Extracting these attributes can provide significant insight into the pattern of relationships in online social networks. However, correctly extracting these patterns can be challenging.

3 Homophily-Based User Embedding (HUE)

We develop our node embedding approach using the concept of homophily in social science. The guiding principle behind the HUE algorithm is to take advantage of second-order proximity at the community level and to represent the members of the social network through their connectivity pattern using smaller samples of selected members. To accomplish this, we introduce a new proximity measure called the ego's alter-network structural similarity, which is used to capture second-order proximity. In the following section, we first explain the ego's alter-network structural similarity calculation and the reasoning behind it, and then present our algorithm.

3.1 The Ego's Alter-Network Structural Similarity

To understand the principle of structural similarity between the ego's alter networks, we need to review the concepts of the ego network and the ego's alter network in graph theory. For each vertex v_i in graph $G(V,E)$, where V is the list of vertices (nodes) and E is the list of edges (links), there is a subgraph $G_{v_i}(V',E')$ where $V' = \{v_j \mid d(v_i,v_j) \leq 1, v_j \in V\}$[3] and $E' = \{(v_a,v_b) \mid v_a, v_b \in V', (v_a,v_b) \in E\}$. This subgraph in graph theory is referred to as the ego network of vertex v_i. In essence, the ego network of a specific vertex is a subgraph that includes all nodes connected to this vertex and any edges among them. Removal of the ego from its own ego network forms another subgraph $\widetilde{G_{v_i}}(V'',E'')$ where $V'' = \{v_j \mid d(v_i,v_j) = 1, v_j \in V\}$ and $E'' = \{(v_a,v_b) \mid v_a, v_b \in V'', (v_a,v_b) \in E\}$. We call this new subgraph the ego's alter network. Figure 1 shows examples of the ego network and the ego's alter network in a randomly generated graph.

Figure 1b shows an ego network (G_A), where A is the ego and other vertices are alters. Removal of A from its ego network keeps only the alters in the network and forms the ego's alter network ($\widetilde{G_A}$), as illustrated in Figure 1c. Figure 1d depicts the ego network for B, and Figure 1e represents its ego's alter network.

The similarity of the two ego's alter networks $\widetilde{G_A}$ and $\widetilde{G_B}$ show how the structure of neighbors for vertices A and B are similar. The following function presents the computation of such similarity (Johnson, 1985):

$$Sim(\widetilde{G_A},\widetilde{G_B}) = \frac{\left(|V(\widetilde{G_A},\widetilde{G_B})| + |E(\widetilde{G_A},\widetilde{G_B})|\right)^2}{\left(|V(\widetilde{G_A})| + |E(\widetilde{G_A})|\right) . \left(|V(\widetilde{G_B})| + |E(\widetilde{G_B})|\right)},$$

where $|V(G)|$ returns the number of vertices and $|E(G)|$ returns the number of edges in graph G.

[3] d is a network distance between vertices in the graph.

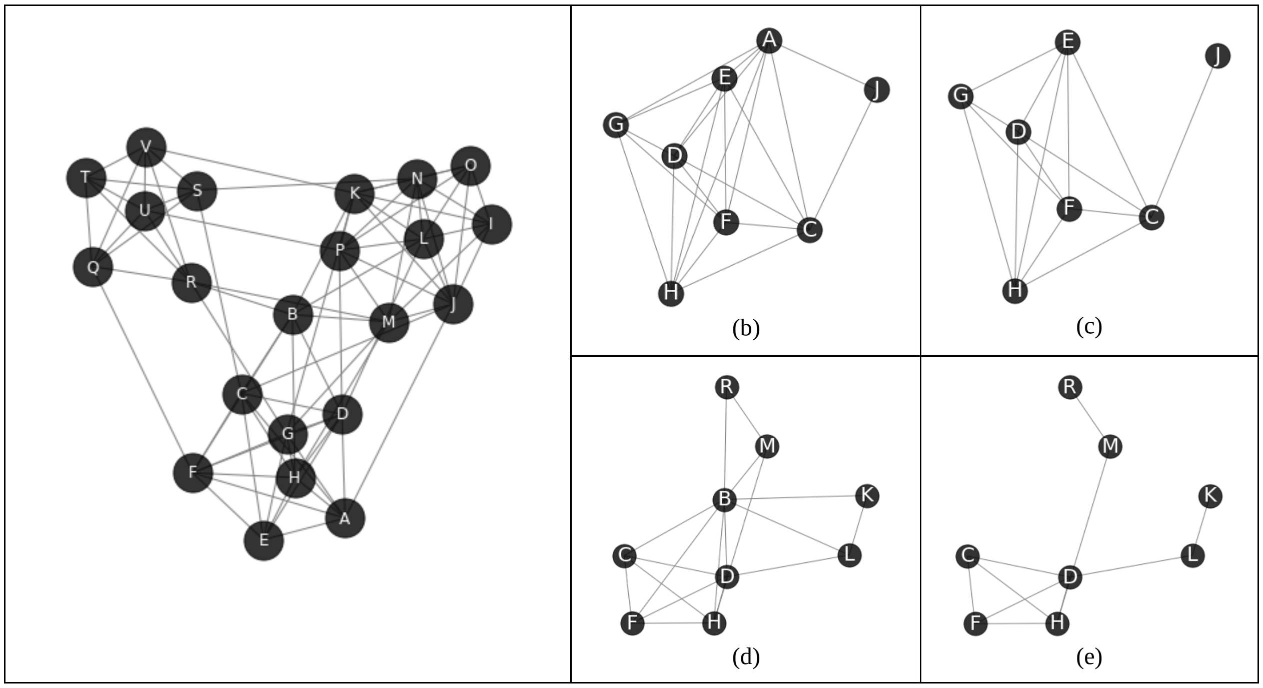


Figure 1. Ego Networks and Ego's Alter Networks

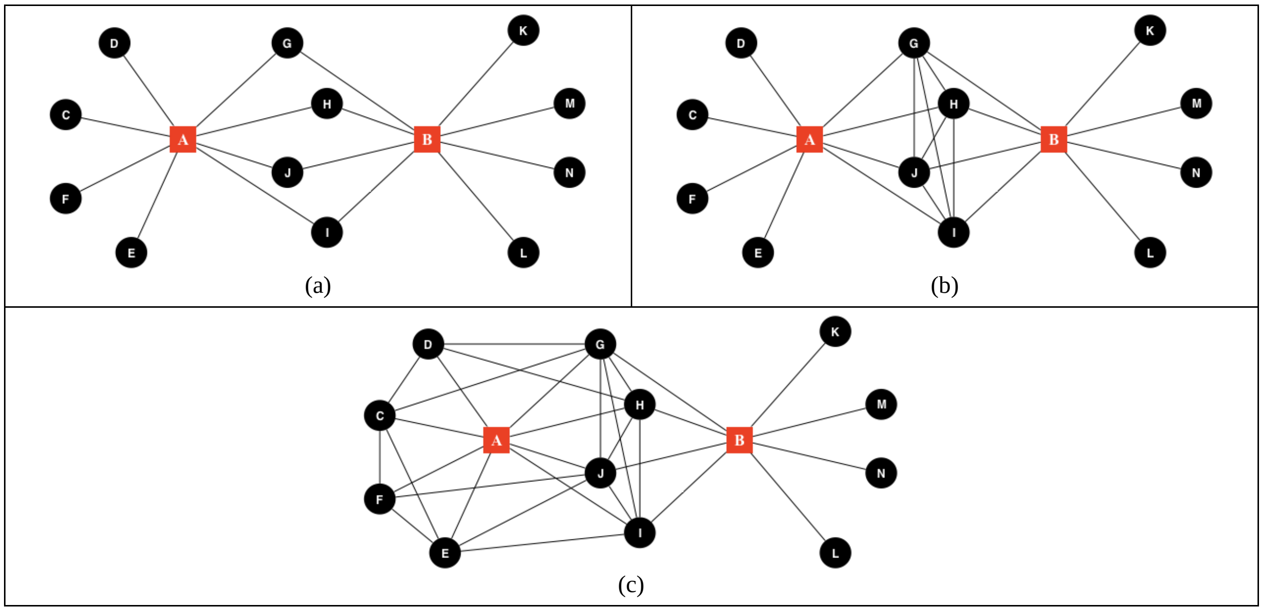


Figure 2. Ego's Alter Network Structural Similarity

$|V(\widetilde{G_A}, \widetilde{G_B})|$ and $|E(\widetilde{G_A}, \widetilde{G_B})|$ are the number of common vertices and edges between $\widetilde{G_A}$ and $\widetilde{G_B}$, respectively. This function considers both the numbers of common neighbors and their relationship in the computation of similarities. It has a range of 0 to 1. One of the main advantages of this similarity function over other second-order proximity functions is that it not only captures similarity by using the number of common neighbors but also incorporates the connection pattern between neighbors. It is this property of the function that is crucial to preserving the community structure of vertices. For example, consider the structural situations presented in Figure 2.

In all the scenarios depicted in Figure 2, using the number of common neighbors as a second-order proximity leads to the same value of similarity for vertices A and B. However, the above structural situations clearly show different community structures around A and B. In Figure 2b, there is a central community that both A and B are well connected to.

Therefore, it is expected that the similarity of vertices A and B will be higher for the scenario illustrated in Figure 2b in comparison with the one in Figure 2a, as they share a common community. This feature can be captured using the ego's alter-network structural similarity. In Figure 2c, we see a shift in the community structure, where Vertex A forms the core of a large community, and Vertex B is peripheral to it. In this scenario, the ego's alter-network structural similarity reduces the similarity of two vertices (compared to the graph in Figure 2b) as Vertex A is more representative of the community than Vertex B.

3.2 User Embedding Algorithm

The proposed embedding algorithm in this study comprises five steps: (1) selection of core vertices, (2) formation of the extended bipartite graph, (3) network simplification, (4) core clustering, and (5) measurement of the embedding features. We describe each step in detail.

Step 1: Selection of Core Vertices. As part of the development of our homophily-based embedding algorithm, we assume that the entire graph can be represented by n vertices. n is a hyperparameter that needs to be set but, generally speaking, increasing the value of n to a certain threshold tends to improve the overall performance of the algorithm. The threshold is a function of the size of the graph. It should be borne in mind that increasing n entails higher computational costs for the algorithm. From now on, we refer to these n representative vertices as the core vertices. The selection of the core vertices can be performed randomly or based on a specific criterion such as degree centrality of the vertices. Having a specific criterion in the selection of core nodes assists in the creation of meaningful embedding features. Figure 3 shows a number of randomly selected core vertices in a graph.

Step 2: Formation of the Extended Bipartite Graph. The next step is to extract the core vertices from the graph and form an extended bipartite graph. An extended bipartite graph is a type of network that consists of two separate graphs: (1) the original graph without the core vertices, and (2) a bipartite network of the core vertices and the other set of vertices in the graph. Figure 4 shows a schematic view of an extended bipartite graph generated from a graph.

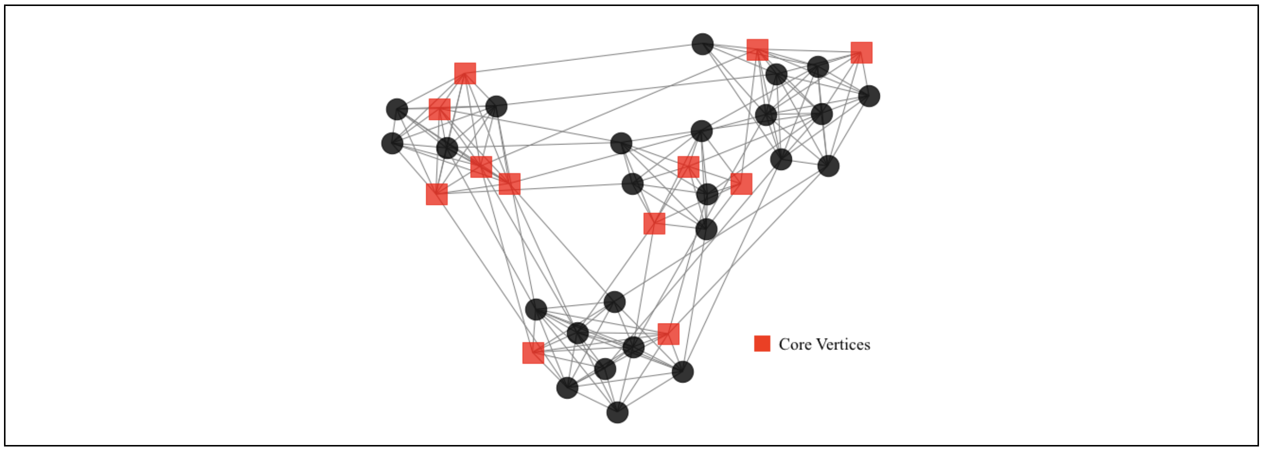


Figure 3. Randomly Selected Core Vertices

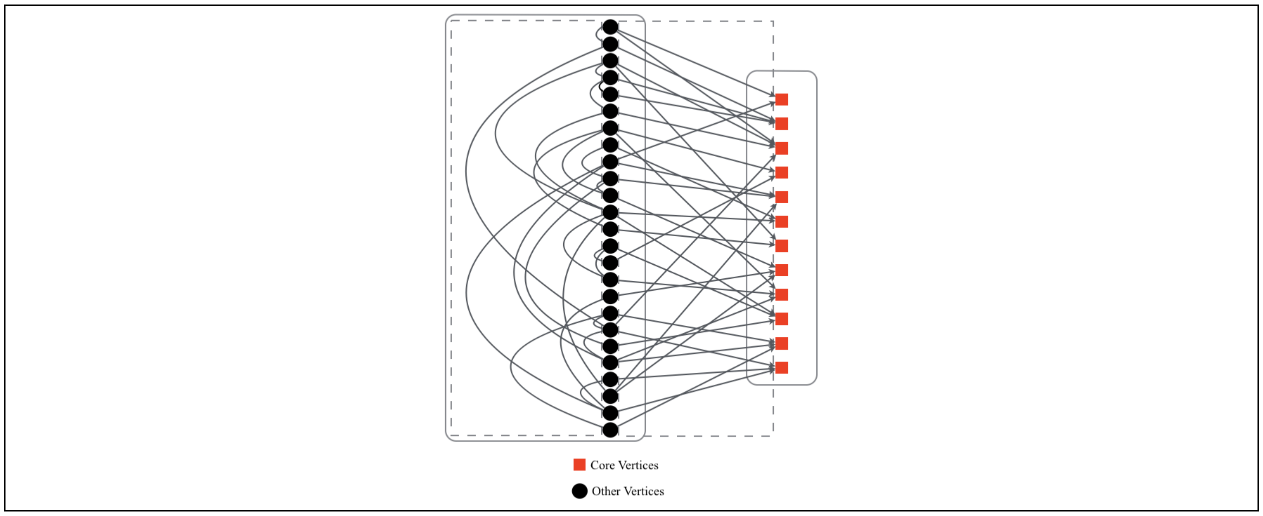


Figure 4. Extended Bipartite Graph

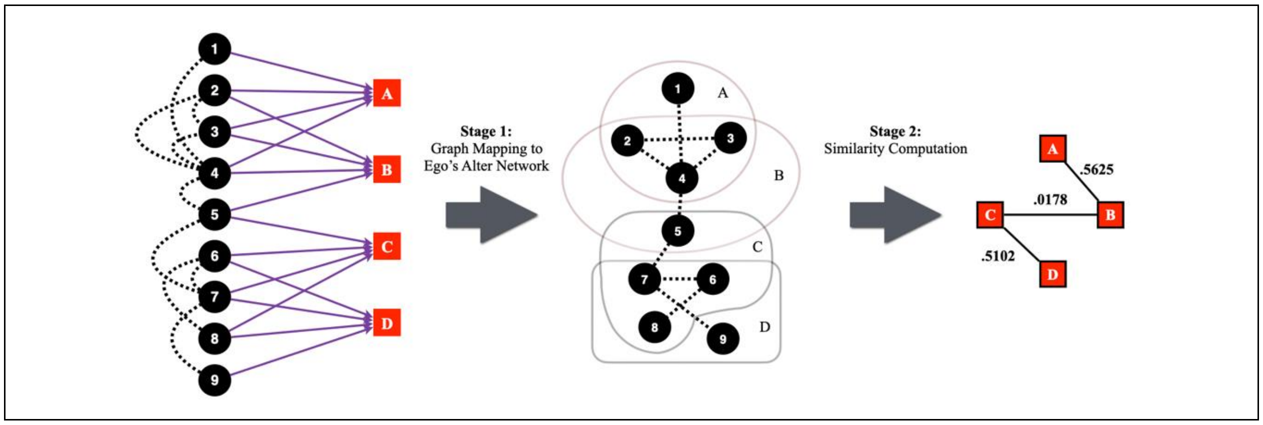


Figure 5. Network Simplification

In this step, the creation of the extended bipartite graph facilitates the extraction of the ego's alter networks and the computation of the similarity between those networks through matrix operations.

Step 3: Network Simplification. Our main purpose in selecting the core nodes was to select a set of vertices that represent different parts of a graph. To do so, we convert the extended bipartite graph to a weighted graph of the core nodes, where the weights represent the similarity of the alter network of the cores. Figure 5 depicts the process of converting an extended bipartite graph to a simplified weighted graph.

Step 4: Core Clustering. The next step is to agglomerate core vertices through the use of clustering methods. The idea behind this clustering step is to cluster the core vertices that represent the same part of the graph. In this study, we use a modified version of the Louvain algorithm (Blondel et al., 2008) for network clustering. Louvain originally developed the algorithm to detect clusters within graphs by seeking to optimize the modularity in a graph. It uses a greedy approach that iteratively optimizes the value of modularity by assigning nodes to different clusters. Modularity (Newman & Girvan, 2004) is a measure of how the structure of multiple clusters in a network is different from a random graph. While this property is statistically appealing, empirical studies show that modularity-based algorithms suffer from resolution convergence and cannot easily identify small clusters within large graphs (Fortunato & Barthelemy, 2007; Aldecoa & Marín, 2013; Traag et al., 2013). An alternative measure of cluster quality termed "surprise" has been proposed to counter the limitations of modularity in graph clustering (Aldecoa & Marín, 2011). Surprise assumes a null model in which edges emerge between nodes randomly. It then measures the deviance of the observed partition from the expected distribution of nodes and links into clusters given that null model (Aldecoa & Marín, 2011). Using this approach, an optimal set of clusters can be detected in a binary (nonweighted) network by maximization of the following objective function:

$$S = -log\left(\sum_{i=p}^{\min(M,n)} \frac{\binom{M}{i}\binom{F-M}{n-i}}{\binom{F}{n}}\right)$$

where F is the maximum possible number of edges between nodes, n is the observed number of edges, M is the maximum possible number of intracluster edges in a given cluster, and p is the total number of observed intracluster edges in that cluster. However, the optimization of this objective function is challenging in large networks. Using an asymptotic approximation of S can facilitate the optimization procedure by assuming that, as the graph grows, the relative number of intracluster edges ($q = \frac{p}{n}$) and relative number of expected intracluster edges ($r = \frac{M}{F}$) remains fixed (Traag et al., 2015). This approximation can be presented in the following manner:

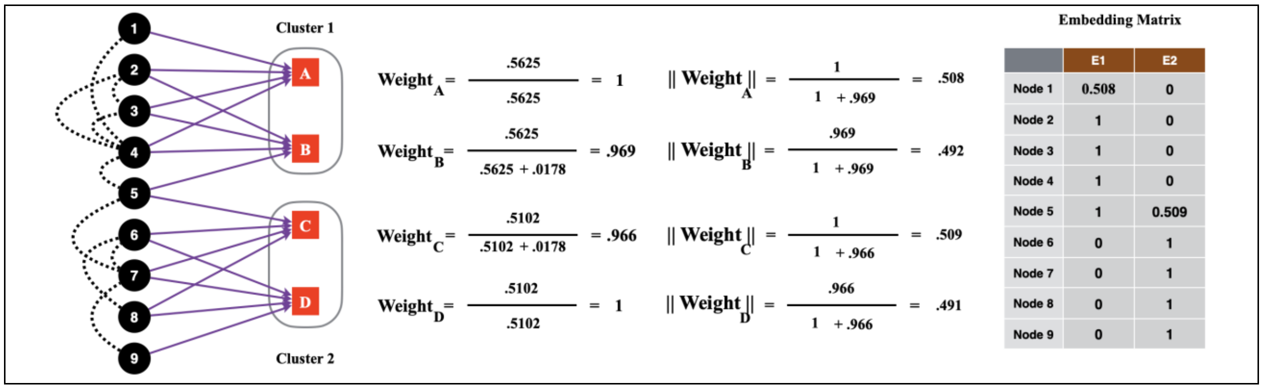
$$S^{\wedge} \approx nD(q||r)$$

where D is the Kullback-Leibler divergence loss function (Kullback & Leibler, 1951) that measures the distance between a two-probability distribution. It can be computed as follows:

$$D(q||r) = q \times ln\left(\frac{q}{r}\right) + (1-q) \times ln\left(\frac{1-q}{1-r}\right)$$

For weighted graphs we can simply change q to $\frac{p_w}{n_w}$, where p_w represents the total weights of intracluster edges and n_w is total weights of edges in the graph. There is no change to r in weighted graphs. It was shown that this asymptotic approximation of surprise can successfully capture clusters within large graphs (Traag et al., 2015).

In our method, we use the Louvain greedy approach (Blondel et al., 2008) to optimize the weighted version of asymptotical surprise function for the purpose of network clustering. Applying the algorithm to the graph in Figure 5 forms two clusters (C1 = {A, B}, C2 = {C, D}).


Cluster 1
Cluster 2
1
2
3
4
5
6
7
8
9
A
B
C
D
$Weight_A = \frac{.5625}{.5625} = 1$
$Weight_B = \frac{.5625}{.5625 + .0178} = .969$
$Weight_C = \frac{.5102}{.5102 + .0178} = .966$
$Weight_D = \frac{.5102}{.5102} = 1$
$\|Weight_A\| = \frac{1}{1 + .969} = .508$
$\|Weight_B\| = \frac{.969}{1 + .969} = .492$
$\|Weight_C\| = \frac{1}{1 + .966} = .509$
$\|Weight_D\| = \frac{.966}{1 + .966} = .491$
Embedding Matrix

	E1	E2
Node 1	0.508	0
Node 2	1	0
Node 3	1	0
Node 4	1	0
Node 5	1	0.509
Node 6	0	1
Node 7	0	1
Node 8	0	1
Node 9	0	1

Figure 6. Node Embedding

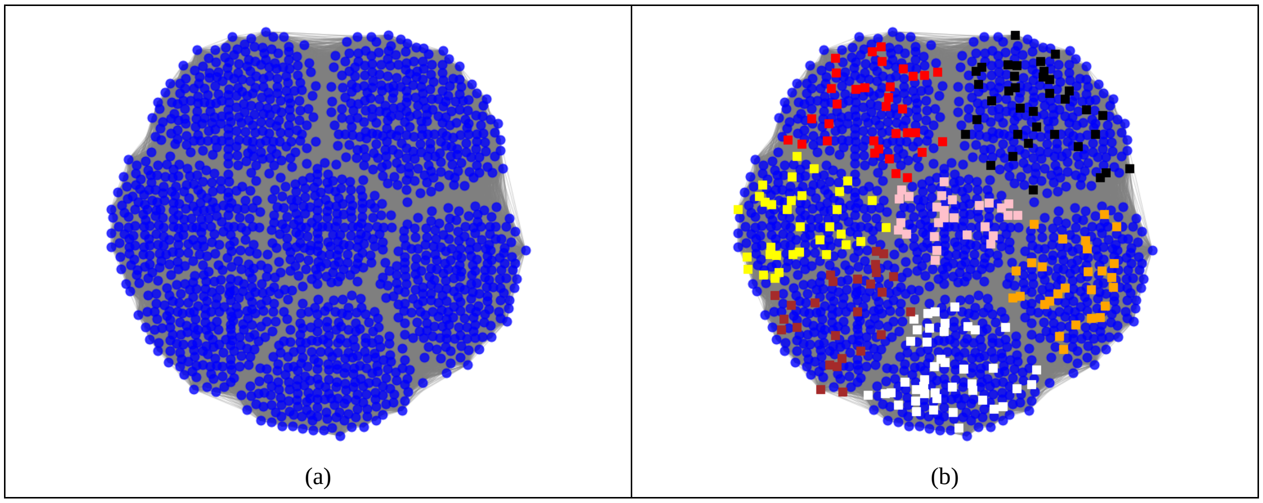


Figure 7. Random Generated Graph Clustering

Step 5: Measurement of the Embedding Features. The dimension of the embedding matrix is determined by the number of clusters identified in the previous step. Vertices in the graph can be represented by their normalized weighted value of their level of connectivity to core vertices in each of the above clusters. The weights are assigned at the core level and for a given core vertex $A \in C_i$ the weight is computed as follows:

$$Weight_A = \frac{\sum_{\forall B \in C_i} Sim(\widetilde{G_A}, \widetilde{G_B})}{\sum_{\forall N} Sim(\widetilde{G_A}, \widetilde{G_N})}$$

The weights are then normalized at the cluster level such that $\sum_{A \in C_i} ||Weight_A|| = 1$. The weighting procedure assigns a greater weight to core vertices that are positioned near the center of their cluster and are more representative of the group of vertices in the cluster. Figure 6 shows an example of this computation for the graph that was initially presented in Figure 5. While the example in Figure 6 shows the computation only for noncore members, this computation can also be used to capture the level of connectivity of cores to different clusters. The pseudocode for the HUE algorithm appears in Appendix A.

3.3 Analysis of HUE Properties Using a Random Generated Graph

Before we illustrate the application of the proposed algorithm in online social networks, we first demonstrate the properties of the proposed algorithm in a randomly generated graph. The generated graph contains 1,500 vertices in seven clusters, where the probability of the formation of inter-cluster links is 0.4 and the intracluster is 0.1. Figure 7a shows this randomly generated graph.

To capture the node embedding features of the graph, we randomly selected 225 core vertices (15% of whole data) and applied the HUE algorithm. Figure 5b shows the result of the core clustering step for our experiment. It is clear that core vertices represent the different parts of the graph well. Next, we used the identified core vertices to measure the node embedding metrics for all the vertices in the graph. In order to see the representativeness of embedding features, we applied the TSNE dimension reduction approach and projected the embedding features to two dimensions. Figure 8 visualizes the projection result.

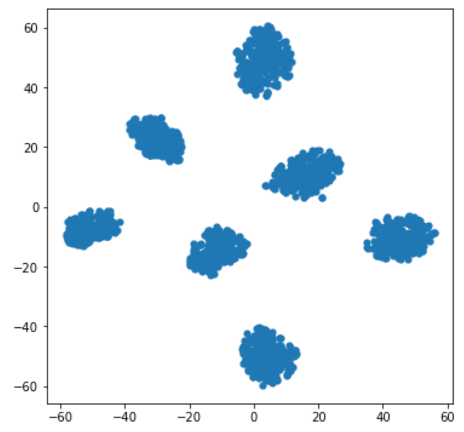


Figure 8. TSNE Dimension Reduction for Randomly Generated Graph

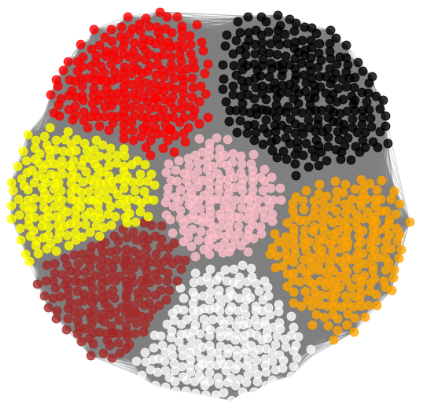


Figure 9. K-Means Clustering in Randomly Generated Graph

Figure 8 shows that the embedding features preserve the seven-cluster structure of the original graph. Then we applied K-means to the embedding features, and set the number of clusters to seven. Figure 9 shows the outcome of the clustering task with K-means.

This clearly shows that a small number of core vertices allows quantification of the entire graph in a rather effective manner. What makes our algorithm appealing for the analysis of larger social networks is that we can use a small fraction of the vertices in the graph to compute the similarity between core vertices. The calculation of the embedding features can then be performed for all nodes on the basis of their pattern of connectivity to the core vertices. Our analysis shows that by using as few as 30% of the vertices in the graph we can achieve reasonably good core clustering. The same idea applies to the addition of new nodes to the graph. As long as the addition of new nodes does not change the overall structure of the graph, it is possible to use the identified core vertices for the calculation of the embedding features of new nodes without any additional requirement for the reestimation of core clusters.

4 User-Embedding in Online Social Networks

The proposed node-embedding approach provides an effective mechanism to analyze graphs representing users in online social networks. In this study, we show that the node-embedding algorithm can be used not only to embed user preferences but also for link prediction. The key factor is to select the core vertices based on a specific criterion, rather than relying on random selection. According to selective exposure theory (Sears and Freedman 1967, Zillmann and Bryant 1985, Zillmann 1988, Huang et al., 2013), people have a tendency to expose themselves to mass communication channels that reinforce their own views and are in agreement with their own preferences and thinking. Therefore, it is expected that while interacting in an online social network, individuals will follow social pages[4] that promote their

[4] A social page refers to an online social network account related to an organization, brand, celebrity, program, news agency or other popular entity that attracts an individual's interest.

own views and are consistent with their preferences. Additionally, social pages in online social networks have high levels of in-degree centrality, making them good choices for core vertices. Prior research in the IS field has considered users with a high level of in-degree centrality as thought leaders in social networks (Faraj et al., 2015). Relying on these assumptions, social pages in online social networks can effectively represent groups of users in the social graph who share similar preferences.

4.1 Empirical Application

In order to demonstrate the utility of the HUE method, we applied it to a real-world data set of users on Twitter. Twitter is a popular social media platform for targeted advertising. The unique structure of Twitter allows for the identification of new trends and the targeting of individuals in real time. However, the real power of this analysis comes from linking and embedding other social network data, e.g., Foursquare or Instagram data, into Twitter content. This provides a set of linked data on different dimensions that affords greater insight into the character of the users. We show that the extracted embedding factors from the network structure of Twitter users can represent individual preferences toward a specific topic of interest that can also represent their check-in behavior. Inferences of this information about users can facilitate the creation of customized messaging and advertising to these individuals, allowing marketers to generate more precisely targeted campaigns, thereby reducing costs associated with wasted promotion efforts. It can also be incorporated into recommendation systems directly for friend recommendations (as shown later in the HUE Link Prediction application) or indirectly by computing the similarity between users over their embedding features and using them in user-to-user collaborative filtering.

4.2 User Population and Data Set

To conduct the empirical application, we used a data set of more than 32,000 individuals across the US who shared their location-based Foursquare check-ins on the Twitter platform. The data set spans a six-month period in early 2014 and contains a social network of users, along with the pattern of social pages they followed on Twitter (for this study we only considered the top 10,000 social pages and used them as core vertices). Details of the attributes included in the data set appear in Table 1. Sparsity measures the extent of interconnectedness among users relative to a fully connected network. At 0.038%, it indicates that users were reciprocally connected to relatively few other users, with an average of six connections each. A similar measure of sparsity between users and social pages was derived, and this was more densely connected at 3.18% (an average of 159 social pages followed). While metrics based on the average number of links are easy to grasp, sparsity metrics are size independent and provide a more meaningful basis for comparison.

4.3 Identification of Core Clusters

As discussed earlier, we consider the social pages to be the core vertices of the social graph that represent different parts of the graph and show individual preferences toward certain topics. As a result, the extended bipartite graph in HUE consists of two parts: (1) a social network of users with established links between users,[5] and (2) a bipartite network with one-way links from the social network users to social pages. HUE clusters social pages based on the similarity of their alter networks.[6] In this context, a cluster of similar social pages (core vertices) forms a community of interest. HUE assigns weight to each social page to show how well a specific community of interest can be represented by a given social page.

To assign meaningful labels to identified communities of interest, we used all tweets from social pages gathered over the six-month period and extracted representative words that were commonly used on social pages within the same community of interest.[7] We also manually checked the description of social pages to obtain additional insights about the communities of interest. Coupling the above set of information, we created a label for each of the communities. Figure 10 shows word clouds of representative words for a sample of communities.

A detailed analysis of communities shows that while some communities represent general preferences of individuals in terms of music or fashion, for example, others have a much narrower focus—such as a TV show or a brand. Appendix C shows the list of prominent communities of interest that were identified along with some samples of social pages within those communities.

We sought to visualize the position of social pages (core vertices) in the social graph by structurally positioning them where they have a larger number of followers. We used the ForceAtlas2 layout (Jacomy et al., 2014) in Python to accomplish this task.

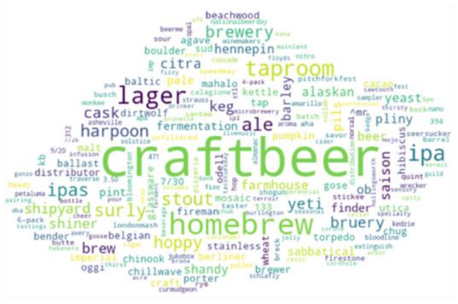
[5] In this research, we used reciprocated relationships between users to form the social network portion of the graph. A reciprocated relationship is formed between two users when both users follow each other in the network. That is an indication of a strong relationship between the two users.

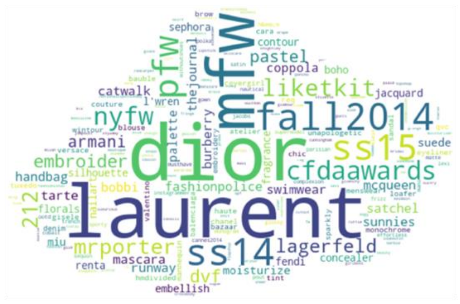
[6] The clustering method used in HUE is a greedy agglomerative approach that can produce a slightly different result in each run. Accordingly, we ran the algorithm 500 times to select the one with the best quality.

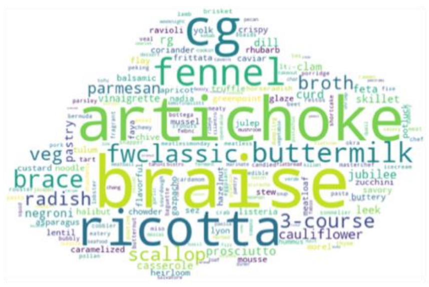
[7] Appendix B describes the process of selecting representative words.

Table 1. Data Set Characteristics

Number of users	32,722
Number of directed links	283,893
Number of reciprocated links	201,796
Sparsity among users	0.038%
Average number of links to other users	6.17
Number of social pages	10,000
Number of links to social pages	5,208,414
Sparsity with social pages	3.18%
Average number of links to social pages	159.17


(a) Beer community of interest


(b) Fashion community of interest


(c) Food community of interest

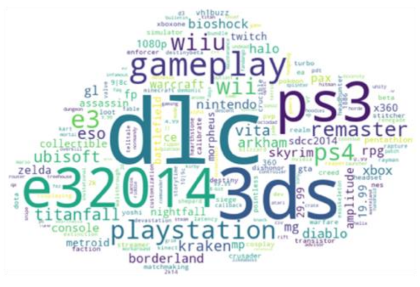

(d) Electronic games community of interest

Figure 10. Word-Cloud of Representative Words for Sample Communities

ForceAtlas2 uses a force-directed layout where nodes repulse each other and edges attract the nodes they are connected to toward each other (Jacomy et al., 2014). Figure 11 visualizes the distribution of social pages as a result of this process. Gray nodes depict individuals in this figure and colored nodes indicate social pages in different communities of interest. Given the large number of social pages in this study, visualization of all data simultaneously proved challenging and we thus selectively depicted certain communities only. Visualization of a single community provides little information beyond the followers of a particular set of pages in that community. However, the visualization of related and complementary communities of interest provides far greater insights. Individuals who are structurally positioned close to social pages belonging to a community of interest have a greater affinity toward the community of interest. In addition, social pages that appear on the periphery of the graph reflect a niche preference among a group of individuals in the social network. Reading across the figures affords a clearer interpretation of individual preferences amid communities of interest, showing, for example, that individuals with a strong interest in fashion are less likely to have strong beer preferences and vice versa.

Next, we measured the embedding features associated with each of the communities of interest. To accomplish this, we computed the weighted attribute of each social page and then computed the embedding matrix. The embedding features provide meaningful values reflecting the relative interest of individuals toward different subjects of interest. Figure 12 shows the distribution of individual preferences to fashion, beer, and the two dominant political parties in the United States. The intensity of the colors shows the magnitude of individuals with a clear preference for that community. Two levels of insight can be gleaned from these figures. The obvious implication is that specific individuals have an affinity for a particular community. Marketers can utilize this to directly target individuals. A second level of inference relates to affinity among communities.

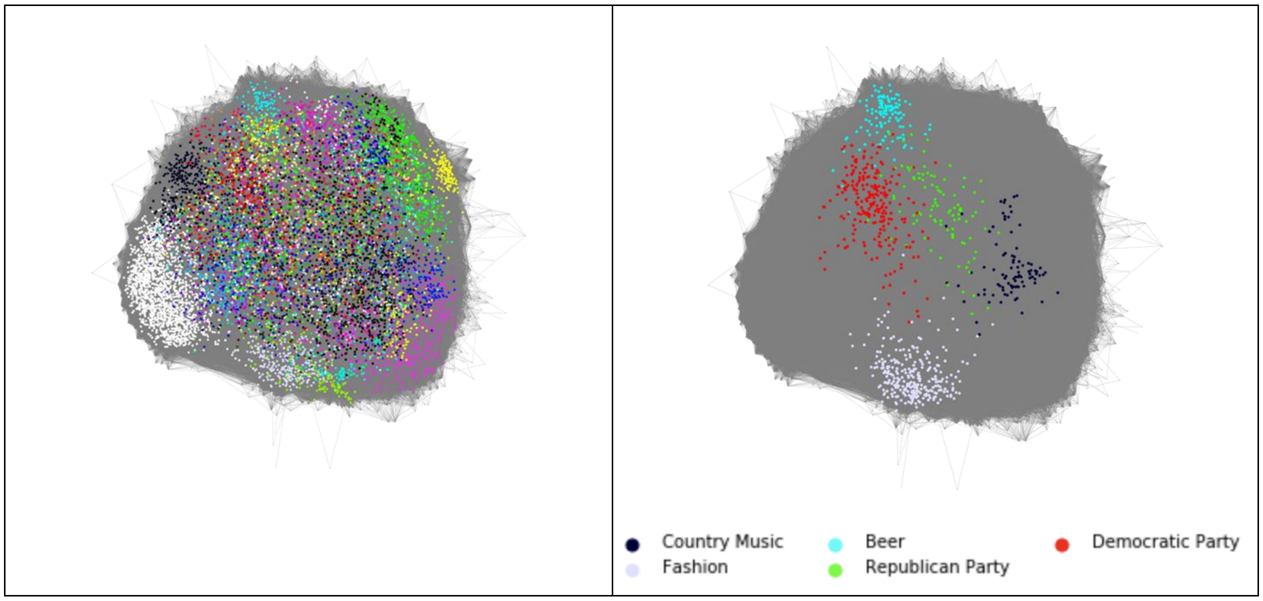
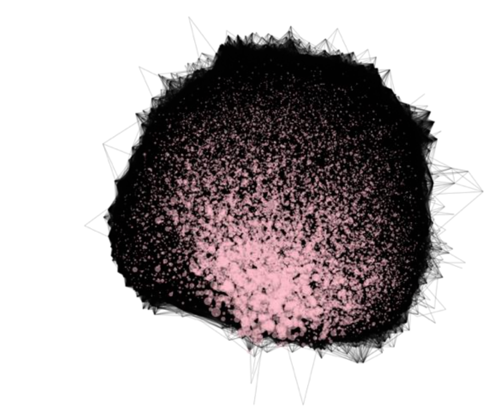
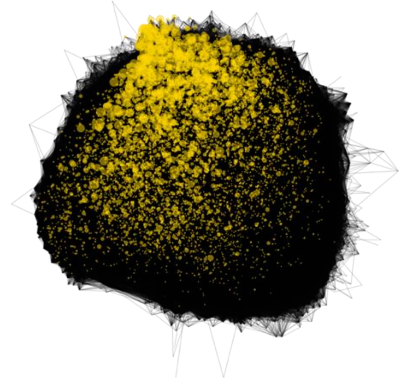


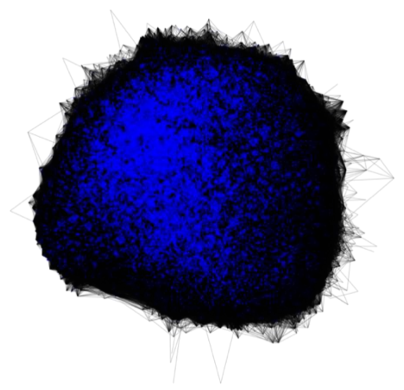
Figure 11. Distribution of Social Pages (Core Vertices)



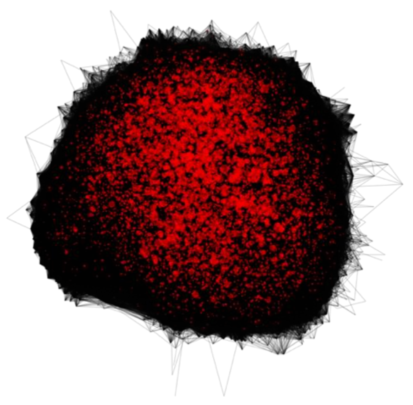
(a) Distribution of fashion preference



(b) Distribution of beer preference



(c) Distribution of Democratic Party preference



(d) Distribution of Republican Party preference

Figure 12. Examples of User-Embedding Feature Distribution in Online Social Network

Thus, for example, it can be deduced that there is little overlap of interest between the beer community and the fashion community. But there is considerably more overlap between the beer community and the political parties. Thus, this overlap or complementary nature can be exploited by marketers who can either cross-sell or target individuals based on their participation in a different but related community. Keep in mind that the individuals in these graphs were taken from the sample data set. New individuals would need to be embedded in the graph based on their interactions and the social pages they follow.

4.4 Explanatory Power of Identified Preferences

To check the validity of the preferences identified through HUE, we used the preferences as explanatory variables for the individual's check-in behavior. Accordingly, we considered the top 10 venue categories where users had the highest number of check-ins and then computed the total number of check-ins per venue category per user over the six-month data collection period. Next, to estimate our models, we used negative binomial regressions. Results are shown in Table 2.

Table 2. Preferences and Location-Check-in Behavior

Preference	**Bar**	**Airport**	**Café**	**Gym**	**Grocery store**	**Hotel**	**Fast food restaurant**	**Music venue**	**Church**
Alternative rock	-	-	-	-	-	-	-	2.018	-
Apple tech	-1.154	-	-	-	-	-1.424	-	-	-
Art & design	-	-	-	-	-	-1.252	-	-	-
Beer	1.209	-	-	-	-	-	-	-	-
Business & tech	-	2.980	-	-	-3.341	1.768	-1.478	-	-
Crossfit	-	-	-	1.961	-	-	-	-	-
Coffee	-	-	2.423	-	-	-	-	-	-
Comedy	-	-	-	-	-	-	-	-	-4.201
Computer programming	-	-	-	-	-1.780	-	-	-	-
Exercise & health	-	-	-	4.960	-	-	-.928	-	-
Fashion	-	-	-	-	-	-	-0.809	-	-
Fast food	-1.821	-2.550	-	-2.005	4.445	-	3.808	-	3.510
Hockey	-	-	-	-	-	-	-	2.217	-
Humor	-	-	-	-	-2.144	-	-	-	-
Las Vegas	-	-	-	-	-	2.223	-	-	-
Liquor	1.208	-	-	-	-	-	-	-	-
LGBT	-	-	-	-	2.036	-	-	-	-
Los Angeles	-	-	1.052	-	-	-	-	-	-
Music media	-	-	-	-	-	-	-	1.924	-
New York	-	-	-	-	-	-	-	2.367	-
Radio music	-	-	-	-	-	-	.872	-	-
Religion	-1.764	-	.987	-	-	-	-	-	7.874
Rock bands	-	-	-	-2.660	-	-	-	2.457	-
Running	-	-	-	1.990	-	-	-	-	-
San Antonio	-	-	-.916	-	-	-	-	-	-
San Francisco	-	-	.994	-	-	-	-	-	-
Social media & e-commerce	-	1.767	-	-	-	-	-	-	-
Spiritual content	-	-	-	-	-	-	-	-	3.505
Travel	-	5.064	-	-	-	3.169	-	-	-
Video games	-	-	-	-	-	-	-	-	-3.713
WWE	-	-1.484	-	-	-	-	-	-	-
Intercept		.737	1.582	1.530	.723	.607	1.546	.273	.051

Note:
1. All the reported coefficients are significant at the 0.001 level
2. We use all the preferences as independent variables, but only report the top five with the highest absolute coefficient values.

Since the number of preferences is a large number, we simply report the coefficients of the top five preferences with the highest absolute values for each check-in category. Since all preferences were normalized using the same scale, the absolute value of each coefficient can be interpreted as the importance of a specific preference in the explanation of the behavior.

The results reveal some expected relationships, e.g., a negative relationship between preference for fast food and the frequency of visiting gyms. However, some other relationships are fascinating and clearly novel, e.g., a positive relationship between preference for fast food and the frequency of visiting churches. Appealing though these relationships are, their exploration remains outside the scope of this paper, as we focus on the ability to link user preferences and user behavior.

4.5 Application of HUE in Link Prediction

One of the main applications of node embedding is link prediction (Grover and Leskovec 2016). Link prediction can be used to satisfy a number of different objectives. It can be applied to determine new links that will emerge over time—typically as a result of maturation within the social network. It can also be used to predict how new additions to the network will interact with the existing members of a social network. A third area of utility is the completion of missing parts of the network. Research has shown that predicting potential social relationships provides opportunities and benefits for decision makers in different fields including marketing (Cheng et al., 2015), healthcare (Almansoori et al., 2012; Dhouioui et al., 2016), and friendship recommendation systems (Xie, 2010; Aiello et al., 2012). Link prediction approaches generally fall into two categories: (1) similarity-based approaches and (2) learning-based approaches (Wang et al., 2015). In the former, a similarity measure is used to assign scores to every potential pair of nodes in the network. Then a ranking system is applied to assigned scores in descending order. A higher similarity score for a pair of nodes indicates a higher likelihood of that pair being linked. There are several measures of similarity that have been used for node pairs in social networks (see Wang et al., 2015 for a list of similarity-based measures). A learning-based approach, on the other hand, treats link prediction as a binary classification problem, and various supervised learning algorithms can be trained using node features in established social networks. These algorithms can then be used to predict new links in social networks. In this study, we conduct learning-based link prediction and use a deep learning model to illustrate the predictive power of embedding features.

4.6 Deep Learning Model for Link Prediction

In order to perform link prediction, we considered two-way relationships among users to form an undirected social graph of users, and randomly removed 30% of links from the social graph while ensuring that the residual network obtained after the edge removal was connected.[8] We used the remaining links as positive samples to train the deep learning models. We divided the removed links into two equal sets (each comprising 15% of edges) to be used as holdouts and test sets. In nearly all social networks, the number of links is a small fraction of all possible links since users interact with a small number of other users. Our data set was no exception. Accordingly, for each established link in training, holdouts, and test sets, we randomly selected 10 unestablished links and used them as negative samples in our work. This approach limits the ratio of established links to unestablished links to no worse than 1:10.

In order to generate node-embedding features, we applied our algorithm and three other prominent node-embedding models, namely DeepWalk (Perozzi et al., 2014), Node2Vec (Grover & Leskovec, 2016), and LINE (Tang et al., 2015) to the training social graph.[9]

We considered the 256 features extracted from the communities of interest with the highest number of core vertices. We also set the number of dimensions in other algorithms to be equal to 256. Since the performance of algorithms varies based on the hyperparameters selected, we used different sets of hyperparameters and compared their performance. Table 3 lists the hyperparameters we used for each of the algorithms.

For link prediction, we need to work at the edge level rather than at the node level. We followed Grover and Leskovec (Grover and Leskovec 2016) and used multiple binary operators to combine feature vectors $f(u)$ and $f(v)$ of two given nodes u and v to generate a feature vector $g(u,v)$ that represents a potential link e_{uv} between a pair of nodes. Table 4 represents the list of binary operators that we employed.

The binary operators generated the same 256 features at the edge level. We used the generated features of each operator as one set of predictors and ran a separate set of learning models for that set of predictors.

[8] While our algorithm can still perform the node-embedding task on disconnected networks, the reason for keeping the residual network connected is to be able to calculate the node-embedding features of the nodes using other algorithms and compare the predictivity power of them.

[9] We also used other algorithms such as SDNE (Wang et al., 2016) and Struct2Vec (Ribeiro et al., 2017); however, the performance of those algorithms was not as promising as other algorithms.

Table 3. Algorithm Specifications

Label	Algorithm	Settings	Description
DeepWalk	DeepWalk	Dimensions:256, Window_size:8, Num_walks:20, Walk_length:40	Use hierarchical softmax for estimation.
Node2Vec_V1	Node2Vec	Dimensions:256, Walk_length:40, Num_walks:20, Window_size:8, p: 1, q: 1	Performs random walk similar to DeepWalk. Use negative sampling for estimation.
Node2Vec_V2	Node2Vec	Dimensions:256, Walk_length:40, Num_walks:20, Window_size:8, p: 2, q: .5	Random walk approximate BFS walk. It encourages moderate exploration and avoids 2-hop redundancy in sampling.
Node2Vec_V3	Node2Vec	Dimensions:256, Walk_length:40, Num_walks:20, Window_size:8, p: 4, q: 2	Random walk approximate DFS walk. It encourages moderate exploration and avoids two-hop redundancy in sampling.
Node2Vec_V4	Node2Vec	Dimensions:256, Walk_length:40, Num_walks:20, Window_size:8, p: .25, q: .5	Random walk approximate BFS walk. It encourages local exploration around starting node.
Node2Vec_V5	Node2Vec	Dimensions:256, Walk_length:40, Num_walks:20, Window_size:8, p: .25, q: 2	Random walk approximate DFS walk. It encourages local exploration around starting node.
LINE_v1	LINE	Dimensions:256, Proximity-order: second-order proximity	Use only second-order proximity to estimate embedding factors
LINE_v2	LINE	Dimensions:256, Proximity-order: first- and second-order proximities	Generate 128 embedding factors using first-order proximity and 128 embedding factors using second-order proximity
HUE	HUE	Dimensions: 256, Num_core_vertices: 10000	Use top 256 communities of interest with the highest number of cores to compute embedding factors

Table 4. List of Binary Operators

Operator	Symbol	Definition
Average	$\boxplus$	$[f(u) \boxplus f(v)]_i = \frac{f_i(u) + f_i(v)}{2}$
Hadamard	$\boxdot$	$[f(u) \boxdot f(v)]_i = f_i(u) * f_i(v)$
Weighted-L1	$\|\| . \|\|_{\bar{1}}$	$\|\| f(u) . f(v) \|\|_{\bar{1}i} = \| f_i(u) - f_i(v) \|$
Weighted-L2	$\|\| . \|\|_{\bar{2}}$	$\|\| f(u) . f(v) \|\|_{\bar{2}i} = \| f_i(u) - f_i(v) \|^2$

In addition, to exploit the capability of multiple operators, we employed a combination of a pair of operators simultaneously. Since there is no a priori basis for the selection of this pair, we used all combinations of the four operators, for an additional six model comparisons. To develop our learning models, we relied on a five-layer, fully connected neural network (1 input, 3 hidden, and 1 output layer) with 128, 64, and 32 neurons, respectively, in the first, second, and third hidden layers. We used the Swish activation function (Ramachandran et al., 2017) for all hidden layers and the Sigmoid activation function for the output layer.[10] To avoid overfitting, we used drop-out in all hidden layers. Additionally, we applied batch normalization to increase the learning speed and reduce the impact of internal covariate shift. We used the Xavier method (Glorot & Bengio, 2010) for initialization of the weights and then the Adam optimizer with exponential decay learning rate for the estimation of weights in all models.

The base loss function for binary outcome variables is binary cross-entropy, specified as:

$$l_i = y_i . \log(\hat{y_i}) + (1 - y_i) . \log(1 - \hat{y_i}) \,,$$

where $y_i \in \{0, 1\}$ shows the ground truth value, and $\hat{y_i}$ is the output of the sigmoid function. In link prediction, we are mainly interested in users who are

[10] We also tried ReLU and Leaky ReLU as an activation function for a few samples of models in their hidden layers. In all cases, the models with the Swish activation function outperformed the other models.

inclined to be connected with other users. In our data set, even after controlling for the ratio of linked users to unlinked ones, the data set still remained imbalanced (1:10). Prior studies have suggested two approaches to tackling imbalanced data sets: (1) assign weights to classes (Ling & Sheng, 2010, Mirza et al., 2013), and (2) use a focal loss function (Lin et al., 2017).

In the first approach, the weighted binary cross-entropy loss function is given by:

$$l_i = W_1 . y_i . \log(\hat{y_i}) + W_2 . (1 - y_i) . \log(1 - \hat{y_i}),$$

where W_1 and W_2 refer to assigned weights to each of the classes. Adopting this strategy can be counterproductive. In some cases, important minority samples may be assigned higher weights in the loss function and eventually increase the misclassification cost for the model.

The second approach was offered recently by Lin et al. (Lin et al., 2017) as an alternative to offset this concern. The focal loss function is a reshaped version of binary cross-entropy loss. It assigns less importance to the loss of easy examples[11] in the training set, thereby mitigating the effect of imbalanced classes. In this case, the loss function is given by:

$$l_i = \alpha . y_i . \left(1 - \hat{y_i}\right)^{\gamma} . \log\left(\hat{y_i}\right) + (1 - \alpha) . (1 - y_i) . (\hat{y_i})^{\lambda} . \log(1 - \hat{y_i}),$$

where $\alpha \in [0,1]$, and can be specified using multiple strategies. One option is to use the inverse class frequency. One can also treat α as a hyperparameter and set it using cross-validation. γ represents a tunable hyperparameter that helps decrease the impact of easy examples. Research shows $\gamma = 2$ is a good choice for this hyperparameter (Lin et al., 2017). In this study, we used both weighted cross entropy and focal loss functions to train our learning models and compared the results.

We trained our models using the training set, reserving the holdout set for model selection. In doing so, we trained each of our models continuously. After each epoch,[12] we evaluated the performance of the model using the holdout samples to select the model with the lowest associated cost. We used an early stopping strategy in the model selection process and stopped the training process when the cost associated with the holdout set did not improve over 20 continuous epochs. Figure 13 shows an example of the model selection process.

We trained the neural network for different features by applying operators to each node-embedding algorithm with two different cost functions (weighted and focal cross-entropy) over five runs. This approach generated a total of 900 different models. Figure 14 shows the cost associated with the holdout set in different models. In the graph, the solid line indicates the average cost across the five runs, and the shaded areas around this provide a sense of the range of costs experienced for each configuration of the neural network. Though not immediately apparent on the graphs, the weighted cost was significantly higher, as noted by the values on the respective y-axes. This difference is due to the presence of additional polynomial factors in the focal loss function. Progression on the x-axes does not provide any semantic content, as these are simply different operators.

The result shows that the embedding features identified by our proposed algorithm result in the lowest costs across all operators and perform best when using both the WeightedL1 and Hadamard feature sets.

After the selection of models, we checked their performance using the test data sets. We evaluated the performance of models using precision, recall, F1, and AUC measures. Precision is a ratio of the number of actual connected links from the set of those predicted to be connected and is a measure of the positive predictive value of the predictor (or rule). Recall, on the other hand, examines all connected individuals, and determines the fraction of those that are correctly predicted as being connected by the model, and is an indicator of the true positive rate associated with the predictor or rule. The f-measure F1 is a harmonic mean of precision and recall.[13] Finally, AUC shows how well the model can distinguish between linked and unlinked groups of users. AUC is a more involved measure that represents the area under the curve of the *receiver operating characteristics* curve that relates the true positive rate and the false positive rate. It ranges between 0 and 1, with a score of 0 representing complete misclassification, 0.5 representing no ability to classify correctly, and 1 indicating perfect classification. The AUC metric is shown in Figure 15 for both the weighted binary cross-entropy and the focal cross-entropy cost functions. Results for the remaining metrics are available in Appendix D.[14] Once again, the solid lines represent the mean scores, and the bands represent the ranges for the five runs for each of the algorithms.

[11] Easy examples are those training samples that can be easily classified by the classifier.

[12] One epoch refers to one forward pass and one backward pass of all the training examples through the neural network.

[13] Precision, recall, and F1 measures are all computed at the threshold value of 0.5.

[14] We did not report accuracy, as accuracy provides biased measures of performance for imbalanced data sets.

The results show similar patterns for both weighted and focal cross-entropy. Since algorithm performance varies in different metrics, the AUC metric could be the best comparison metric for comparing the overall performance of algorithms in link prediction. The AUC is perceived to be a more robust measure of prediction in the presence of imbalance (Hanley & McNeil, 1982). Its value can be interpreted as the probability that a randomly selected missing link between users has a higher score than a randomly selected nonexistent link. The results for the AUC metric show that our proposed algorithm outperformed other algorithms when we used variables from the Hadamard operator and a combination of two operators (other than those from WeightedL1 & WeightedL2). The AUC value for the case in which we used both the Hadamard and WeightedL1 variables is the highest AUC of all possible cases with a value close to 0.95, indicating strong predictive power of our algorithm for link prediction. For the Average operator, our algorithm is in the second position, with a slight difference from the Node2Vec embeddings. Deep Walk outperformed our algorithm for Weighted-L1 and Weighted-L2 and for the combination of these operators. LINE_V2 also outperformed our algorithm for the Weighted-L2 operator.

The precision score is one of the metrics on which our algorithm has relatively lower performance. However, the precision score on its own does not provide a good picture of the predictive power of algorithms. Since the main objective of link prediction is to identify connected users, it is important to increase the recall value while maintaining a reasonable precision score. In our predictions, the precision value fluctuated between 0.3 and 0.8, reaching its highest point at around 0.8 when we used both the Average and WeightedL1 operators together in weighted binary cross-entropy cost models.

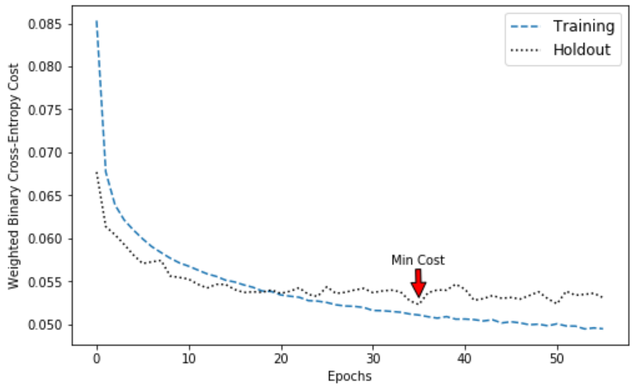


Figure 13. Example of Model Selection Process

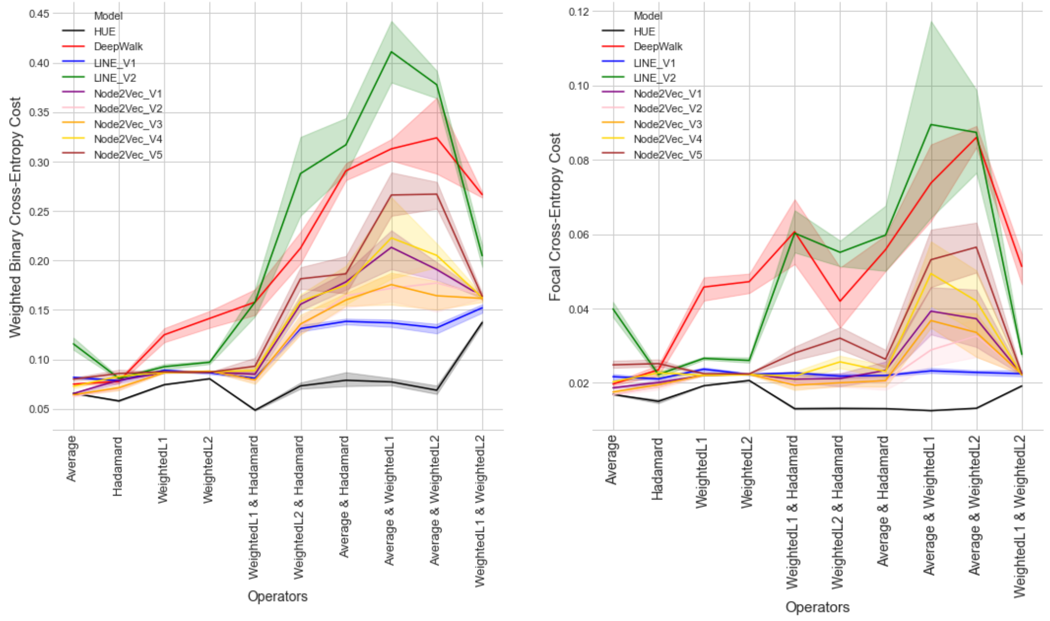


Figure 14. Model Costs

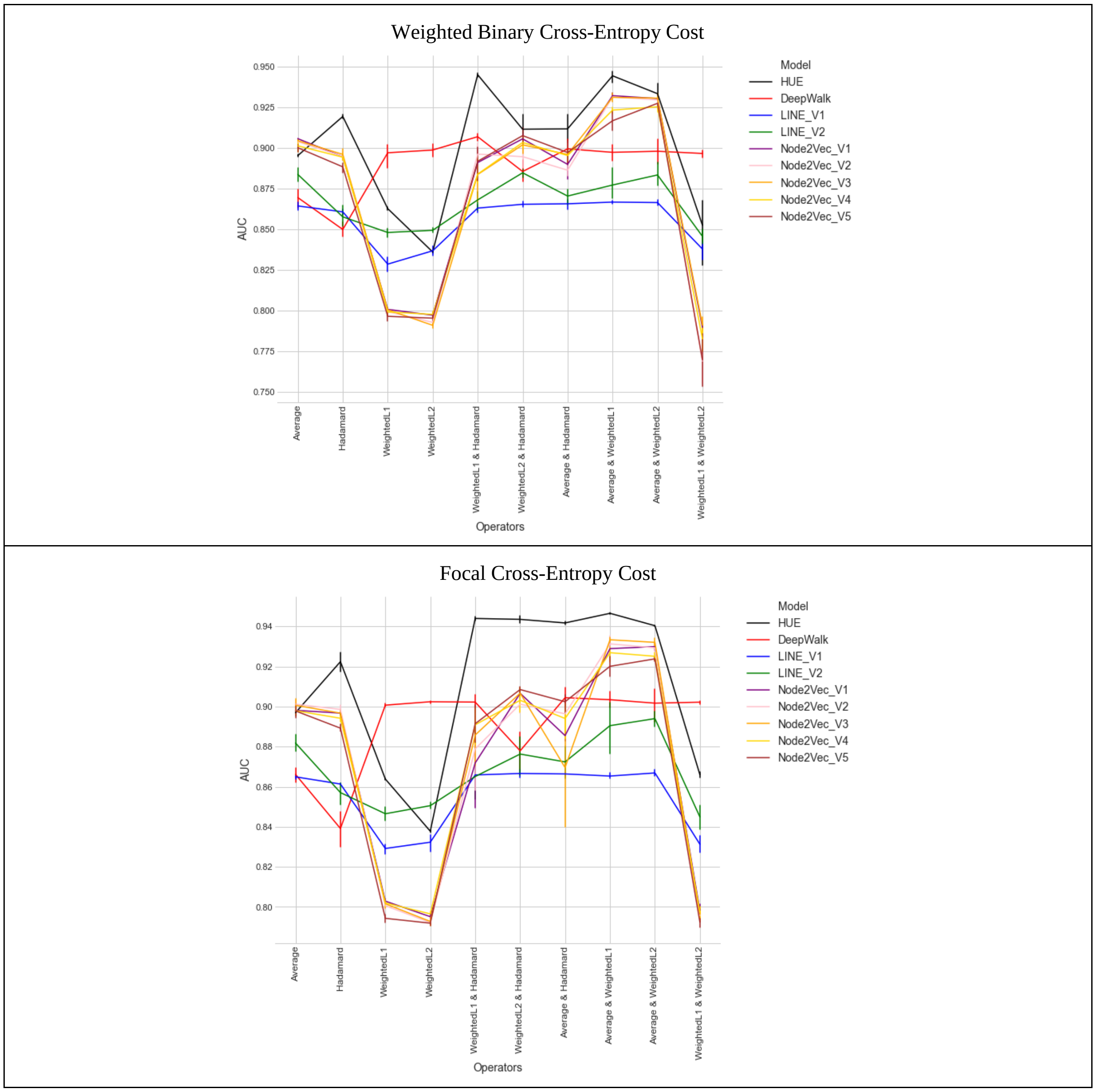


Figure 15. AUC Metric for Weighted and Focal Cross-Entropy

5 Discussion

Online social networks differ from traditional consumer networks in a variety of ways. At the outset, most of them have zero cost to join. As a result, they tend to be much larger, often spanning millions of members, even billions in some cases. Zero cost also encourages increased participation within the network and greater content creation. The sheer volume of data generated and managed within online social networks is truly staggering, reflecting large active user populations. Another key difference in online social networks is the speed at which information flows through the network. Many users constantly check their pages and are often configured to receive push notifications, encouraging them to respond and forward messages quickly. This velocity can be exploited by marketers to reach targeted users in a cascading manner rather than by issuing a blast to a motley set of customers. Reaching these users is a different proposition altogether. While some information about users can be gleaned from the user's public profile, the true essence of the user is buried in user content and activity, which is typically not available outside the platform. Marketers must rely on platform administrators to select the users that a marketer is trying to reach, with no guarantees regarding the appropriateness of the selection or the effectiveness of the message. Mechanisms that allow marketers to identify appropriate users from publicly available profiles provide marketers with an alternative approach to reaching customers. However, the profile information may be scant, leading to simplistic identification techniques. There is considerably more information that can be gleaned from the social network structure when coupled with user profile information. However, the sheer size of most social networks precludes effective analysis and a judicious portion of the relevant network needs to be extracted for meaningful analysis. In this study, we show that the proper embedding of the structure of online social networks can provide this opportunity and benefit different stakeholders.

Platform managers can use this knowledge to make the platform more efficient, effective, and responsive for individual users by integrating embedded features in their recommendation systems. In addition, the techniques could be used to identify anomalies and outliers among users, especially spurious users created to infiltrate or influence a social network. Content providers can rely on embedded variables to recognize specific communities of interest and use them to refine the experience of users in terms of delivering appropriate content and reducing the odds of providing irrelevant content and omitting pertinent content. For advertisers, this knowledge could allow them to develop and deliver targeted ads to small cohesive groups, thereby reducing ad creation efforts while diminishing the risk of ads being viewed as irrelevant. In addition, social network users can also benefit through more effective use of their most precious commodity, i.e., time, and through being subjected to a smaller barrage of information.

This paper introduces a novel homophily-based approach to embedding the structure of social graphs in a low-dimensional space while retaining the community information contained in the graph. We thus take advantage of a new second-order proximity metric in our research, which not only measures the similarity of the core vertices based on their common number of neighbors but also on basis of the pattern of connectivity between them. Prior studies have provided power algorithms for node embedding that can be applied to social graphs. However, the inability to derive meaningful dimensions from social networks significantly limits their application. Our study addresses this gap by using the semantic information present on certain nodes in online social networks. Our research is in line with selective exposure theory (Sears & Freedman, 1967; Zillmann & Bryant, 1985; Zillmann, 1988; Huang et al., 2013) in which users follow specific mass media networks that reinforce their own views and preferences. In addition, the flexibility in the selection of core vertices in our method enables the identification of preferences in other domains, such as movie/music streaming platforms.

Empirical analysis of a large network of Twitter users shows that the proposed homophily-based user embedding method can effectively embed the structure of the social network and reflect it in a reduced space. Our further exploration of social pages within communities of interests confirms the functional property of embedding metrics. We also use the embedded features identified by our proposed algorithm for link prediction. Results show that our algorithm is superior to other node-embedding methods in the literature, suggesting potential for the application of this approach in friendship recommendation systems.

6 Implications

This study makes several novel contributions to theory and practice. First, it offers a novel node-embedding approach for graph embedding, which has tangible implications for social network analysis. To our knowledge, HUE is the only node-embedding approach that enables the extraction of meaningful variables from the structure of online social networks. The extracted variables include demographic as well as user preference information and demonstrate the potential of homophily-based approaches for social networks.

Second, this study contributes to practice by offering a tool that can be easily adopted by marketing programs to provide personalized services to users of online social networks based on their preferences, which would allow platform managers, content providers, and advertisers to target individuals effectively and reduce the costs associated with customer retention. While the obvious implication is to use this information to reinforce behavior, it also affords the opportunity to bring about change through targeted prosocial interventions. Thus, for example, social marketing programs can benefit from the results of this study by identifying users who have preferences for undesirable behaviors and provide alternative motivational activities that match their other preferences.

Third, the proposed HUE method could be adopted in recommendation systems in online social networks to identify relevant social pages and potential friends to individuals based on their preferences. It would also be useful in setting up recommendations for new entrants to a social network. In addition, it could be used to control blind recommendations in online social networks where users are pushed toward extreme content.[15] Through this approach, administrators of online social network platforms

[15] https://www.cnn.com/2019/03/17/tech/youtube-facebook-twitter-radicalization-new-zealand/index.html

would be able to identify various communities of interests on their platform and evaluate the risks and benefits associated with each community through the assessment of posted content within that community.

Cybersecurity and safety experts can also benefit from this approach. There is no dearth of social pages that propagate inappropriate content in online social networks. Likewise, there are a number of pages that are designed to induce users to introduce malware and other harmful software into individual or corporate devices and platforms. Identifying communities of interest could facilitate the process of detecting these pages within online social networks. Our exploration of data shows that some communities of interest contain social pages that have recently been suspended by Twitter. Paying additional attention to other members of such communities would allow platforms such as Twitter to identify potentially troubling social pages more quickly and accurately.

Fifth, the extracted user preferences could also be used to study political partisanship in online social networks. One of the takeaways from this study is that supporters of major political parties in the US receive news from partisan communities of interest that tends to reinforce their own political ideology. This finding affords policy makers the opportunity to monitor and track political communities of interest. However, this could be viewed as a double-edged sword, where some entities seek to increase the level of bipartisanship in the network, while others seek to drive a wedge between the groups for partisan reasons.

The ability to go beyond surface-level information embedded in user profiles and incorporate network structures as well as linkages to additional networks makes the HUE method particularly effective for marketers. The application of the HUE method is not limited to online social networks. Other research studies could benefit from HUE by applying it to their context. For example, movie recommendation systems could take advantage of this research by creating an extended bipartite graph of users and movies, where movies form the core vertices. Through this approach, companies such as Netflix could quantify individual preferences for different types of movies. In summary, HUE is a novel approach that enables the extraction of information based on interactions between two different entities.

7 Limitations and Future Research

As with all methods, HUE has its limitations. While the method offers a comprehensive approach for embedding user preferences from the structure of social graphs, it is highly dependent on the selection of core vertices. Different sampling strategies for the selection of core vertices may result in alternative embedding dimensions. In addition, the cohesiveness of the sampled vertices may yield overconfidence or lack of focus in the findings. For example, a particularly cohesive sample would indicate stronger relations than are present in the general population. Conversely, a very diffuse sample would mask some clear relationships. Future studies could expand our work by addressing the implications of different strategies for the selection of core vertices. In addition, the proposed method was developed for undirected graphs. This ignores the weak relationship of individuals in online social networks, which is quite common in some networks, especially Twitter. Another area where a homophily-based user embedding approach might be less effective is in networks with a formal group structure, like organizational networks. In this case, the number of interactions is probably less significant than whom the interactions are with. Finally, multiple social network users sharing an account would likely trip up the homophily-based approach, as would also be the case with most other interaction-based approaches.

Future extensions of our work could apply the use of HUE to content recommendation systems, allowing recommendation systems to suggest appropriate content and topics to new users, thereby improving the quality of their interactions with the platform.

References

Ahmed, A., Shervashidze, N., Narayanamurthy, S., Josifovski, V., & Smola, A. J. (2013). Distributed large-scale natural graph factorization. *Proceedings of the 22nd International Conference on World Wide Web* (pp. 37-48).

Aguirre, E., Mahr, D., Grewal, D., de Ruyter, K., & Wetzels, M. (2015). Unraveling the personalization paradox: The effect of information collection and trust-building strategies on online advertisement effectiveness. *Journal of Retailing, 91*(1) 34-49.

Aiello, L. M., Barrat, A., Schifanella, R., Cattuto, C., Markines, B., & Menczer, F. (2012). Friendship prediction and homophily in social media. *ACM Transactions on the Web 6*(2) 9.

Aldecoa, R., & Marín, I. (2011), Deciphering network community structure by surprise. *PloS ONE, 6*(9), Article e24195.

Aldecoa, R., & Marín, I. (2013). Surprise maximization reveals the community structure of complex networks. *Scientific Reports, 3*, 1060-1073.

Almansoori W, Gao, S., Jarada, T. N., Elsheikh, A. M., Murshed, A. N., Jida, J., Alhajj, R., & Rokne, J. (2012). Link prediction and classification in social networks and its application in healthcare and systems biology. *Network Modeling Analysis in Health Informatics and Bioinformatics, 1*(1-2), 27-36.

Arnoux, P. H., Xu, A., Boyette, N., Mahmud, J., Akkiraju, R., & Sinha, V. (2017). 25 Tweets to Know You: A New Model to Predict Personality with Social Media. *Proceedings of the 11th International AAAI Conference on Web and Social Media*.

Belkin, M., & Niyogi, P. (2001, Dec 3 – Dec 6). Laplacian eigenmaps and spectral techniques for embedding and clustering. *Proceedings of the 14th Advances in Neural Information Processing Systems Conference* (pp. 585-591).

Bleier A, & Eisenbeiss, M. (2015). The importance of trust for personalized online advertising. *Journal of Retailing, 91*(3), 390-409.

Blondel, V. D., Guillaume, J. L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment, 10*, Article P10008.

Cai, H., Zheng, V. W., & Chang, K. C. C. (2018). A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Transactions on Knowledge and Data Engineering, 30*(9), 1616-1637.

Cao, S., Lu, W., & Xu, Q. (2015). Grarep: Learning graph representations with global structural information. *Proceedings of the 24th ACM International Conference on Information and Knowledge Management* (pp. 891-900).

Chen, J., & Stallaert, J. (2014). An economic analysis of online advertising using behavioral targeting. *MIS Quarterly, 38*(2), 429-449.

Cheng, R., Pang, J., & Zhang, Y. (2015). Inferring friendship from check-in data of location-based social networks. *Proceedings of IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 1284-1291

Dhouioui Z, Tlich, H., Toujeni, R., & Akaichi, J. (2016). A fuzzy model for friendship prediction in healthcare social networks. *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining* (pp. 1050-1054).

Faraj, S., Kudaravalli, S., & Wasko, M. (2015). Leading collaboration in online communities. *MIS Quarterly, 39*(2), 393-412.

Fortunato, S., & Barthelemy, M. (2007). Resolution Limit in Community Detection. *Proceedings of the National Academy of Sciences, 104*(1), 36-41.

Gal-Or, E., Gal-Or, M., May, J. H., & Spangler, W. E. (2006). Targeted advertising strategies on television. *Management Science, 52*(5), 713-725.

Glorot, X, & Bengio Y. (2010). Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (pp. 249-256).

Goldfarb, A., & Tucker, C. E. (2011). Privacy regulation and online advertising. *Management Science, 57*(1), 57-71.

Goyal, P., & Ferrara, E. (2018). Graph embedding techniques, applications, and performance: A survey, *Knowledge-Based Systems, 151*(1), 8-94.

Grover, A., & Leskovec, J. (2016). Node2vec: Scalable feature learning for networks. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 855-864).

Gu, B., Konana, P., Raghunathan, R., & Chen, H. M. (2014). Research note—The allure of homophily in social media: Evidence from investor responses on virtual communities. *Information Systems Research, 25*(3), 604-617.

Hanley, J. A., & McNeil, B. J. (1982). The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology, 143*(1), 29-36.

Huang Y, Shen, C., & Contractor, N. S. (2013). Distance matters: Exploring proximity and homophily in virtual world networks. *Decision Support Systems*, *55*(4), 969-977.

Jacomy, M., Venturini, T., Heymann, S., & Bastian, M. (2014). ForceAtlas2: A continuous graph layout algorithm for handy network visualization designed for the Gephi software. *PloS ONE, 9*(6), Article e98679.

Johnson, M. (1985). Relating metrics, lines and variables defined on graphs to problems in medicinal chemistry. In Alavi, Y., Chartrand, G., Lesniak, L., Lick, D. and Wall, C. (Eds), *Graph theory with applications to algorithms and computer science* (pp. 457-470) Wiley.

Kullback, S., & Leibler, R. A. (1951). On information and sufficiency. *The Annals of Mathematical Statistics*, *22*(1), 79-86.

Kossinets, G., & Watts, D. J. (2009). Origins of homophily in an evolving social network. *American Journal of Sociology, 115*(2), 405-450.

Lee, D. D., & Seung, H. S. (2001) Algorithms for non-negative matrix factorization. In T. Leen, T. Dietterich, & V. Tresp, V. (Eds), *Advances in neural information processing systems* (pp. 556-562) MIT Press.

Li Y., Wu, C., Luo, P., & Zhang, W. (2013). Exploring the characteristics of innovation adoption in social networks: Structure, homophily, and strategy. *Entropy*, *15*(7), 2662-2678.

Lin, T. Y, Goyal, P., Girshick, R., He, K., & Dollár P. (2017). Focal loss for dense object detection. *Proceedings of the IEEE International Conference on Computer Vision* (pp. 2980-2988).

Ling, C. X., & Sheng, V. S. (2010). Cost-sensitive learning. In C. Sammut & G. I. Webb (Eds). *Encyclopedia of Machine Learning*. Springer. https://doi.org/10.1007/978-0-387-30164-8_181

McPherson, M., Smith-Lovin, L., & Cook, J. M. (2001). Birds of a feather: Homophily in social networks. *Annual Review of Sociology, 27,* 415-444.

Mirza, B., Lin Z, & Toh, K. A. (2013). Weighted online sequential extreme learning machine for class imbalance learning. Neural Processing Letters, *38*(3), 465-486.

Newman, M. E., & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E, 69*(2), Article 026113.

Niepert, M., Ahmed, M., & Kutzkov, K. (2016). Learning convolutional neural networks for graphs. *Proceedings of the International Conference on Machine Learning* (pp. 2014-2023).

Perozzi, B., Al-Rfou, R., & Skiena, S. (2014). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 701-710).

Ramachandran, P., Zoph, B., & Le, Q. V. (2017). Searching for activation functions. https://arxiv.org/abs/1710.05941

Ribeiro, L. F., Saverese, P. H., & Figueiredo, D. R. (2017). Struc2vec: Learning node representations from structural identity. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 385-394)

Rollins, B., Anitsal, I., & Anitsal, M. M. (2014). Viral marketing: Techniques and implementation. *Entrepreneurial Executive, 19,* 1-17.

Roweis, S. T., & Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science, 290*(5500), 2323-2326.

Sears, D. O., Freedman, J. L. (1967). Selective exposure to information: A critical review. *Public Opinion Quarterly, 31*(2), 194-213.

Shi, Z., Whinston, A. B. (2013). Network structure and observational learning: Evidence from a location-based social network. *Journal of Management Information Systems, 30*(2), 185-212.

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., & Mei, Q. (2015). Line: Large-scale information network embedding. *Proceedings of the 24th International Conference on World Wide Web* (pp. 1067-1077)

Tenenbaum, J. B., De Silva, V., Langford, J. C. (2000). A Global geometric framework for nonlinear dimensionality reduction. *Science, 290*(5500), 2319-2323.

Traag, V. A., Krings, G., & Van Dooren, P. (2013). Significant scales in community structure. *Scientific Reports, 3*, Article 2930.

Traag, V. A., Aldecoa, R., & Delvenne, J. C. (2015). Detecting communities using asymptotical surprise. *Physical Review E*, *92*(2), Article 022816.

Trusov, M., Ma, L., & Jamal Z. (2016). Crumbs of the cookie: User profiling in customer-base analysis and behavioral targeting. *Marketing Science*, *35*(3), 405-426.

Wang, P., Xu, B., Wu, Y., & Zhou, X. (2015). Link prediction in social networks: the state-of-the-art. *Science China Information Sciences, 58*(1), 1-38.

Wang, D., Cui, P., & Zhu, W. (2016). Structural deep network embedding. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 1225-1234).

Xie, X. (2010). Potential friend recommendation in online social network. *Proceedings of IEEE/ACM International Conference on Green Computing and Communications and International Conference on Cyber, Physical and Social Computing* (pp. 831-835).

Yan, J., Liu, N., Wang, G., Zhang, W., Jiang, Y., & Chen, Z. (2009). How much can behavioral targeting help online advertising? *Proceedings of the 18th International Conference on World Wide Web* (pp. 261-270)

Zillmann, D., & Bryant, J. (1985). Affect, mood, and emotion as determinants of selective exposure. In D. Zillmann & J. Bryant (Eds.), *Selective exposure to communication* (pp. 157-190). Lawrence Erlbaum.

Zillmann, D. (1988). Mood management through communication choices. *American Behavioral Scientist*, *31*(3), 327-341.

Appendix A: HUE pseudocode

The following section shows the pseudocode implementation of HUE. The actual implementation of the HUE algorithm utilizes matrix operations instead of nested looping structures.

Algorithm: Homophily-based User Embedding

```
initialization
CoreVertices = selectCoreVertices (Graph, SizeOfCores=N, method=['random','predefined'])
BipartiteGraph = formBipartiteGraph(Graph, CoreVertices)

SimilarityMatrix = initializeSimilarityMatrix(N,N)
for i = 1 to N do
    MappedGraph[i] = findEgosAlterNetwork(BipartiteGraph, CoreVertices[i])
    for j = i+1 to N do
      MappedGraph[j] = findEgosAlterNetwork (BipartiteGraph, CoreVertices[i]))
      SimilarityMatrix[i,j] = findGraphSimilarity(MappedGraph[i], MappedGraph[j])
      SimilarityMatrix[j,i] = findGraphSimilarity(MappedGraph[i], MappedGraph[j])
    end for
end for
SimplifiedCoreGraph = createWeightedGraph (SimilarityMatrix)
ClusteringQuality = 0
BestClusters = {}
for iteration = 1 to maxIterations do
    Clusters, Quality = findClusters(SimplifiedCoreGraph, LouvainAlgorithm, SurpriseObjectiveFunction)
    if Quality > ClusteringQuality then
        ClusteringQuality = Quality
        BestClusters = Clusters
    end if
end for

for i=1 to N do:
   CoreWeights[i] = computeWeightsOfCores(BestClusters, SimilarityMatrix)
end for
EmbeddingMatrix = initializeEmbeddingMatrix(size(Vertices),size(BestClusters))
for i = 1 to size(Vertices) do
    for j = 1 to size(BestClusters) do
        EmbeddingMatrix [i,j] = computeEmbeddingValues(Vertices[i], BestClusters[j], BipartiteGraph, CoreWeights)
    end for
end for
```

Figure A1. HUE Algorithm

Appendix B: Selection of Representative Terms from Tweets

To gain insight into the communities of interest, we analyzed the tweets posted on social pages that are followed by Twitter users. One of the main challenges in extracting representative words from the community of interest is the high frequency of words used by all social pages for communication on Twitter. These are words that do not characterize the community of interest but are prominent because of their frequency. In order to remove these words from our analysis, we adopted the following six-step procedure.

Step 1. We selected all social pages belonging to a particular community and then aggregated the tweets posted at the social page level over a six-month period. In our approach, tweets from a social page form a document, and all the documents in one community form a corpus.

Step 2. We randomly selected an identical number of social pages in Step 1, but this time from outside the community. This new collection serves as a control group which allows non-representative terms to be removed. As with Step 1, we aggregated the tweets of the randomly selected social pages into the second corpus.

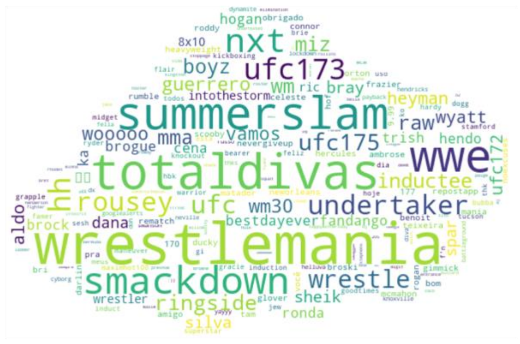
Step 3. Our pre-processing activities comprised tokenization, lemmatization, and the elimination of stop words. We then picked the top 5000 high-frequency terms from the first corpus to develop the vocabulary set. This vocabulary set includes both representative terms of the community of interest and those common terms shared with other communities of interest.

Step 4. We used the vocabulary set identified in Step 3 to form normalized document-term-frequency matrices, where the values are normalized at the document level. We formed the matrix for each corpus separately.

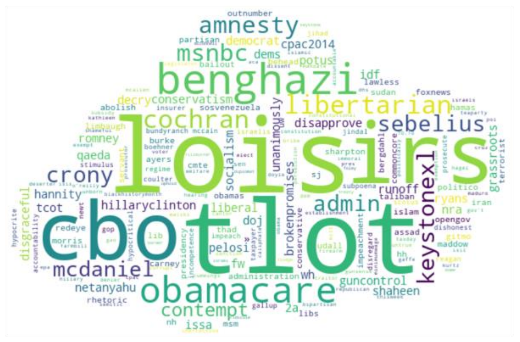
Step 5. For each term, we computed the sum of the normalized values across the documents in each corpus. This procedure gave us two scores for each term, one for the usage of terms within the community and another for usage outside the community.

Step 6. Finally, for each term, we divided the inside community score by the outside community score to measure the weight of each term in the vocabulary. As a result, community-specific terms have high weights, and common terms across communities are weighted low. In addition, noncommunity terms are weighted extremely low.

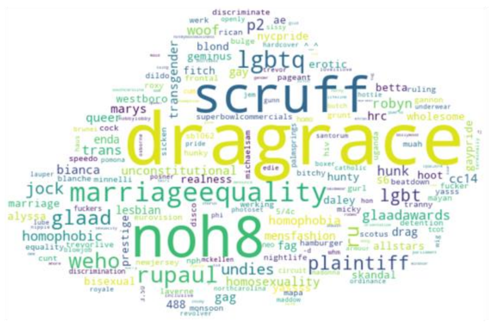
Figure B1 illustrates some additional word clouds of the communities of interest that are not presented in the main manuscript.



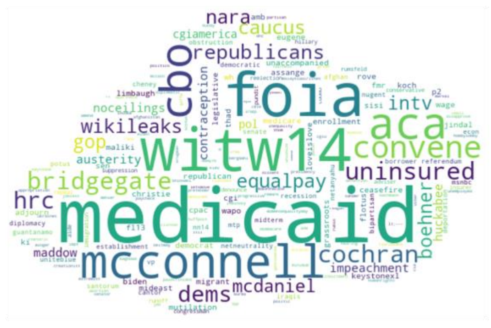
(a) World Wrestling Entertainment



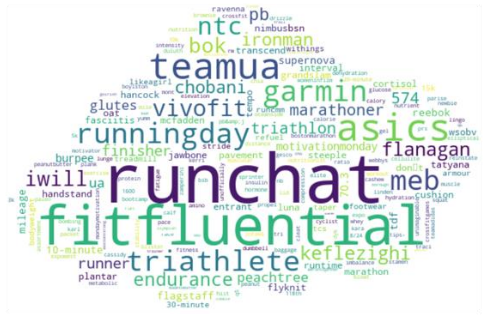
(b) Republicans



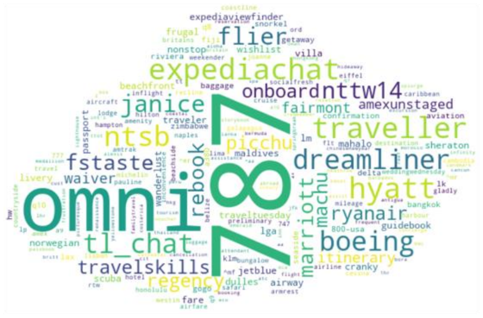
(c) LGBT



(d) Democrats



(e) Running



(f) Travel

Figure B1. Word-Cloud of Representative Words for Sample Communities

Appendix C: List of Top Identified Communities of Interest

Table C1 shows the list of top communities of interest.

Table C1. Communities of Interests by Categories

Art and Design	**Geographical**	**TV Shows and Movie Series**	**Movies**
- Art & Design - Photography - Web Design	- Atlanta - Austin - Baltimore - Boston - Cleveland - Chicago - Dallas - Detroit - Denver - Houston - Las Vegas - Los Angeles - Miami - Milwaukee - Minnesota - New Orleans - New York - Philadelphia - Pittsburgh - San Francisco - Seattle - Washington DC	- Bachelor & Bachelorette - Chelsea Lately Show - Elvis Duran & the Morning Shows - Girl Code - Harry Potter - Howard Stern Show - Real Housewives Show - Teen and Pregnancy - Walking Dead Series - Workaholics	- Comedy - Criminal - Drama - Paranormal - Sitcoms - Science Fiction - Film Academy
Artists			
- African American Artists - Celebrity Personalities - Latino Artists - MTV Stars			
Business and Entrepreneurship		**Sports**	**Media**
- Digital Advertising - Job and Technology - Leadership & Entrepreneurship - Marketing - Public Relations - Real Estate - Business and Technology		- Auto Racing - Basketball & Football - Golf - Hockey - Los Angeles Sport - New York Sport - Olympic Sports - Soccer - Washington DC Sport - Wrestling (WWE)	- ABC, CW, and FXM channels - CBS Channel - Marvel Channel - NBC Channel - NPR - Reality TV (History & Discovery) - SYFY Channel - USA Network
Entertainment	**Health and Activity**	**News**	**Other**
- Broadway - Disney - Console Games - Travel - Entertainment Influencers	- Dance - Men's Health - Health (Preventive Care) - Health & Fitness	- ABC News - Business News - Journalism - Technology News - Weather News	- Humor - Life Facts - Porn
Environmental and Social Concerns	**Music**	**Technology Brands**	
- African American Social Activist - Charity - Education - LGBTQ - Wildlife	- Boy Bands - Christian Music - Country Music - Electronic Dance Music - Hard Rock Music - Hip Hop Music - Indie Rock Music - Music Record - Pop Rock Music - Pop Music - Punk Music - Soul Music	- Amazon - Twitter - Microsoft - Apple - Google	
Politics and Government		**Shopping**	
- Conservative Party - Defense & Homeland Security - Liberal Party		- Automobile - Fashion - Sport Wear - Retail	
Food and Beverages	**Religion and Spirituality**	**Reading and Learning**	
- Beer - Chefs - Cocktail and Whisky - Food and Wine	- Biblical Quotes - Motivational Quotes - Wisdom Quotes	- Book Publishers - Science - Screen Writers	

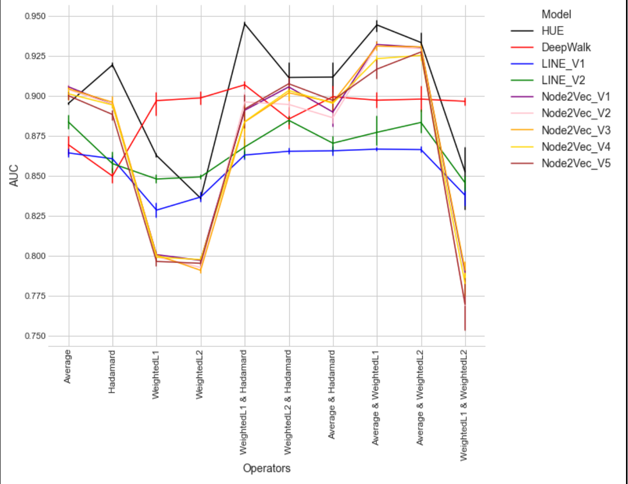
Table C2 depicts a sample set of social pages for selected communities of interest.

Table C2. Sample of Social Pages in Communities of Interest

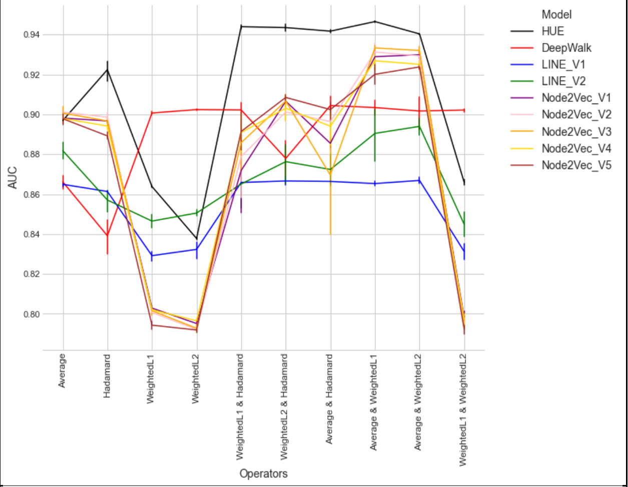
Country Music Community of Interest	**Charity Community of Interest**	**Health Community of Interest**	**Conservative Community of Interest**
- Miranda Lambert - Luke Bryan - Brad Paisley - Keith Urban - Tim McGraw - Reba McEntire - Lady Antebellum - Jason Aldean - Martina McBride - Eric Church	- RED - Gates Foundation - Charity water - UNICEF - DoSomething - ONE Campaign - Amnesty USA - DonorsChoose - Kiva - Peace Corps	- American Red Cross - CDC Emergency - Health - WebMD - NYT Health - The Scope - American Cancer Society - Mayo Clinic - WHO - NPR Health News	- Fox News - Mitt Romney - Sarah Palin - John McCain - John Boehner - Governor Christie - Paul Ryan - Michelle Malkin - Newt Gingrich - Sean Hannity
Automobile Community of Interest	**Video Games Community of Interest**	**Travel Community of Interest**	**Google Community of Interest**
- Tesla - Ford Motor Company - Audi - Chevrolet - Jeep - Volkswagen - Toyota - Lexus - General Motors - Mercedes-Benz	- PlayStation - Xbox - IGN - Larry Hryb - Electronic Arts - Rockstar Games - Nintendo of America - GameStop - Kevin Pereira - SEGA	- Southwest Airlines - JetBlue Airways - American Airlines - Virgin America - Delta - United Airlines - Travel Channel - Travel + Leisure - Orbitz - Airfarewatchdog	- Gmail - Tumblr - Google Chrome - Android - Google Maps - Nexus - Google Play - Google Mobile - Google Analytics - SwiftKey

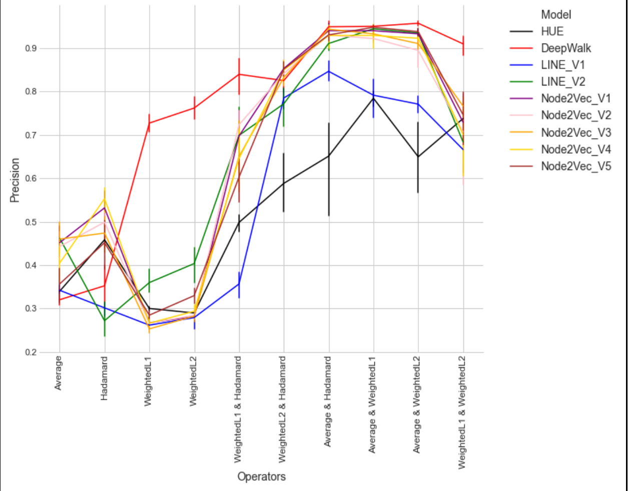
Appendix D: Performance Metrics for Link Prediction Task

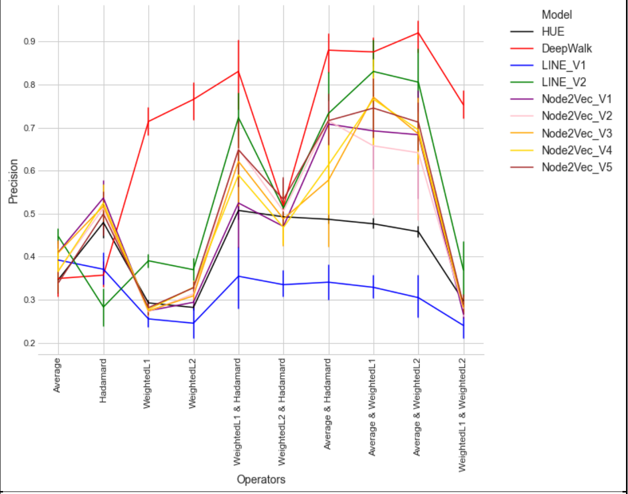
Weighted Binary Cross-Entropy Cost

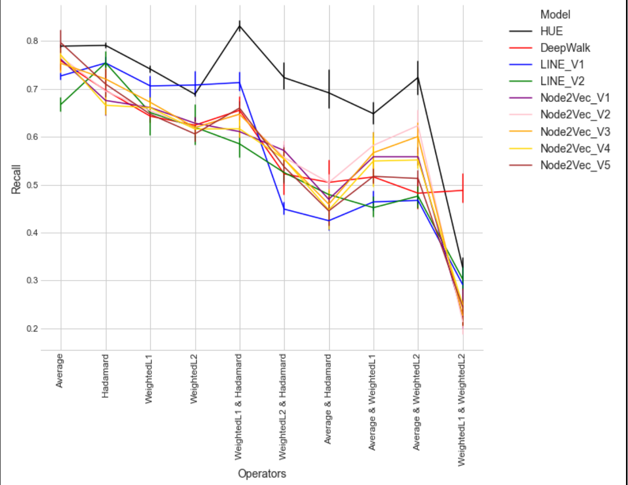


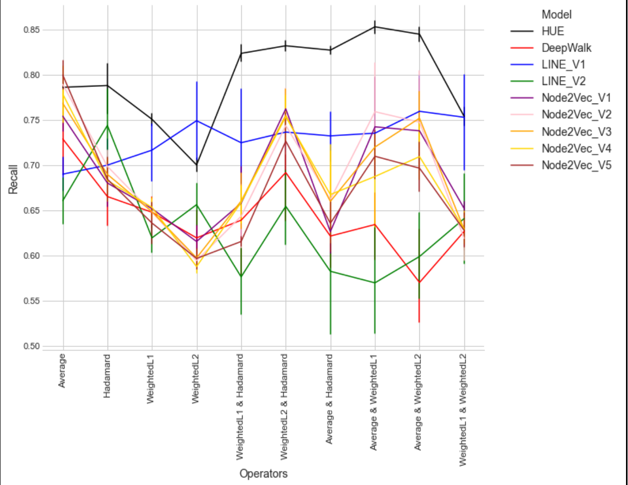
Focal Cross-Entropy Cost











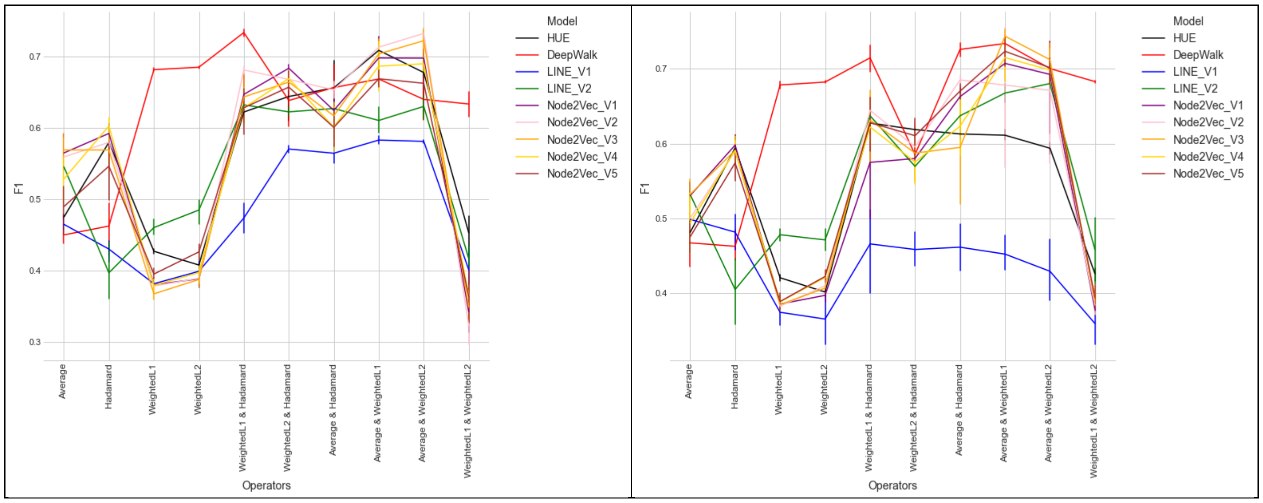


Figure D1. Performance Metrics for Weighted and Focal Cross-Entropy

Table D1. Performance Metrics for Weighted Binary Cross-Entropy Cost

Operator	Algorithm	Average precision	Average recall	Average F1	Average AUC
Average	HUE	0.340	0.789	0.475	0.895
	Deep Walk	0.320	0.759	0.450	0.869
	LINE_V1	0.342	0.727	0.465	0.864
	LINE_V2	0.462	0.668	0.546	0.884
	Node2Vec_V1	0.452	0.761	0.565	0.906
	Node2Vec_V2	0.443	0.767	0.560	0.904
	Node2Vec_V3	0.460	0.753	0.569	0.904
	Node2Vec_V4	0.405	0.770	0.529	0.901
	Node2Vec_V5	0.357	0.796	0.490	0.900
Hadamard	HUE	0.458	0.791	0.580	0.919
	Deep Walk	0.352	0.697	0.463	0.850
	LINE_V1	0.302	0.754	0.430	0.861
	LINE_V2	0.272	0.754	0.397	0.858
	Node2Vec_V1	0.532	0.676	0.592	0.895
	Node2Vec_V2	0.499	0.695	0.580	0.895
	Node2Vec_V3	0.474	0.721	0.569	0.896
	Node2Vec_V4	0.553	0.666	0.603	0.894
	Node2Vec_V5	0.452	0.710	0.547	0.888
Weighted-L1	HUE	0.300	0.741	0.427	0.863
	Deep Walk	0.727	0.643	0.682	0.897
	LINE_V1	0.262	0.706	0.382	0.829
	LINE_V2	0.360	0.650	0.460	0.848
	Node2Vec_V1	0.267	0.661	0.380	0.801
	Node2Vec_V2	0.266	0.654	0.378	0.799
	Node2Vec_V3	0.253	0.672	0.368	0.801
	Node2Vec_V4	0.266	0.661	0.379	0.799
	Node2Vec_V5	0.285	0.647	0.395	0.797
Weighted-L2	HUE	0.290	0.689	0.408	0.836
	Deep Walk	0.762	0.624	0.685	0.899
	LINE_V1	0.279	0.708	0.399	0.837
	LINE_V2	0.404	0.620	0.485	0.849
	Node2Vec_V1	0.282	0.628	0.389	0.797
	Node2Vec_V2	0.284	0.619	0.390	0.793
	Node2Vec_V3	0.283	0.619	0.388	0.791
	Node2Vec_V4	0.294	0.616	0.397	0.798
	Node2Vec_V5	0.330	0.606	0.426	0.795

Weighted-L1 & Hadamard	HUE	0.498	0.831	0.623	0.945
	Deep Walk	0.839	0.655	0.734	0.907
	LINE_V1	0.357	0.713	0.474	0.863
	LINE_V2	0.699	0.585	0.633	0.868
	Node2Vec_V1	0.697	0.611	0.647	0.891
	Node2Vec_V2	0.723	0.647	0.682	0.896
	Node2Vec_V3	0.650	0.647	0.644	0.884
	Node2Vec_V4	0.646	0.617	0.628	0.884
	Node2Vec_V5	0.602	0.660	0.628	0.892
Weighted-L2 & Hadamard	HUE	0.588	0.723	0.644	0.911
	Deep Walk	0.825	0.523	0.639	0.885
	LINE_V1	0.785	0.449	0.571	0.865
	LINE_V2	0.771	0.525	0.623	0.885
	Node2Vec_V1	0.853	0.571	0.684	0.906
	Node2Vec_V2	0.841	0.557	0.668	0.895
	Node2Vec_V3	0.832	0.554	0.664	0.902
	Node2Vec_V4	0.849	0.554	0.669	0.903
	Node2Vec_V5	0.851	0.537	0.657	0.908
Average & Hadamard	HUE	0.651	0.692	0.656	0.912
	Deep Walk	0.949	0.505	0.657	0.899
	LINE_V1	0.846	0.425	0.565	0.866
	LINE_V2	0.910	0.479	0.627	0.870
	Node2Vec_V1	0.940	0.469	0.625	0.890
	Node2Vec_V2	0.930	0.504	0.654	0.886
	Node2Vec_V3	0.943	0.459	0.617	0.896
	Node2Vec_V4	0.929	0.447	0.601	0.895
	Node2Vec_V5	0.930	0.445	0.601	0.897
Average & Weighted-L1	HUE	0.784	0.648	0.709	0.944
	Deep Walk	0.950	0.516	0.669	0.897
	LINE_V1	0.791	0.464	0.583	0.867
	LINE_V2	0.944	0.452	0.611	0.877
	Node2Vec_V1	0.940	0.558	0.698	0.932
	Node2Vec_V2	0.922	0.582	0.713	0.931
	Node2Vec_V3	0.932	0.567	0.704	0.931
	Node2Vec_V4	0.929	0.549	0.687	0.923
	Node2Vec_V5	0.948	0.517	0.669	0.917
Average & Weighted-L2	HUE	0.649	0.724	0.678	0.933
	Deep Walk	0.957	0.482	0.640	0.898
	LINE_V1	0.771	0.467	0.581	0.866
	LINE_V2	0.935	0.476	0.630	0.883
	Node2Vec_V1	0.933	0.558	0.698	0.930
	Node2Vec_V2	0.894	0.623	0.733	0.929
	Node2Vec_V3	0.910	0.601	0.723	0.931
	Node2Vec_V4	0.922	0.552	0.690	0.925
	Node2Vec_V5	0.938	0.513	0.663	0.927
Weighted-L1 & Weighted-L2	HUE	0.737	0.327	0.453	0.853
	Deep Walk	0.909	0.488	0.634	0.897
	LINE_V1	0.666	0.291	0.401	0.838
	LINE_V2	0.684	0.303	0.417	0.846
	Node2Vec_V1	0.729	0.226	0.342	0.790
	Node2Vec_V2	0.706	0.214	0.323	0.769
	Node2Vec_V3	0.766	0.227	0.349	0.791
	Node2Vec_V4	0.701	0.249	0.366	0.783
	Node2Vec_V5	0.746	0.244	0.364	0.770

Table D2. Performance Metrics for Focal Cross-Entropy Cost

Operator	Algorithm	Average precision	Average recall	Average F1	Average AUC
Average	HUE	0.347	0.786	0.482	0.897
	Deep Walk	0.349	0.729	0.467	0.866
	LINE_V1	0.392	0.690	0.499	0.865
	LINE_V2	0.447	0.661	0.532	0.882
	Node2Vec_V1	0.411	0.754	0.531	0.898
	Node2Vec_V2	0.368	0.790	0.500	0.901
	Node2Vec_V3	0.410	0.769	0.532	0.901
	Node2Vec_V4	0.367	0.779	0.494	0.898
	Node2Vec_V5	0.339	0.800	0.476	0.898
Hadamard	HUE	0.479	0.788	0.594	0.922
	Deep Walk	0.357	0.665	0.463	0.839
	LINE_V1	0.371	0.700	0.482	0.861
	LINE_V2	0.283	0.744	0.405	0.857
	Node2Vec_V1	0.537	0.680	0.598	0.897
	Node2Vec_V2	0.513	0.699	0.591	0.899
	Node2Vec_V3	0.518	0.690	0.591	0.897
	Node2Vec_V4	0.525	0.683	0.591	0.894
	Node2Vec_V5	0.499	0.685	0.574	0.889
Weighted-L1	HUE	0.292	0.751	0.421	0.864
	Deep Walk	0.714	0.648	0.678	0.901
	LINE_V1	0.255	0.717	0.375	0.829
	LINE_V2	0.390	0.620	0.478	0.847
	Node2Vec_V1	0.274	0.652	0.386	0.803
	Node2Vec_V2	0.272	0.648	0.383	0.801
	Node2Vec_V3	0.274	0.649	0.385	0.802
	Node2Vec_V4	0.277	0.653	0.389	0.802
	Node2Vec_V5	0.281	0.636	0.389	0.794
Weighted-L2	HUE	0.282	0.700	0.402	0.838
	Deep Walk	0.766	0.620	0.682	0.902
	LINE_V1	0.245	0.749	0.366	0.832
	LINE_V2	0.370	0.656	0.471	0.851
	Node2Vec_V1	0.294	0.616	0.397	0.795
	Node2Vec_V2	0.313	0.594	0.410	0.792
	Node2Vec_V3	0.309	0.597	0.407	0.793
	Node2Vec_V4	0.329	0.588	0.421	0.797
	Node2Vec_V5	0.328	0.597	0.423	0.792
Weighted-L1 & Hadamard	HUE	0.507	0.824	0.628	0.944
	Deep Walk	0.831	0.639	0.715	0.902
	LINE_V1	0.354	0.725	0.466	0.866
	LINE_V2	0.723	0.576	0.637	0.865
	Node2Vec_V1	0.525	0.658	0.575	0.872
	Node2Vec_V2	0.651	0.644	0.645	0.879
	Node2Vec_V3	0.622	0.660	0.632	0.886
	Node2Vec_V4	0.590	0.659	0.622	0.891
	Node2Vec_V5	0.648	0.616	0.628	0.891
Weighted-L2 & Hadamard	HUE	0.493	0.832	0.619	0.944
	Deep Walk	0.517	0.692	0.585	0.878
	LINE_V1	0.335	0.737	0.459	0.867
	LINE_V2	0.513	0.655	0.570	0.876
	Node2Vec_V1	0.470	0.763	0.580	0.907
	Node2Vec_V2	0.506	0.744	0.596	0.901
	Node2Vec_V3	0.488	0.755	0.587	0.907
	Node2Vec_V4	0.468	0.754	0.575	0.903
	Node2Vec_V5	0.533	0.727	0.611	0.909

Average & Hadamard	HUE	0.487	0.828	0.613	0.942
	Deep Walk	0.880	0.622	0.726	0.904
	LINE_V1	0.341	0.733	0.462	0.866
	LINE_V2	0.733	0.583	0.637	0.872
	Node2Vec_V1	0.708	0.626	0.663	0.886
	Node2Vec_V2	0.716	0.662	0.685	0.896
	Node2Vec_V3	0.578	0.660	0.595	0.870
	Node2Vec_V4	0.613	0.668	0.623	0.894
	Node2Vec_V5	0.716	0.636	0.670	0.902
Average & Weighted-L1	HUE	0.476	0.853	0.611	0.947
	Deep Walk	0.876	0.634	0.734	0.903
	LINE_V1	0.329	0.736	0.452	0.865
	LINE_V2	0.831	0.570	0.668	0.890
	Node2Vec_V1	0.692	0.743	0.707	0.929
	Node2Vec_V2	0.658	0.759	0.678	0.931
	Node2Vec_V3	0.771	0.720	0.744	0.933
	Node2Vec_V4	0.766	0.688	0.715	0.927
	Node2Vec_V5	0.745	0.710	0.723	0.920
Average & Weighted-L2	HUE	0.458	0.845	0.594	0.940
	Deep Walk	0.920	0.570	0.700	0.902
	LINE_V1	0.305	0.760	0.430	0.867
	LINE_V2	0.806	0.599	0.680	0.894
	Node2Vec_V1	0.683	0.738	0.692	0.930
	Node2Vec_V2	0.642	0.748	0.671	0.929
	Node2Vec_V3	0.685	0.752	0.712	0.932
	Node2Vec_V4	0.695	0.710	0.698	0.925
	Node2Vec_V5	0.713	0.697	0.701	0.924
Weighted-L1 & Weighted-L2	HUE	0.297	0.754	0.426	0.866
	Deep Walk	0.753	0.627	0.683	0.902
	LINE_V1	0.240	0.753	0.360	0.831
	LINE_V2	0.369	0.641	0.458	0.845
	Node2Vec_V1	0.266	0.652	0.378	0.798
	Node2Vec_V2	0.280	0.619	0.384	0.796
	Node2Vec_V3	0.282	0.629	0.390	0.796
	Node2Vec_V4	0.288	0.629	0.394	0.797
	Node2Vec_V5	0.289	0.628	0.394	0.792

About the Authors

Mahyar Sharif Vaghefi is an assistant professor in the Department of Information Systems and Operations Management at the College of Business Administration at the University of Texas at Arlington. He received his PhD in information technology management from University of Wisconsin-Milwaukee. His research interests include social media analytics, e-commerce, marketing, crowdsourcing, and health analytics. His work has appeared in journals such as *Marketing Letters*, *Electronic Commerce Research and Applications*, *International Journal of Information Management*, and *IEEE Transactions on Computational Social Systems*.

Derek L. Nazareth is an associate professor of information technology management at the Lubar School of Business at the University of Wisconsin-Milwaukee. He received his PhD in management information/decision systems from Case Western Reserve University. His current research interests include information security and privacy, medical informatics, evolutionary algorithms, and network science. He has published over 40 articles in leading journals including *IEEE Transactions on Knowledge and Data Engineering, ACM Transactions on Management Information Systems, Journal of Management Information Systems, Journal of the Association for Information Systems, IEEE Transactions on Systems Man & Cybernetics, Decision Support Systems, Communications of the ACM, Information & Management,* among others. In addition, he has 50 papers in refereed conference proceedings and workshops. He has served as associate editor for *IEEE Transactions on Services Computing.* He has also served as the program chair for AMCIS, and as treasurer for ICIS.