Mining Online Social Networks: Deriving User Preferences through Node Embedding

(Forthcoming at Journal of Association for Information Systems)[*]

Mahyar Sharif Vaghefi
Department of Information Systems and Operations Management
College of Business Administration
University of Texas at Arlington
mahyar.sharifvaghefi@uta.edu

Derek L. Nazareth
Lubar School of Business
University of Wisconsin-Milwaukee
derek@uwm.edu

November 2020

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 to online social networks. Our proposed method not only quantifies social graphs in a structured format, but also enables the 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. 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 recommender systems as well as detection of outliers and anomalies.

Keywords. Communities of Interest, Individual Preference, Recommender Systems, Social Network Analysis, User Embedding, Visualization.

Suggested Citation:
Sharif Vaghefi, M and Nazareth, DL, (2021, forthcoming), Online Social Networks: Deriving User Preferences through Node Embedding, *Journal of the Association for Information Systems*, https://aisel.aisnet.org/jais/.

Introduction

Online social network as an IT-based artifact has led to drastic changes in our way of living and has played a significant role in disseminating information during global events such as the 2016 US presidential election and the 2020 coronavirus pandemic. These platforms have also greatly revolutionized online marketing. Targeted advertising, also known as “behavioral targeting,” is one area that has benefited the most from this new paradigm. Targeted advertising, which was pioneered by search engines such as Google, uses cookies and other tracking tools to derive individual interest from their 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 intention of individuals (Yan et al. 2009, Goldfarb and Tucker 2011). It also provides a more relaxed environment for competition among advertisers by enabling them to target different segments of users. This ultimately leads to lower advertising costs for businesses compared to the traditional form of advertising (Chen and Stallaert 2014). The high granularity of information collected/shared on social media platforms makes this type of media even more attractive to marketers for targeted advertising. In many cases, what users share on social media platforms can reveal much more about them than they might realize. Consider a simple 140-character tweet posted on Twitter, the information that can be captured is well beyond those few characters. The new analytical tools make it possible to extract complicated patterns from the large volume of data that can reveal individual demographic information, living location, personality (Arnoux et. al 2017) and even their brand perception (Culotta et al. 2016). Such information can be utilized by marketers to pitch their products and services directly to users on social platforms.

However, behavioral targeting is not without its challenges. Advertisers should 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 shows 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 need to limit their predicted target group to smaller samples in which they have more user information and are more confident about their behavior. However, with this approach, companies may lose a lot of potential customers as the available user level information are generally limited. On the other hand, achieving high recognition rate requires broadening the predicted target group and reach out to higher number of users. This approach may lead to targeting of the wrong audience, which wastes resources on the company as well as time for individuals. Targeting the wrong audience is generally expected to have an adverse effect on the company in the long run by challenging customer retention and customer relationship management (Rollins et al. 2014). In addition, it fosters the spread of negative word of mouth in social networks. Thus, the role of profiling methods in identification of user attributes can be crucial as it can improve the quality of predicted target groups. A good profiling method can improve both the accuracy and recognition metrics.

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 use cookies and keep track of transactions, product views and individual shopping carts. These data collection approaches raise privacy concerns, as many people do not like to be continuously tracked (Aguirre et al. 2015) even though they perceive personalized ads useful (Bleier and Eisenbeiss 2015). Analysis of such large datasets is also

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

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. In fact, 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 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 and 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 characteristics of users and making link prediction 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. users joining, leaving or altering their relationships in the social network and are ineffective in 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 that is robust and can tolerate incremental changes in the network. Using the concept of homophily from social science, and a graph-based representation of the social network, we are able to extract actionable user preferences that can be used for targeted advertising. Our proposed algorithm is consistent with recent node-embedding research works (Perozzi et al. 2014, Grover and Leskovec 2016) and is aimed at transforming 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 the application to 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 literature on node embedding approaches and highlights the limitation 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 the following section. We then demonstrate the application of our user embedding method in link prediction task for a real dataset of users. A discussion of the findings, and an assessment of managerial and research implications round out the paper.

Literature Review

Extracting underlying meaning from a graph typically rely 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 were manifest in techniques like multidimensional scaling, which typically sought 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 2002) and LLE (Roweis and 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 network are different, and need other techniques for reducing complexity. Matrix Factorization (Ahmed et al. 2013) and Non-Negative Matrix Factorization (Lee and Seung 2001) are another group of dimension reduction approach that can be applied to the adjacency matrix of the graphs. Non-negative matrix factorization has also been used in literature for the extraction of homophilic features in online social networks (Shi and 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, which are designed to retain graph features. We can classify these approaches into three areas: (1) node embedding, (2) edge embedding, (3) entire graph embedding. In this study our main focus is on the node embedding methods. See Cai et al. (2018) for comprehensive review of literature.

The main objective of the 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 and 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 and 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 the embedding vectors.

Models without random walk apply deep learning techniques to the matrix representing the graph in order to maintain proximity among nodes (Niepert et al. 2016, Wang et al. 2016). There are several options available for deep learning application. LINE (Tang et al 2015) is one of most prominent algorithms in this collection. It specifies two conditional and empirical distribution functions for the 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 non-interpretable variables, meaning that while they can be used to determine a user's structural similarity to other

[2] Node2Vec uses two hyperparameters *p* and *q* that controls 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.

users or to community membership, they do not provide us with interpretable values such as preferences to analyze user behavior. Second, they suffer from a practical issue when it comes to accommodating new users joining the network. This requires re-computing the embeddings for the entire network and making the recommendation of new links unfeasible. In this study, we plan to address both these shortcomings using the concept of homophily in social science.

The homophily concept suggests that individuals have a strong tendency to interact with people who have similar attributes instead of people with different attributes (McPherson et. al 2001). Homophily has roots in 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 homophilous relationships: (i) it increases the chance of being liked by others, (ii) it makes it easier for individuals to get confirmation from other similar individuals, (iii) the ongoing cost of maintaining relationships with similar others is lower than with dissimilar ones due to ease of developing trust and solidarity with them, and (iv) individual choices of relationship are frequently constrained by factors such as geographical locations, neighborhoods, working places, and schools. These constraints lead to homogenous choices of relationships by individuals in a social network (Kossinets and 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 of 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.

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 the second-order proximity at the community level and to represent the members of the social network using their connectivity pattern to smaller samples of selected members. To accomplish this, we introduce a new proximity measure called ego's alter-network structural similarity, that 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.

Ego's alter-network structural similarity

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

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

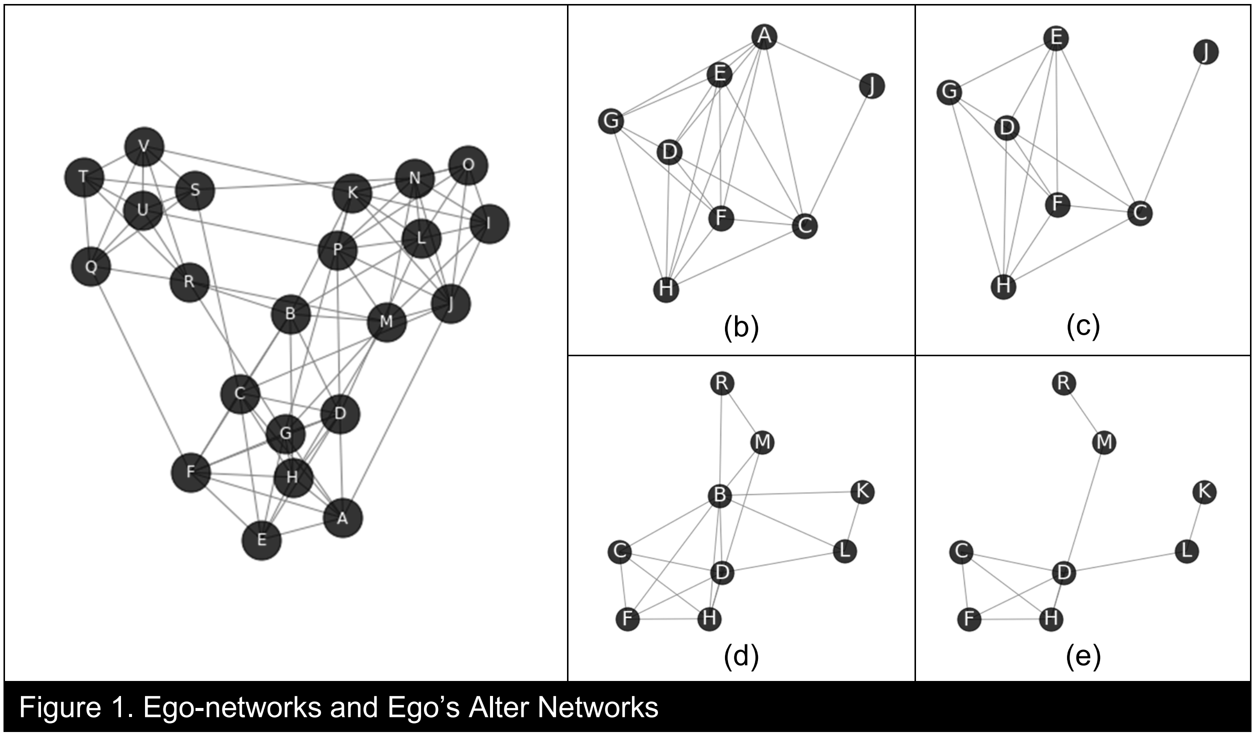
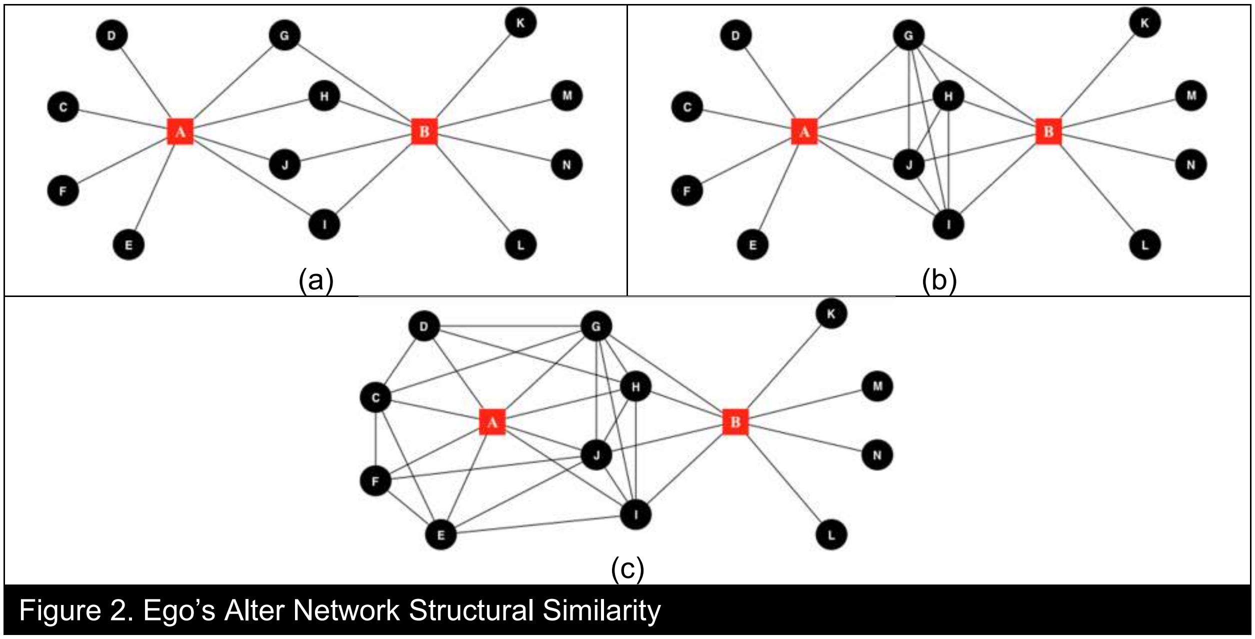


Figure 1. Ego-networks and Ego's Alter Networks

Figure 1(b) 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 1(c). Figure 1(d) depicts the ego-network for B, and Figure 1(e) represents its ego alter-network. The similarity of two ego's alter-networks $\widetilde{G_A}$, $\widetilde{G_B}$ show how structure of neighbors for two vertex A and B are similar. The following function allows the computation of such similarity (Johnson 1985):

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

where $|V(G)|$ returns the number of vertices and $|E(G)|$ returns the number of edges in graph G. $\left|V(\widetilde{G_A},\widetilde{G_B})\right|$ and $\left|E(\widetilde{G_A},\widetilde{G_B})\right|$ 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 following structural situations in Figure 2.



(a)

(b)

(c)

Figure 2. Ego's Alter Network Structural Similarity

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 2(b), 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 2(b) in comparison with the one in Figure 2(a), as they share a common community. This feature can be captured using the ego's alter-network structural similarity. In Figure 2(c), we see a shift in the community structure, where the vertex A forms the core of a large community, and 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 2(b)) as the vertex A is more representative of the community than the vertex B.

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.

Step1. 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 helps the creation of meaningful embedding features. Figure 3 shows a number of randomly selected core vertices in a graph.

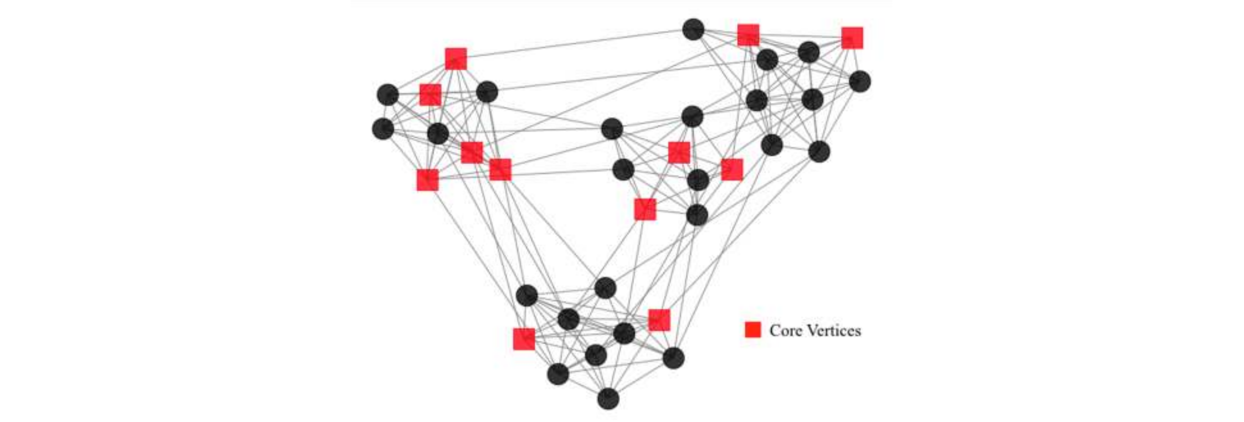


Figure 3. Randomly Selected Core Vertices

Step2. 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: (i) the original graph without the core vertices, and

(ii) 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.

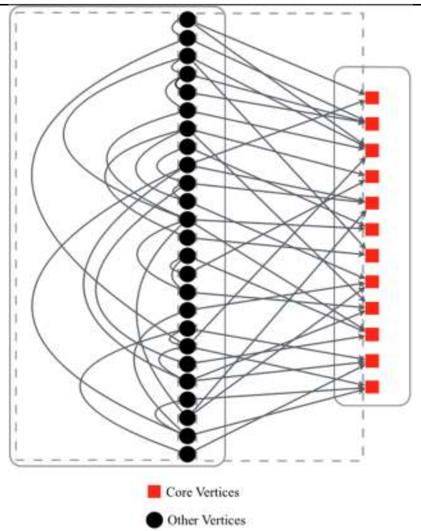


Figure 4. Extended Bipartite Graph

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. In doing 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.

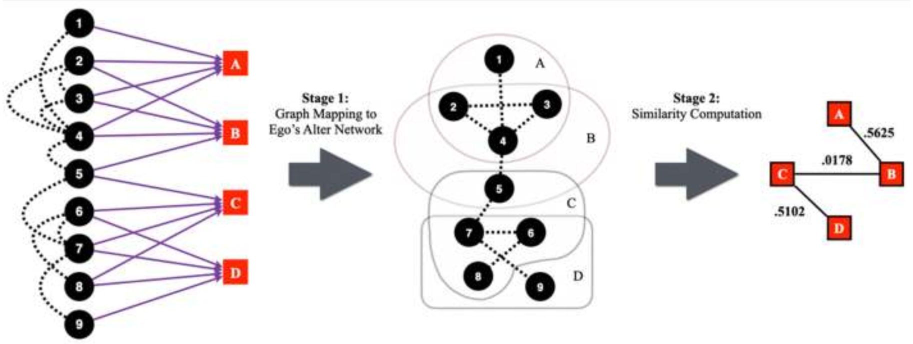


Figure 5. Network Simplification

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 and 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 and Barthelemy 2007, Aldecoa and 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 and 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 and Marín 2011). Using this approach, an optimal set of clusters can be detected in a binary (non-weighted) 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 intra-cluster edges in a given cluster, and p is the total number of observed intra-cluster edges in that cluster. However, 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 intra-cluster edges ($q = \frac{p}{n}$) and relative number of expected intra-cluster 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 and Leibler 1951) that measures the distance between 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 intra-cluster 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).

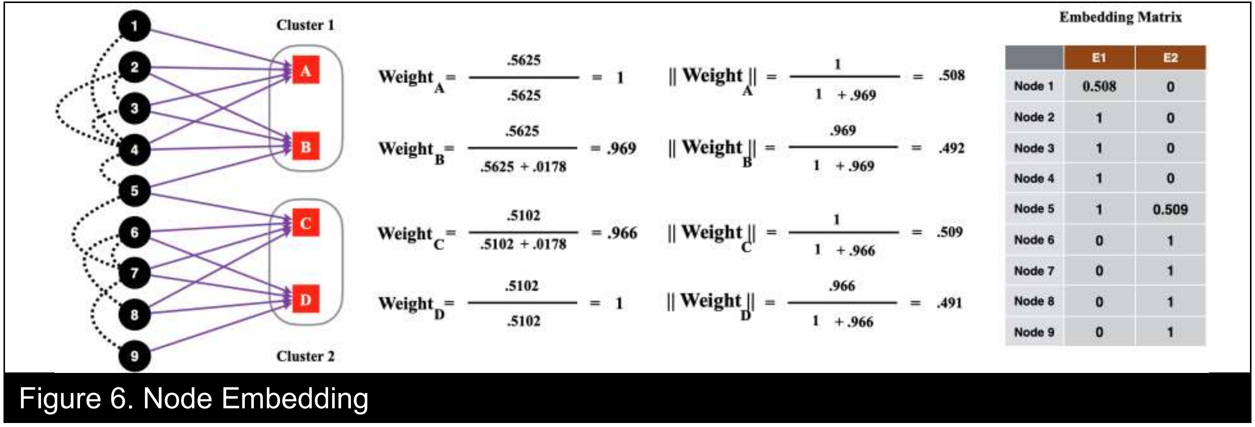
In our method, we use the Louvain greedy approach (Blondel et al. 2008) to optimize the weighted version of asymptotical surprise function for the purpose of network clustering. Applying the algorithm to the graph in Figure5 forms two clusters (C1 = {A, B}, C2 = {C, D}).

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. In fact, 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 in 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 non-core members, this computation can also be used to capture the level of connectivity of cores to different clusters.

also be used to capture the level of connectivity of cores to different clusters.



$$Weight_A = \frac{.5625}{.5625} = 1 \qquad \|Weight_A\| = \frac{1}{1 + .969} = .508$$

$$Weight_B = \frac{.5625}{.5625 + .0178} = .969 \qquad \|Weight_B\| = \frac{.969}{1 + .969} = .492$$

$$Weight_C = \frac{.5102}{.5102 + .0178} = .966 \qquad \|Weight_C\| = \frac{1}{1 + .966} = .509$$

$$Weight_D = \frac{.5102}{.5102} = 1 \qquad \|Weight_D\| = \frac{.966}{1 + .966} = .491$$

Embedding Matrix

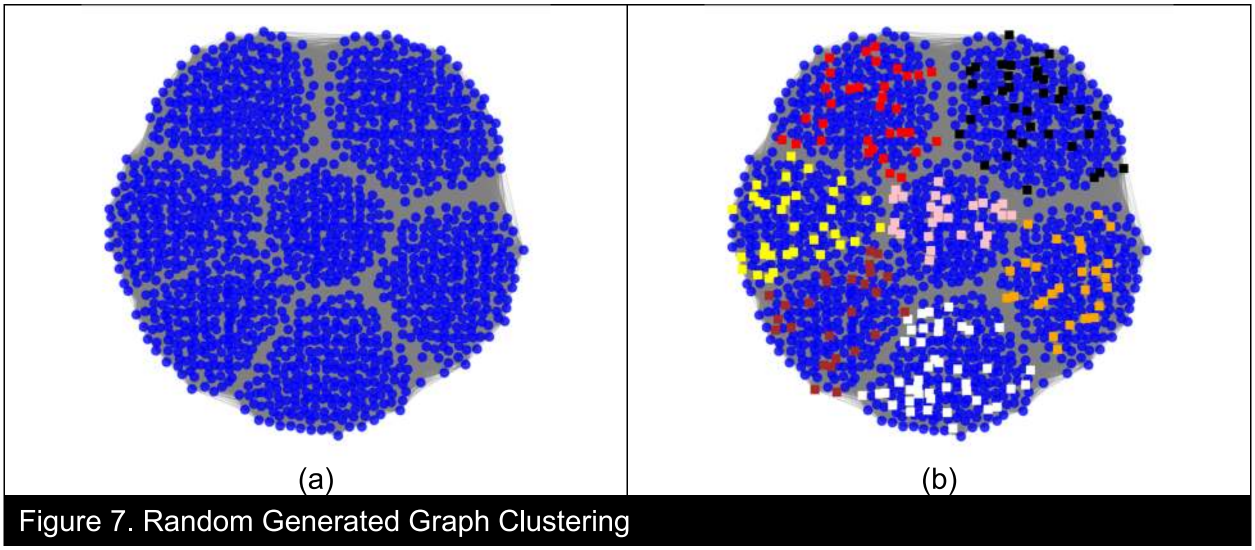
	E1	E2
Node 1	0.508	0
Node 2	1	0
Node 3	1	0
Node 4	1	0
Node 5	1	0.509
Node 6	0	1
Node 7	0	1
Node 8	0	1
Node 9	0	1

Figure 6. Node Embedding

The pseudocode for the HUE algorithm appears in **Appendix A**.

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 7 clusters, where the probability of the formation of inter-cluster links is 0.4 and the intra-cluster is 0.1. Figure 7(a) shows this randomly generated graph.


(a)

(b)

Figure 7. Random Generated Graph Clustering

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 5(b) shows the result of core clustering step for our experiment. It is clear that core vertices well represent the different parts of the graph. Next, we used the identified core vertices to measure the node embedding metrics for all the vertices in the graph. In order to see how embedding features are representative, we applied TSNE dimension reduction approach and projected the embedding features to two dimensions. Figure 8 visualizes the projection result.

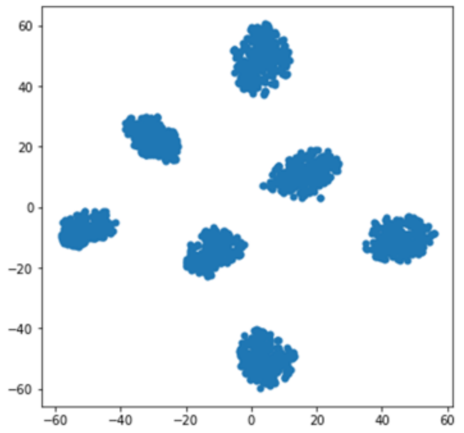


Figure 8. TSNE Dimension Reduction for Randomly Generated Graph

Figure 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 this resulted in seven clusters. Figure 9 shows the outcome of the clustering task with K-means.

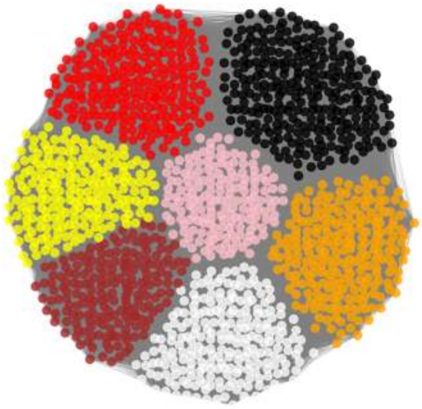


Figure 9. K-means Clustering in Randomly Generated Graph

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 embeddings can then be performed for all nodes on the basis of their pattern of connectivity to the core vertices. Our analysis shows that 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 re-estimation of core clusters.

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 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 those mass communication channels which reinforce their own views and are in agreement with their own preferences and thinking. Therefore, it is expected that in interaction in an online social network, individuals will follow the social pages[4] that promote their own views and are consistent with their preferences. Additionally, social pages in online social networks have high

[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 interest.

level of in-degree centrality making them good choices for core vertices. Prior research in IS field has considered users with 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.

Empirical Application

In order to demonstrate the utility of HUE method, we apply it to a real-world dataset of users in Twitter. Twitter is a popular social media platform for targeted advertising. The unique structure of Twitter allows for identification of new trends and allows 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 precise targeted campaigns, thereby reducing costs associated with wasted promotion effort. It can also be incorporated into recommender systems directly for a friend recommendation (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.

User Population and Data Set

To conduct the empirical application, we use a dataset of more than 32,000 individuals across the U.S. who shared their location-based Foursquare check-ins within the Twitter platform. The dataset spans a six-month period in early 2014 and contains a social network of users, along with the pattern of social pages they followed in 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 dataset appear in Table 1. The sparsity measures the extent of interconnectedness among users relative to a fully connected network. At 0.038%, it indicates that users are reciprocally connected to relatively few other users, with an average of 6 connections apiece. A similar measure of sparsity between users and social pages is derived, and this is 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.

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

Identification of Core Clusters

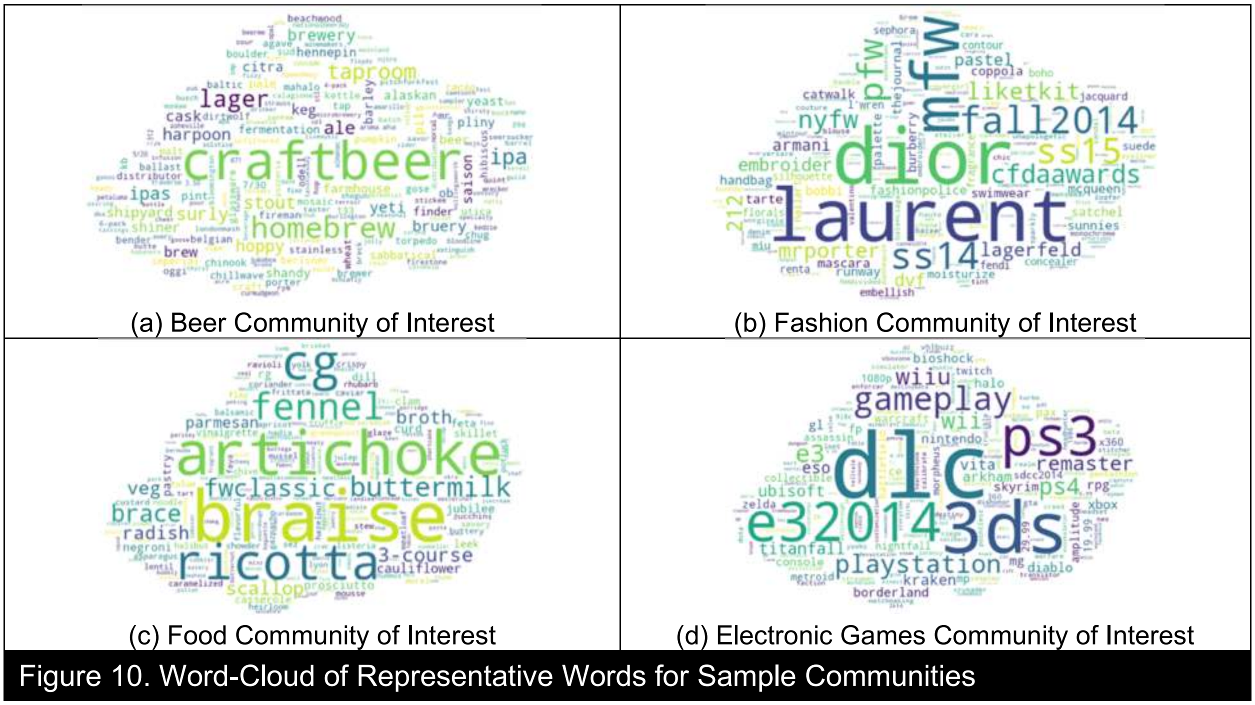
As discussed earlier, we considered the social pages to be the core vertices of the social graph that represent different parts of the graph and show individual preferences toward certain topics. As a result, the extended bipartite graph in HUE consists of two parts: (1) a social network of users with 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

[5] In this research we used reciprocated relationships between users to form the social network portion of the graph. Reciprocated relationship is formed between two users when both users follow each other in the network. That is an indication of 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 for 500 times to select the one with best quality.

community of interest can be represented by that 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 are commonly used by 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 come up with a label for each of the communities. Figure 10 shows word-clouds of representative words for a sample of communities.



(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

A detailed analysis of communities shows that while some communities represent general preferences of individuals like music or fashion, others have a much narrower focus like a TV show or a brand. **Appendix C** shows the list of prominent communities of interest that were identified along with some sample of social pages within those communities.

We then visualize the position of social pages (core vertices) in the social graph by structurally

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

positioned them where they have a larger number of followers. We used the ForceAtlas2 layout (Jacomy et al. 2014) in Python to accomplish this task. ForceAtlas2 is 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 proves challenging, and we have selectively depicted some communities. Visualization of a single community provides little information other than the followers of a particular set of pages in that community. On the other hand, visualization of related and complementary communities of interest provides far greater insight. Individuals who are structurally positioned close to social pages belonging to a community of interest have a greater affinity toward that community of interest. In addition, social pages that appear on the periphery of the graph reflect a niche preference among a group of individuals on the social network. Reading across the figures affords a clearer interpretation of individual preferences amid communities of interest. For example, it can be seen that individuals with a strong interest in fashion are less likely to have strong beer preferences and vice versa.

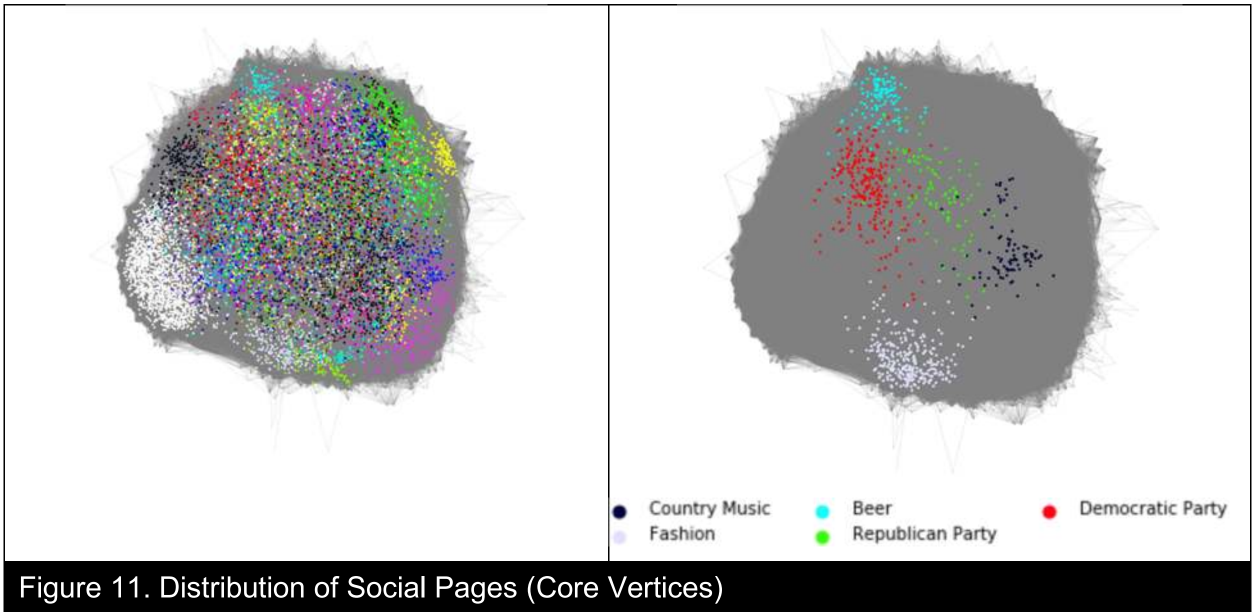
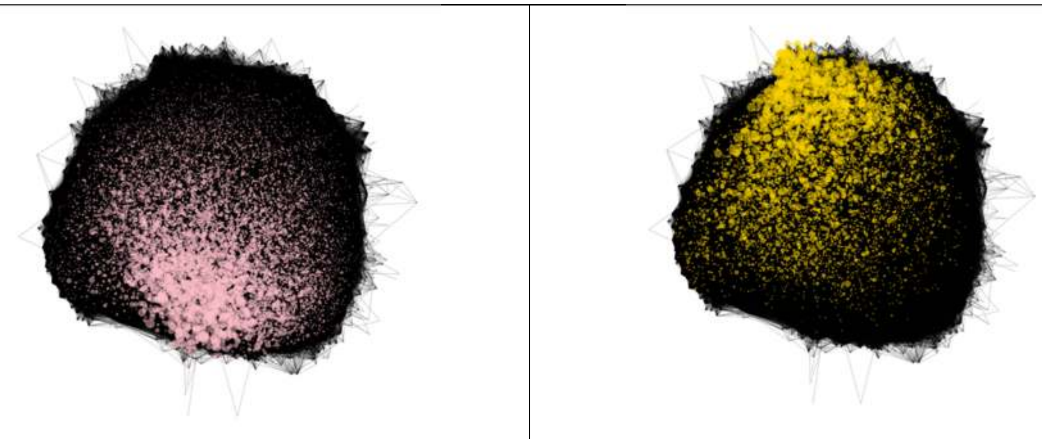


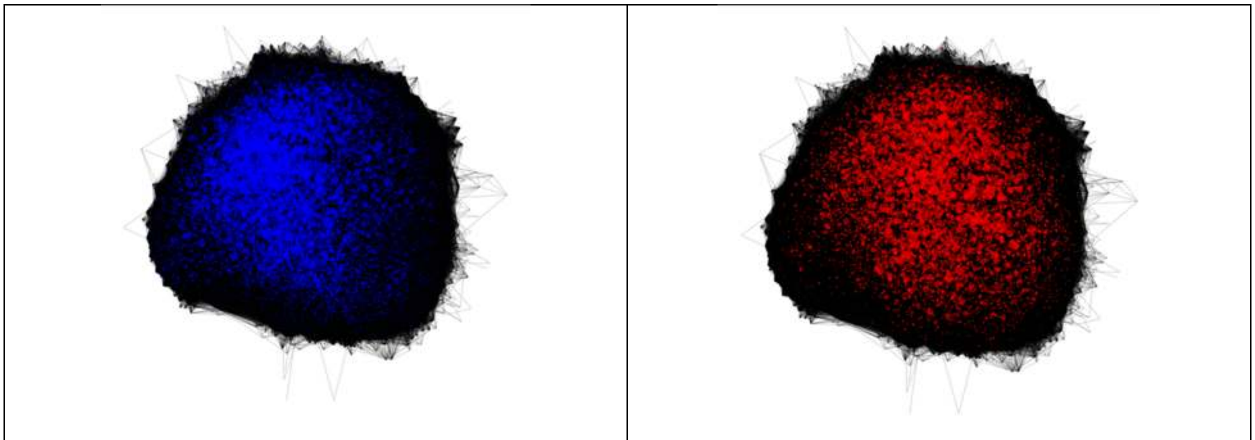
Figure 11. Distribution of Social Pages (Core Vertices)

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 towards 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. Thus, for example, it can be deduced that there is little overlap of individuals between the beer community and the fashion community. But there is considerably more overlap between the beer community and a political party. 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 from the sample data set. New individuals need to be embedded in the graph, based on their interactions and following of social pages.



(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-EmeddingFeature Disttribution in Online Social Network

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 ten 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. Since the number of preferences is a large number, we simply report the coefficients of the top 5 preferences with the highest absolute values for each check-in category. Since all preferences are 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.

Table 2. Preferences and Location Check0in 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

Notes:
1. All the reported coefficients are significant at .001 level
2. We use all the preferences as independent variables, but only report top 5 with highest absolute coefficient values

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.

Application of HUE in Link Prediction

One of the main applications of node embedding is link prediction (Grover and Leskovec 2016). Link prediction can have utility in a number of different purposes. 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 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 recommender systems (Xie 2010, Aiello et al. 2012). Link prediction approaches generally fall into two categories: (i) similarity-based, and (ii) 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 rank assigned scores in a decreasing 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 a learning-based link prediction and use a deep learning model to illustrate the predictive power of embedding features.

Deep Learning Model for Link Prediction

In order to perform link prediction, we considered two-way relationships among users to form 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 removals is 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 dataset was no exception. Accordingly, for each established link in train, holdouts, and test sets, we randomly selected ten unestablished links and use them as negative samples in our work. This approach limits the ratio of established links to unestablished links to be 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 and Leskovec 2016), and LINE (Tang et al. 2015) to the training social graph.[9] We considered the 256 features extracted from 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 will vary based on the hyperparameters selected, we used different sets of hyperparameters and compared their performance. Table 3 lists the used hyperparameters for each of the algorithms.

[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 on the same level 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 2-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 highest number of cores to compute embedding factors

For link prediction, we need to work at the edge level rather than at the node level. We follow Grover and Leskovec (Grover and Leskovec 2016) and use 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.

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	$\|\|.\|\|_{\bar{2}}$	$\|\| f(u) . f(v) \|\|_{2i} = \| f_i(u) - f_i(v) \|^2$

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. 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 rely 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 over-fitting, 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 and 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 inclined to be connected with other users. In our dataset, even after controlling for the ratio of linked users to unlinked ones, the

[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.

dataset still remained imbalanced (1:10). Prior studies suggested two approaches to tackle imbalanced datasets: (i) assign weights to classes (Ling and Sheng 2011, Mirza et al. 2013), and (ii) 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:

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

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

[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.

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.

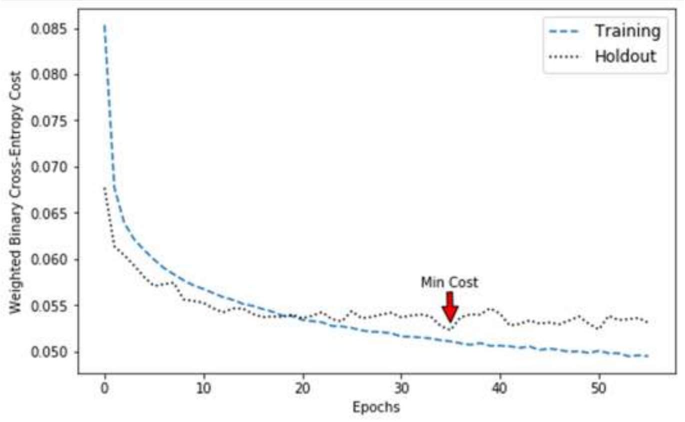


Figure 13. Example of Model Selection Process

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) for five times. 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 5 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.

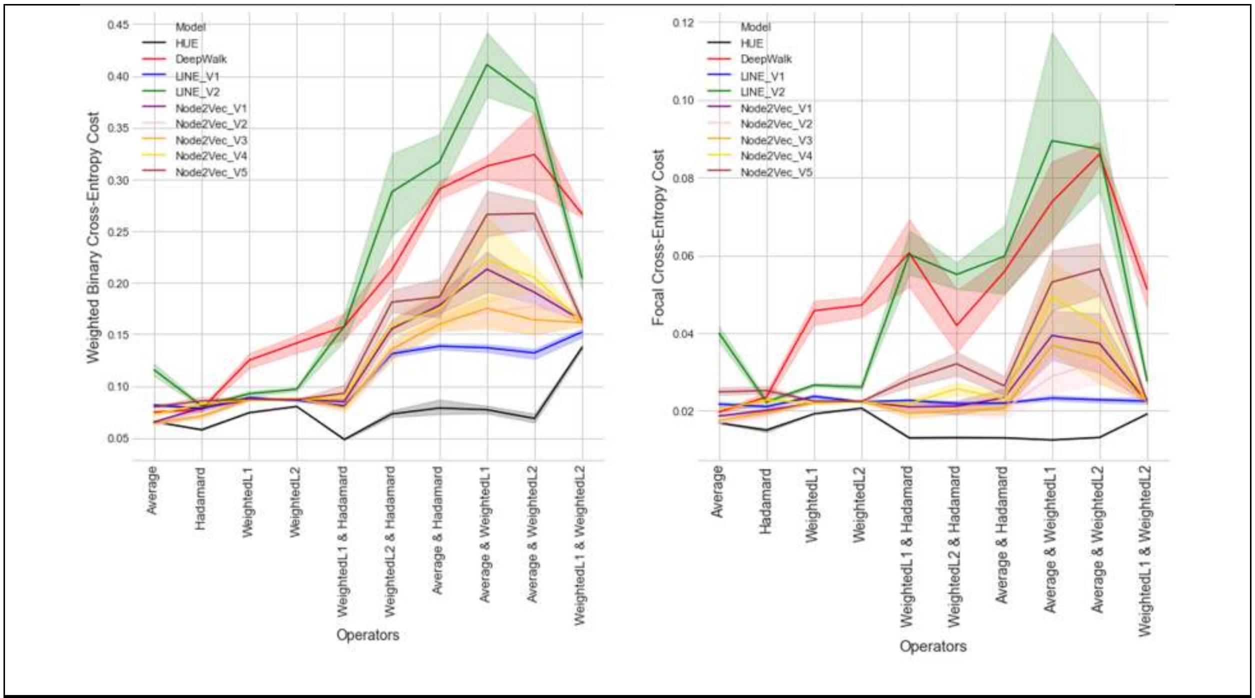


Figure 14. Model Costs

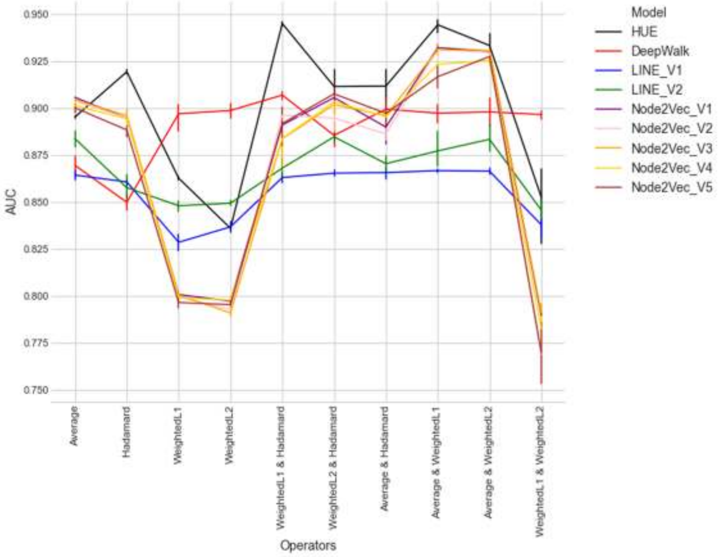
The result shows that the embedding features identified by our proposed algorithm result in lowest costs across all operators and have performed best when using both the WeightedL1 and Hadamard feature sets.

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, examined all connected individuals, and determines the fraction of those that were 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 group of users. AUC is a more involved measure that represents the area under the curve of the Receiver Operating

[13] Precision, Recall, and F1 measures are all computed at the threshold value of .5.

Characteristics curve that relates the true positive rate and the false positive rate. It ranges between zero and one, with a score of 0 representing complete misclassification, 0.5 represents 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 means scores, and the bands represent the ranges for the 5 runs for each of the algorithms.

Weighted Binary Cross-Entropy Cost



Focal Cross-Entropy Cost

[14] We did not report accuracy, as accuracy provides biased measure of performance for imbalanced datasets.

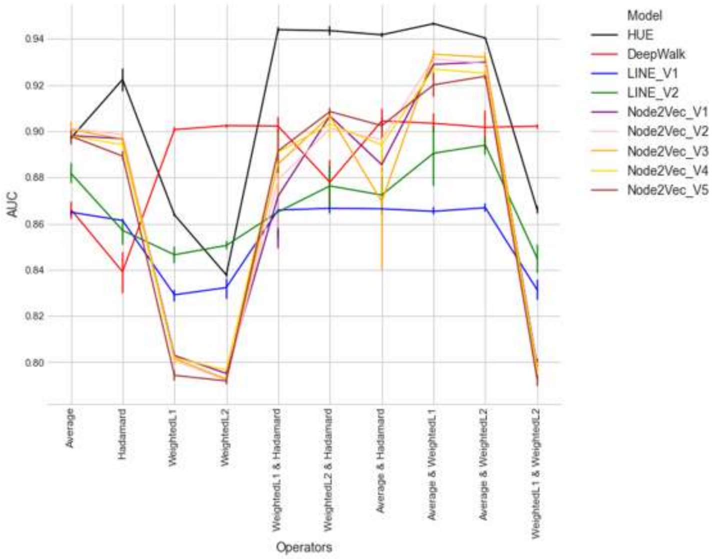


Figure 15. AUC Metric for Weighted and Focal Cross Entropy

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 and 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 non-existent link. The results for the AUC metric show that our proposed algorithm outperformed other algorithms when we used variables from Hadamard operator, and combination of two operators (except from WeightedL1 & WeightedL2). The AUC value for the case that we use both the Hadamard and WeightedL1 variables is the highest AUC of all possible cases with a value close to .95 indicating strong predictive power of our algorithm for link prediction. For the Average Operator, our algorithm is in second position with a slight difference from the Node2Vec embeddings. Deep Walk has outperformed our algorithm for Weighted-L1 and Weighted-L2 and combination of these operators. LINE_V2 has also outperformed our algorithm for Weighted-L2 operator.

The precision score is one of the metrics that our algorithm has relatively lower performance on it. However, considering 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 fluctuates between 0.3 and 0.8, reaching its highest point at around 0.8 when we use both Average and WeightedL1 operators together in weighted binary cross-entropy cost models.

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, to 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 an online social network is truly staggering, indicating a large active user population. 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, and tend to respond and forward messages quickly. This velocity can be exploited by marketers to reach targeted users in a cascading manner rather than 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 the 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 guarantee about the appropriateness of the selection, or the effectiveness of the message. Mechanisms that allow marketers to identify appropriate users from publicly available profiles provide the marketer 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 the 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 argued that the right embedding of the structure of online social networks could 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 the embedded features in their recommender systems. In addition, the techniques can 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 interests and use it to refine the experience of users in terms of delivering appropriate content with lowered odds of serving up irrelevant content and omitting pertinent content. For advertisers, this knowledge allows them to develop and serve up targeted ads to small cohesive groups, thereby reducing the effort in ad creation, while diminishing the risk that the ads are viewed as irrelevant. In addition, social network users also benefit in that their most precious commodity, viz. time, can be spent more effectively, while also being subjected to a smaller barrage of information.

This paper introduced 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. In doing so, we have taken 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 lack of ability to derive meaningful dimensions from social networks significantly limits their application. Our study addressed this gap by using the semantic information present on certain nodes in online social networks. Our research is in line with the selective exposure theory (Sears and Freedman 1967, Zillmann and 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 used 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 recommender systems.

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 the best of our knowledge, HUE is the only node embedding approach that allows extraction of meaningful variables from the structure of online social networks. The extracted variables include demographic as well as user preference information. It demonstrates 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. This allows platform managers, content providers, and advertisers to target individuals effectively and reduce the cost of 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 intervention. Thus, for example, social marketing programs can benefit from result 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 can be adopted in recommender systems in online social networks to identify relevant social pages and potential friends to individuals based on their preferences. It is also useful in setting up recommendations for new entrants to the social network. In addition, it can 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 are able to identify various communities of interests on their platform and evaluate the risks and benefits associated with each community through assessment of posted content within that community.

Cyber security 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. Identification of communities of interest facilitates the process of detecting these pages within online social networks. Our exploration of data shows that some communities of interest contain social pages which were recently suspended by Twitter. Paying additional attention to other members of such communities allows platforms such as Twitter to identify potentially troubling social pages more quickly and accurately.

Fifth, the extracted user preferences can 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 U.S. receive news from partisan communities of interest that tend to reinforce their own political ideology. This finding affords policy makers the opportunity to monitor and track political communities of interest. However, this can 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

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

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 can benefit from HUE by applying it to their context. For example, movie-recommender systems can 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 can quantify individual preferences for different types of movies. In summary, HUE is a novel approach that allows extraction of information based on the interactions of two different entities.

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 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 will indicate stronger relations than are present in the general population. Conversely, a very diffuse sample will mask some clear relationships. Future studies will expand our work by addressing the implications of different strategies for selection of core vertices. In addition, the proposed method is developed for undirected graphs. This ignores weak relationship of individuals in online social networks, which is quite common in some networks, especially Twitter. Another area where 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 who the interactions are with. Finally, the case where multiple social network users share an account will likely trip up the homophily-based approach, as it would most other interaction-based approaches.

Future extensions of our work involve the use of HUE in content recommender systems. This would allow recommender systems to suggest the right content and topics to new users thereby improving the quality of interactions with the platform.

References

Ahmed A, Shervashidze N, Narayanamurthy S, Josifovski V, Smola AJ (2013) Distributed large-scale natural graph factorization. In Proceedings of the 22nd international conference on World Wide Web 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 LM, 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) p.e24195.

Aldecoa R, Marín I (2013) Surprise Maximization Reveals the Community Structure of Complex Networks. *Scientific reports* **3** p.1060-1073

Almansoori W, Gao S, Jarada TN, Elsheikh AM, Murshed AN, 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 PH, 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. *In Eleventh International AAAI Conference on Web and Social Media*.

Belkin M, Niyogi P (2002) Laplacian eigenmaps and spectral techniques for embedding and clustering. *In Advances in neural information processing systems*. 585-591.

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

Blondel VD, Guillaume JL, Lambiotte R, Lefebvre (2008) Fast Unfolding of Communities in Large Networks. *Journal of statistical mechanics: theory and experiment* 2008(10) P10008.

Cai H, Zheng VW, Chang KCC (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. *In Proceedings of the 24th ACM international on conference on information and knowledge management* 891-900.

Chen J, Stallaert J (2014). An economic analysis of online advertising using behavioral targeting. MIS Quarterly **38**(2) 429-A7.

Cheng R, Pang J, Zhang Y (2015) Inferring friendship from check-in data of location-based social networks. In 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. In IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 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 JH, Spangler WE (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. In Proceedings of the thirteenth international conference on artificial intelligence and statistics 249-256.

Goldfarb A, Tucker CE (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. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining 855-864.

Gu B, Konana P, Raghunathan R, Chen HM (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 JA, McNeil BJ (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 NS (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) e98679.

Johnson M (1985) Relating metrics, lines and variables defined on graphs to problems in medicinal chemistry. In Graph theory with applications to algorithms and computer science 457-470.

Kullback S, Leibler RA (1951) On Information and Sufficiency. *The annals of mathematical statistics* **22**(1) 79-86.

Kossinets G, Watts DJ (2009) Origins of Homophily in an Evolving Social Network. *American Journal of Sociology* **115**(2) 405-450.

Lee DD, Seung HS (2001) Algorithms for non-negative matrix factorization. *In Advances in neural information processing systems* 556-562

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 TY, Goyal P, Girshick R, He K, Dollár P (2017) Focal loss for dense object detection. In Proceedings of the IEEE international conference on computer vision 2980-2988.

Ling CX, Sheng VS (2011) Cost-sensitive learning and the class imbalance problem. Encyclopedia of Machine Learning: Springer, **24**.

McPherson M, Smith-Lovin L, Cook JM (2001) Birds of a Feather: Homophily in Social Networks. *Annual Review of Sociology* 415-444.

Mirza B, Lin Z, Toh KA (2013) Weighted online sequential extreme learning machine for class imbalance learning. Neural processing letters **38**(3) 465-486.

Newman ME, Girvan M (2004) Finding and Evaluating Community Structure in Networks. *Physical review E* **69**(2) 026113.

Niepert M, Ahmed M, Kutzkov K (2016) Learning convolutional neural networks for graphs. In

International conference on machine learning. 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. 701-710.

Ramachandran P, Zoph B, Le QV (2017) Searching for activation functions. arXiv preprint arXiv:1710.05941.

Ribeiro LF, Saverese PH, Figueiredo DR (2017) struc2vec: Learning node representations from structural identity. In Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining 385-394

Rollins B, Anitsal I, Anitsal MM (2014) Viral Marketing: Techniques and Implementation. *Entrepreneurial Executive* **19**.

Roweis ST, Saul LK (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science* **290**(5500) 2323-2326.

Sears DO, Freedman JL (1967) Selective Exposure to Information: A Critical Review. *Public Opinion Quarterly* **31**(2) 194-213.

Shi Z, Whinston AB (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. In Proceedings of the 24th international conference on world wide web 1067-1077

Tenenbaum JB, De Silva V, Langford JC (2000). A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science* **290**(5500) 2319-2323.

Traag VA, Krings G, Van Dooren P (2013). Significant Scales in Community Structure. *Scientific reports* **3**(2930)

Traag VA, Aldecoa R, Delvenne JC (2015). Detecting Communities Using Asymptotical Surprise. *Physical Review E*, **92**(2), 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. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining 1225-1234

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

Yan J, Liu N, Wang G, Zhang W, Jiang Y, Chen Z (2009) How much can behavioral targeting help online advertising? *In Proceedings of the 18th international conference on World wide web* 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* 157–190 Hillsdale, NJ: Lawrence Erlbaum Associates.

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.

```
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. Homophily-based User Embedding 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.

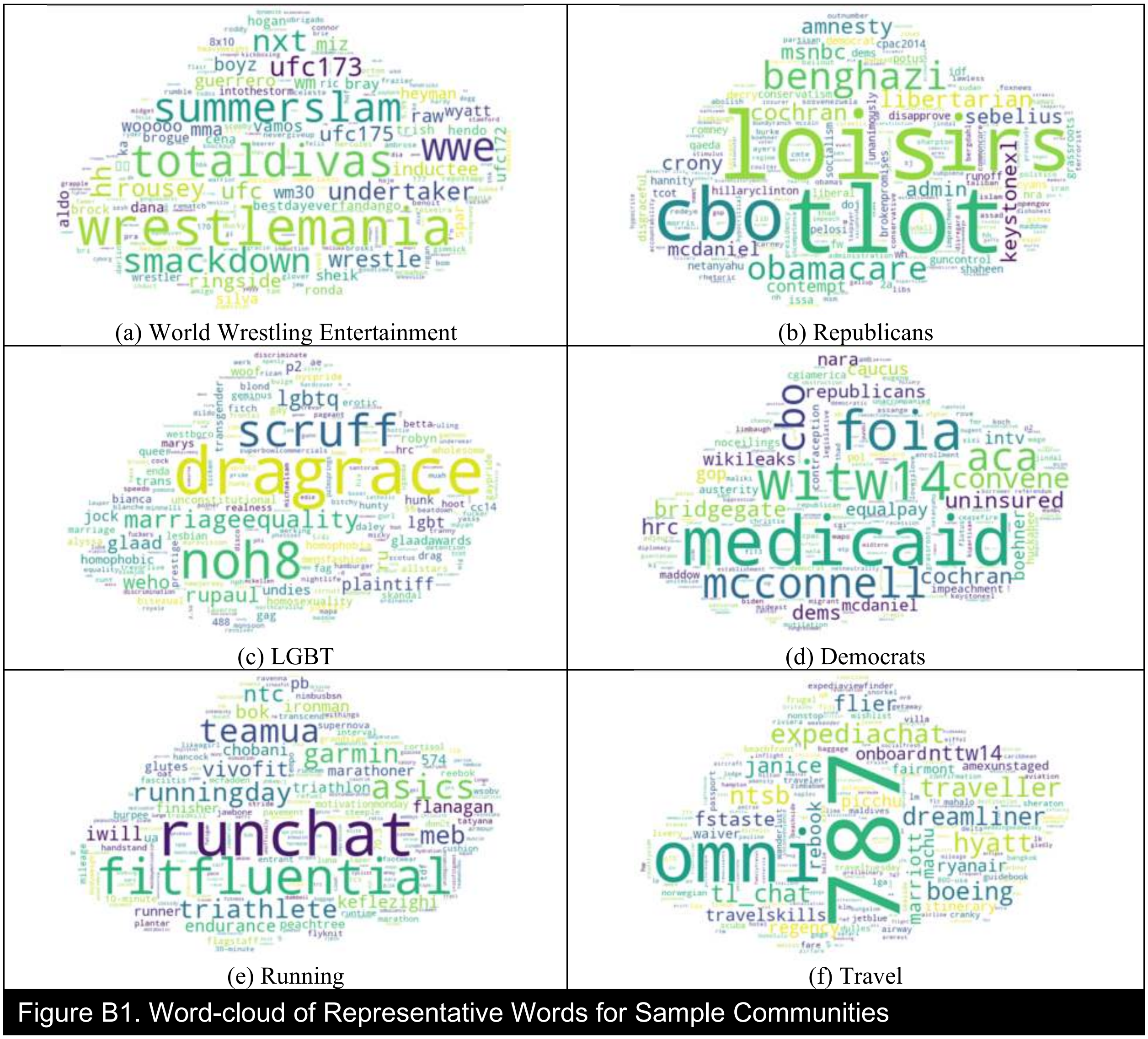
Step 3. Our pre-processing activities comprised tokenization, lemmatization, and the elimination of stop words. We then picked the top 5000 high-frequency terms from the first corpus to develop the vocabulary set. This vocabulary set includes both representative terms of the community of interest and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

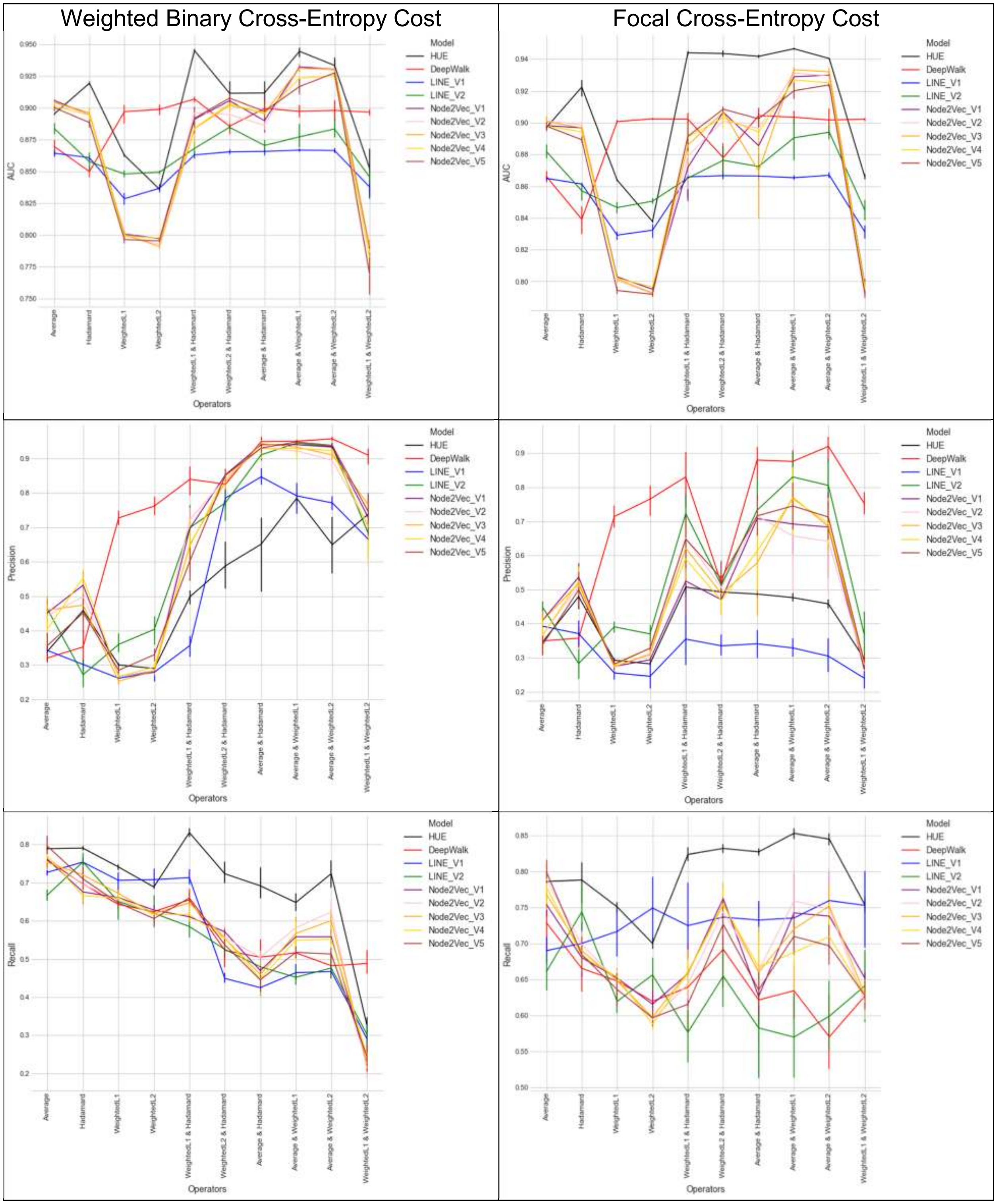
Art and Design	Geographical	TV Shows and Movie Series	Movie
- Art & Design - Photography - Web Design	- Atlanta - Austin - Baltimore - Boston - Cleveland - Chicago - Dallas - Detroit - Denver - Houston - Las Vegas - Los Angeles - Miami - Milwaukee - Minnesota - New Orleans - New York - Philadelphia - 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		**Sport**	**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 community of interests.

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

Appendix D – Performance Metrics for Link Prediction Task



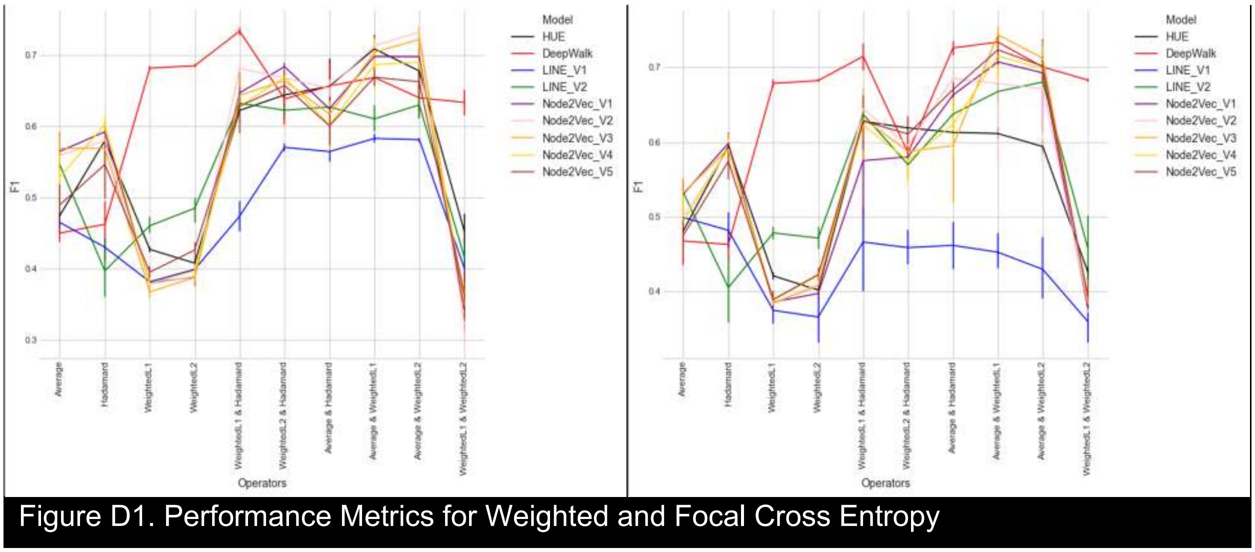


Figure D1. Performance Metrics for Weighted and Focal Cross Entropy

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

Author Biographies

Mahyar Sharif Vaghefi is Assistant Professor in 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 Associate Professor of Information Technology Management at the Lubar School of Business at 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 Program Chair for AMCIS, Treasurer for ICIS, and is a charter member of AIS.