Evaluating Network Embedding Models for Machine Learning Tasks

Ikenna Oluigbo, Mohammed Haddad and Hamida Seba

Université de Lyon, CNRS, Université Lyon 1
LIRIS, UMR5205, F-69622 Lyon, France

Abstract. Network embedding is a representation learning paradigm that seeks to learn a compact low-dimensional distributed vector representation for each vertex in the network; this learned low-dimensional vector representation can thus be used for different machine learning tasks. Over the years, so many network embedding models have been worked upon based on several approach-es. In this paper, we study vector embeddings of 10 different representation learning models, with the sole aim of carrying out two machine learning tasks on these learned representations – unsupervised community clustering and link prediction analysis. The goal is to compare the output of these tasks using the 10 models, and draw inference based on the obtained results. We analyze the results using 4 link prediction baseline heuristic measures for the link prediction analysis; and a combination of silhouette score analysis and dissimilarity metric index for the community analysis.

Keywords: Link Prediction, Clustering, Network Embedding, Graph Learning.

1 Introduction

In many real-world scenarios today, graphs have been used to represent important information. Representations as graphs have been found to be extremely useful, in that, intricate attributes and properties of these entities can be fully explored and intelligently mined, therefore allowing for accurate comparison between them. A social network, for example, encompasses individuals and groups who are tightly knitted together as friends and families to form a very dense graph structure; but while family members (vertices) might share direct connection (edges | links) amongst themselves (vertex neighbors), certain individuals in the network might be loosely tied to one another but may still share many other common attributes together.

Traditionally, a graph or a network can be represented in memory in the form of Adjacency Matrix (AM), Incidence Matrix (IM), Adjacency List (AL). However, these forms can pose problems for machine learning tasks in that they belong to a very high dimensional space and largely sparse in nature which can make their processing very complex. To address these problems, especially the sparsity issue, researchers over the years have developed representation learning embeddings to represent graphs. Network embedding is a representation learning technique that aims at mapping high dimensional networks into a latent low dimensional vector space, while preserving the information and properties of the original network [6]. Network embedding models are

mostly unsupervised, with the possibility of retaining various information and attributes such as structural dependency and homophily [3, 5, 6].

Recently, there has been so much interest in the domain of graph representation learning and performing series of tasks with the resulting vector embeddings. Using Skip-gram, a word representation model in natural language processing, [5] proposed DEEPWALK which uses graph local properties obtained from truncated random walks to learn latent representations of vertices in a structural network encoded in a continuous vector space. However, the idea of truncated random walks was further modified in [1] by skipping over steps in each random walk to generate a dictionary of vertex pairs from fixed length paths. Bandyopadhyay et al. in [6] proposed a nonnegative matrix factorization-based framework - FSCNMF which toggles between the network's structure and the properties of each node while learning low dimensional representation of each node in the network. Furthermore, Grover and Leskovec in [3] proposed node2vec which uses a number of tunable hyperparameters (In-out parameter (p & q)) to embed nodes in low-dimensional space while preserving the neighborhood properties of the nodes.

The efficacy and trustworthiness of a network embedding can only be judged when evaluated against other existing state-of-the-art techniques, with the result compared with baseline heuristic scores. With different approaches being; employed in learning low-dimensional vector embeddings for nodes in the network, applying these embeddings on machine learning tasks would give a clear judgement when drawing inferences for the suitability of each model for different tasks. In this paper, we developed an algorithmic framework to compare 10 network embeddings models on two machine learning task – link prediction and community detection. As a contribution, we tried to show how the 10 state-of-the-art embedding models perform in predicting missing nodes (since each model learns low-dimensional vector embeddings for similar nodes), and to evaluate how well the nodes would cluster for each distinct community.

2 Related Work

Existing graph learning models can be classified according to the following categories: **Random Sampling Approach:** Sampling based techniques have been widely adopted by a number of researchers. The node2vec embedding model of Grover and Leskovec in [3] uses a search biased 2^{nd} order random walk paradigm in graphs to explore diverse neighbors in the network, and generate network neighborhood representation for nodes. The model learns the best network embeddings of nodes based on their functional roles or in communities each node belongs. Owing to the popularity of the Skip-gram model especially in NLP [7], DEEPWALK [5] uses the uniformly truncated random walk from a node to a randomly selected neighbor, until the predefined number of walk length is reached. Similar to NLP where a corpus is represented by sequences of words, DEEPWALK tries to represent a network corpus as sequence of nodes. DEEPWALK learns representations from a network without considering any other vital node information during its random walk [8]. In contrast to [5], the model in [1] known as WALKLETS, learns multiscale nodes relationships by subsampling short random walks through every node in the network. Like in [5], the walk terminates when a maximum walk length is reached. GRAREP, a learning representation model proposed by

[9] embeds weighted graphs. The model leverages on DEEPWALK [5] and the Skip-gram model [7] to learn low-dimensional vectors by integrating the global structural information of the graph as part of the learning embedding process. The learned global representation from the model was applied to clustering, classification and visualization tasks as part of its evaluation.

Deep Learning Based Embedding: Several network embedding models have leveraged on deep learning techniques to learn representations of vertices. This is shown in the study of [2], which propose HIVEC, a model that learns latent vector representations by capturing semantic dependencies of co-occurrence information in graph sub-structures. While this learning process can be likened to a deep-learning model, the hierarchical embedding technique involved allows vertices of a network to be embedded using the similarities in distances between sub-structures. The model was evaluated on a graph classification task. Similarly, a deep-learning embedding technique proposed by [4] and called DANE, learns network attributes and dynamically evolving patterns in a dynamic network. According to [4] the model was proposed to effectively tackle the problem of noisy and incomplete node patterns as well as a constantly changing graph structure. DANE leverages matrix perturbation theory to maintain a real-time network embedding as the network changes in structure.

Matrix Factorization Based Embedding: Matrix factorization performs well at maintaining node properties and the underlying semantics of a network. Such is seen in the work of [6] which proposes a nonnegative matrix factorization framework (FSCNMF) to learn a low dimensional vector representation for each node in the network while maintaining the network structure and node attributes. Learned representations from the model was implemented on machine learning tasks such as node clustering, visualization and multi-class classification for datasets with limited number of nodes. Furthermore, the study of [11] proposed a network embedding model (BANE) to learn binary node representations through a Weisfeiler-Lehman proximity matrix to capture data dependence between node links and attributes following a layer-wise approach of gathering attributes and links from neighbors of a node; this process continues until all nodes are explored. The learned representations were experimented on node classification and link prediction tasks. The results from our study will be compared with the results derived from the study of [11].

Heat Wavelength Diffusion Pattern Embedding: The study of [10] propose the GRAPHWAVE model that learns representations for each node neighbor in a network, by leveraging heat wavelet diffusion patterns. According to [10], the model's runtime performs linearly with the number of edges, and also possesses the ability of capturing structural roles for nodes in a network in an unsupervised way. Heat wavelets are considered as probability distributions over the graph; this way, the manner with which diffusion spreads over the network is captured [10].

In our study, we propose to evaluate the learned representations from 10 network models on two machine learning tasks, namely link prediction and community clustering. We also seek to find out which of the four categories performs better over each task. Conclusions will be drawn from the derived results.

3 Analysis

3.1 Preliminaries

Punj and Stewart, in [12], describe cluster analysis as a statistical and purely empirical method for classification with no prior assumptions about important differences within a sampled population; making it an inductive technique. The endpoint of every clustering process is to classify a sample of a target population (or objects) on the basis of a set of measured variables into a number of different groups such that similar samples (subjects with common attributes) are usually placed in a homogenous and distinct group. Generally, clusters should exhibit high internal homogeneity and high external heterogeneity. A few authors have carried out different experiments using clustering analysis with varying results. Setyaningsih in [13] examines the successful performance of Small and Medium Enterprises (SMEs) using cluster analysis to map the pattern of growth mode and strategies. The study of [15] proposed GEMSEC, a graph embedding algorithm which learns a high-quality clustering of the nodes simultaneously with computing their features on real world social networks.

Clustering task requires a proper measure for proximity; this can either be through (1) A similarity measure $s(a_i, a_j)$ where the similarity measure increases if a_i, a_j are similar. (2) dissimilarity measure (also known as distance measure) $d(a_i, a_j)$ where the distance measure decreases if a_i, a_j are similar. Also, an ideal algorithm has to be chosen for computing a community clustering by optimizing the criterion function [14].

Link prediction is a machine learning task, basically used to forecast future possible links (edges) in a network, or to predict possible missing links in an incomplete network or incomplete set of data. There actually exist very limited existing studies on link prediction task. Lichtenwalter et al. in [16] studied several factors such as network observational period, generality of existing methods, variance reduction, topological causes and degrees of imbalance, and sampling approaches; and thereafter proposed a supervised flow-based predicting algorithm for sparse network link predictions. Another study in [3] also proposed a link prediction framework to predict missing edges in a network. In their study, a 50% training dataset was retained for the experiment while ensuring that the residual network obtained after the edge removal remains connected.
Definition I: Given a connected graph G = (V, E) with Edges e = (x, y), one can aim to predict missing or incomplete links in the network such that an output similarity score S_{xy} would be considerably high.

Table 1: Heuristic Baseline Scores for Link Prediction in pair of nodes (u, v); $\Gamma(u)$ and $\Gamma(v)$ represents set of neighbors for node u and node v respectively.

Heuristics	Definition
Adamic Adar	$\sum_{\omega \in \Gamma(v) \cap \Gamma(v)} \frac{1}{\log\lvert\Gamma(\omega)\rvert}$
Preferential Attachment	$\lvert\Gamma(v) * \Gamma(v)\rvert$
Jaccard Coefficient	$\frac{\lvert\Gamma(v) \cap \Gamma(v)\rvert}{\lvert\Gamma(v) \cup \Gamma(v)\rvert}$
Resource Allocation Index	$\sum_{\omega \in \Gamma(v) \cap \Gamma(v)} \frac{1}{\lvert\Gamma(\omega)\rvert}$

Heuristics measures are used for evaluating unsupervised link predictions results by using some or all of them as features; with a comparison made between similarity score S_{xy} and one or more heuristics measure scores. Popular heuristic measures commonly used includes Common Neighbors, Adamic Adar Index, Preferential Attachment, Jaccard Coefficient, vertex degree, and Resource allocation index. These heuristic scores are used because they require no training, easy to calculate and interpret, and scales well in performance.

The heuristics measure score identified for use are defined in terms of the neighborhood node set for each pair of nodes (u, v) [3].

3.2 Contributions

As already stated earlier, the aim of this paper is to compare and evaluate the performance of node similarity embeddings from 10 representation learning models using community clustering and a link prediction analysis.

Algorithm 1: Community Clustering

Input: Vector Embeddings Unpack Embeddings
Probability Distribution μ
$\gamma = \mu$ (vectors f, perplexity ρ, learning rate = ℓ)
return γ

1. **Clustering** (γ)
2. Let number of clusters $\Bbbk$ be maximum silhouette score value
3. **for** each $z \in \Bbbk$, initialize r (z) to be randomly chosen point from $\mathbb{D}$
4. **While** changes in clusters $\Delta_{\Bbbk}$ **do**
5. form initial clusters
6. **for** each $z \in \Bbbk$ **do**
7. $\Delta_{\Bbbk} = \{x \in \mathbb{D} \mid \text{distance d } (r_z, x) \leq \text{distance d } (r_n, x) \text{ for all n in } \Bbbk, n \neq z\}$
8. **end for**
9. compute new centroids
10. **Repeat** 6 and 7 until centroid convergence is stable
11. **end**

Output: Community Clusters in 2 dimensions

For the link prediction task, we would like to determine the accuracy with which missing nodes can be predicted using embeddings from each representation learning model. This study proposes a two-step algorithmic technique for community clustering on low-dimensional vector embeddings. The first step is to create a probability distribution that computes the relationships between various nodes neighboring points for each of the vector embeddings. A stochastic method is defined to further render the resulting distribution into a 2-dimensional set of arrays. The second step involves an algorithmic computation of the 2D arrays into defined community clusters for similar nodes. This is shown in algorithm 1.

The idea is that if each model has indeed accurately learned embeddings for similar nodes in a network, these nodes should be clustered together; and the dissimilarity index should be as minimal as possible. Since there are not many existing state-of-the-art

works on link predictions, the result of our study was measured against standard heuristic scores outlined in Table 1.

4 Experimental Design

Experiments were conducted to evaluate the accuracy and effectiveness of network embeddings from ten different representation learning models grouped under Random Sampling approach, Deep Learning base approach, Matrix Factorization base approach, and Heat Wavelength Diffusion Pattern approach. In particular, we attempt to proffer answers to the following two questions: (1) To what extent does each of the four approaches of network embedding models accurately cluster similar nodes with the minimal dissimilarity index? (2) To what extent does each of the ten representation learning models accurately predict missing nodes when compared to standard heuristic measure scores?

4.1 Datasets

For our experimental evaluation, we used four data sets: Facebook, Protein-Protein Interactions, EU Email, and WikiVote.
Facebook: Facebook is a social relationship network; Nodes represent users who are registered on the network, and edges represent a friendship relation between any two users. The Facebook network used for the experiment has 8763 nodes and 79864 edges.
Protein-Protein Interaction (PPI): PPIs network is made up of organized functional protein units, designed to carry out biological and molecular processes in organisms. A rattus PPI network is used in this experiment, with nodes corresponding to proteins and edges representing the PPIs. The network is made up of 8912 nodes and 159728 edges.
EU Email: The EU email network depicts a communication flow between individuals in a project team. Nodes represent the people sending the mails, while the edges in the network indicate relations between named entities and represent a co-occurrence in the same email part, paragraph, sentence or a composite named entity. The network comprises 1005 nodes and 51142 edges.
WikiVote: The Wikipedia vote network contains all the Wikipedia voting data over a period of years. Nodes in the network represents Wikipedia users and an edge between two nodes denotes that a user voted on another user. The network is made up of 7115 nodes and 103689 edges [17].

4.2 Experimental Settings

In this study, we have chosen two common unsupervised machine learning tasks to evaluate the quality of the representation learning embeddings: Network Clustering and Link Prediction. In order to measure the network clustering results derived, we compare these with standard performance metric (dissimilarity proximity measure). To compute the probability distribution and reduce the embedding into two-dimension, we achieve this using T-distributed Stochastic Neighbor Embedding (t-SNE). We pass the two-dimensional probability distribution into a modified K-means partitioned heuristic framework using sklearn. Since there is the possibility of K-means converging to local

minima from different initializations, the algorithm was run a number of times for accuracy. The algorithm is relatively efficient with an *O(kdi)* time complexity; where *k* is the number of clusters, *d* is the number of data objects, and *i* the number of algorithm iterations. In addition, we have chosen the silhouette method to validate the value of *k* for each of the dataset used in the experiment. Monitoring the distribution of data points across groups provides insight into how the algorithm is splitting the data for each *k* [4].

The second evaluation method involves conducting a link prediction task using the embeddings derived from the models. The idea is to measure their ability in accurately predicting similar missing nodes or even future nodes. To achieve this, we split the learned representations embeddings of all nodes via a 10-fold cross validation, using 60% of the nodes to train the prediction model running on tensorflow backend engine and the remaining 40% was reserved for testing the trained model. The process was repeated 5 times, with the mean accuracy performance reported. Since we were unable to get existing studies to compare with our result with the exception of a study conducted by [3], we evaluated our result using standard heuristic scores in Table 1.

There is still an open research problem when it comes to determining the optimal embedding dimensions for learning embeddings for nodes in a network; hence we chose dimension of 64 across all 10 models. We also assigned a weight of 1 for all edges in the dataset, and the relationship between nodes are all undirected. In computing the probability distribution for each embedding, we used a perplexity value of 30 with a learning rate of 200.0; this rate ensures an ideal equidistant between a node and the nearest neighbor. If the learning rate is too low, most nodes may get compressed in a dense cloud.

5 Experimental Analysis and Results

We will start with the community clustering task for two datasets across all 10 models, for which we evaluated the results of the clustering task using the dissimilarity proximity measure.

1. Text-Associated DeepWalk -TADW [8] – Incorporates text features of vertices into network representation learning under the framework of matrix factorization.
2. DEEPWALK [5] – uses local information obtained from truncated random walks to learn latent representations. It follows a Random Sampling approach.
3. WALKLETS [1] – learns multiscale relationships by subsampling short random walks on the vertices of a graph. It follows a Random Sampling approach.
4. DANE [4] – provides an offline method for embedding and then leverages matrix perturbation to maintain an end embedding results in an online manner. It follows a Deep Learning approach.
5. GRAPHWAVE [10] – learns each node's network neighborhood via a low-dimensional embedding by leveraging heat wavelet diffusion patterns.
6. FSCNMF [6] – A matrix factorization framework which uses the network structure and the content of the nodes while learning a low dimensional representation of each node in the network.

7. GRAREP [9] – integrates global structural information in learning low dimensional vectors for vertices in the network. It follows a Random Sampling approach.
8. Binarized Attributed Network Embedding – BANE [11] - learns binary node representation by adopting Weisfeiler-Lehman proximity matrix to capture data dependence between node links and attributes.
9. NODE2VEC [3] - learns a mapping of nodes to a low-dimensional space that maximizes the likelihood of preserving network neighborhoods of nodes. It follows a Random Sampling approach.
10. HIVEC [2] – learns latent vector representations of graphs in a manner that captures the semantic dependencies of sub-structures. It follows a Deep Learning approach.

5.1 Community Clustering Task

We start off by conducting a silhouette metric analysis on the dataset in other to determine the accurate value of *k* clusters for use in our algorithm. The silhouette table and plot display a measure of how close each point in one cluster is to points in the neighboring clusters and thus provides a way to assess certain parameters like the optimal number of clusters. This measure has a range of [-1, 1]. The Cluster with the highest silhouette score is deemed valid and optimal for use.

The Silhouette Score analysis shows $k = 7$ as ideal number of clusters polling the highest probability of 47.44%. The 10 models used in this study are summarized in Fig 1 and Table 2.

Perhaps the first way to validate the output of a cluster is to visually examine the results, especially looking out for undesirable outliers in a technique known as compactness or cluster cohesion. Compactness or Cluster Cohesion measures how close the objects within the same clusters are. A look at the result shows DANE (Fig. 2d), BANE (Fig. 2h), and DEEPWALK (Fig. 2b) having good measure of compactness for cluster objects. WALKLETS (Fig. 2c) and NODE2VEC (Fig. 2i) shows fair cluster cohesion of objects in each cluster. In addition to the silhouette coefficient which we have used to measure how well the observations are clustered (basically measuring the ideal number of clusters), we have also adopted the dissimilarity measure (also known as distance measure) $d(a_i, a_j)$ where the distance measure decreases if a_i, a_j are similar. Since the 10 learning representation models have been configured to learn embeddings for similar nodes, each cluster will be deemed a good fit when the mean distance proximity measure computed from all centroid in the clusters is as minimal as possible.

Table 3 evaluates the result of the community clustering task shown in Figure 2. For both Facebook and WikiVote datasets, DANE model showed better clustering result in that it gives the most minimal dissimilarity index of all 10 models. This is aptly followed by the BANE model and DEEPWALK model respectively. On a second look at the result of Figure 2, the evaluation result matches closely with the cluster cohesion described as DANE, BANE, and DEEPWALK show better cluster cohesion. From this result, we can conclude that for the 10 models analyzed in this study, DANE model outperforms the other 9 models with better cluster cohesion.

5.2 Link Prediction

Of all the existing works that was reviewed for our study, only NODE2VEC [3] was found to have carried out link prediction task as part of its study. Still, an extensive comparison was not done with other models; as NODE2VEC [3] was only compared with DEEPWALK [5], LINE [18], and Spectral Clustering [19]. As a result, we decided to evaluate our link prediction results against standard heuristics measures that have achieved good performance in link prediction. For the link prediction task, we consider a network, with a certain fraction of the network edges removed; the aim is to be able to predict the missing network edges. We generated the dataset of edges by randomly removing 40% of the edges from the network while also ensuring that the network obtained after removing the edges still remains connected. We test our benchmark on three of the four datasets already highlighted in Section 4; Protein-Protein Interaction Rattus Network, Facebook network, and the EU Email network as seen in Table 4.

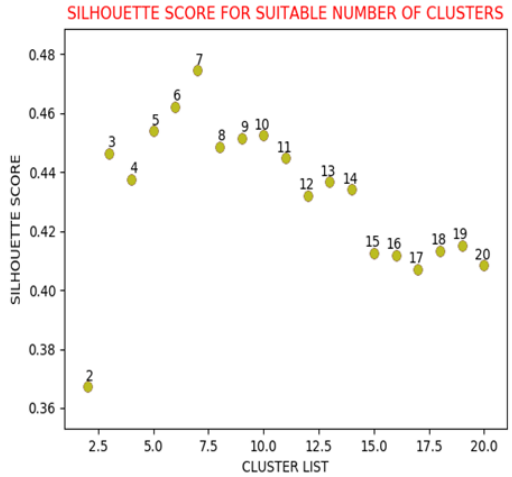

Fig 1: Silhouette Score plot for desired number of clusters

Table 2: Silhouette Score against cluster sizes

Clusters	Silhouette Score
2	0.367224
3	0.44635
4	0.437617
5	0.453904
6	0.46229
7	**0.474415**
8	0.448693
9	0.451635
10	0.452439
11	0.444739
12	0.432087
13	0.436623
14	0.434014
15	0.412669
16	0.411903
17	0.406925
18	0.413304
19	0.415156
20	0.40832

Table 3: Mean Dissimilarity Index for Facebook Data sets

Model Type	Dissimilarity Matrix	Index
TADW	[24.09258293216734, 23.272426609849978, 22.31372088469526, 23.835034086462842, 22.701292267860444, 24.049022357489 36, 23.701118051526446]	23.2236
DEEPWALK	[17.899597185693455, 17.925333713220148, 18.308366708525906, 15.65619725428297, 16.219751422606866, 15.053926383492577, 16.30509707107752]	16.5326
NODE2VEC	[17.68396846159482, 15.545018519921527, 19.62222865710842, 17.91932622331429, 15.516374436830244, 16.705728828149574, 17.293179700947903]	17.0551
WALKLETS	[20.393350954314233, 18.825351545906994, 19.514357330738807, 17.926636539922047,	19.3215

	20.073275841539694, 20.416489680391575]	19.600705637830497,	
DANE	[12.952821860718945, 12.661904065411399, 13.128929261952898, 12.089262792319495]	13.249624085929065, 13.145025949122703, 11.128012811988654,	12.6222
GRAPHWAVE	[20.1936862996862, 23.883609763022765, 24.999890681405944, 21.311120070524776]	23.79586678889978, 20.25024004683036, 20.645759677050954,	22.1543
FSCNMF	[21.219608344974475, 25.75631172425781, 24.073273591895568, 22.181106566044992]	23.429491557707227, 20.592362194142257, 20.981620135348678,	22.6048
GRAREP	[21.606417191726546, 24.260629216423315, 22.385479061600286, 23.33660811757346]	21.991344043124702, 20.812562519189584, 20.976267249573393,	22.1956
BANE	[15.7316162439905, 18.422650905307407, 16.682025870711747, 14.829922419467298]	14.236503946269604, 15.877252420022643, 16.505557474252285,	15.0408
HIVEC	[19.228950356899386, 21.13244445238893, 20.315110137060913, 22.221200209028606]	21.12903632689628, 19.558942071011625, 20.9136822397307,	20.6428

Table 4: Area Under Curve (AUC) scores for link prediction compared with results of popular heuristics baselines

	Algorithm	Datasets		
		PPI	Facebook	EU Email
	Adamic Adar Score	0.7178	0.7393	0.7198
	Preferential Attachment	0.6746	0.6943	0.6702
	Jaccard Coefficient	0.7299	0.7115	0.7145
	Resource Allocation Index	0.7330	0.7220	0.7376
1	TADW	0.6419	0.6135	0.7303
2	BANE	0.5817	0.5954	0.6403
3	DEEPWALK	0.7685	0.7489	0.7816
4	GRAPHWAVE	0.6833	0.6914	0.6701
5	GRAREP	0.7296	0.6875	0.7058
6	FSCNMF	0.6674	0.6940	0.6727
7	NODE2VEC	0.7811	**0.7948**	0.7599
8	HIVEC	0.7013	0.7620	0.7259
9	WALKLETS	0.6840	0.6612	0.6449
10	DANE	**0.7964**	0.7801	**0.7910**

We summarize the results for link prediction in Table 4. From a general observation of the result, we can infer that over the three datasets, some of the representation learning models for network nodes significantly outperform the heuristic bench mark score. On PPI Rattus and Email network, DANE outperforms other models, outperforming NODE2VEC by 1.9% on PPI network and DEEPWALK by 1.2% on the Email network respectively in the AUC scores. However, the NODE2VEC model outperforms other models on the Facebook network. Across the three datasets, DANE, NODE2VEC, DEEPWALK, HIVEC and GRAREP outperforms the heuristic benchmark scores with

DANE, NODE2VEC and DEEPWALK especially achieving better AUC improvement over the best performing baseline Resource Allocation Index and Adamic Adar Score.

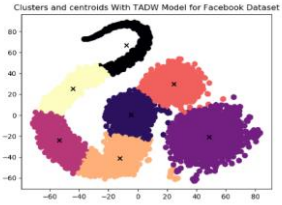

Figure 2a: TADW Clusters

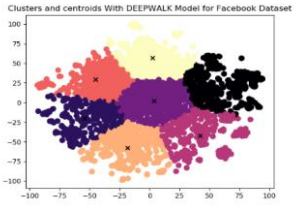

Figure 2b: DEEPWALK Clusters

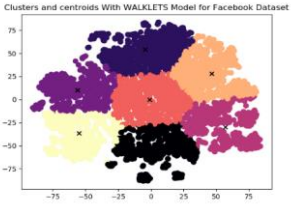

Figure 2c: WALKLETS Clusters

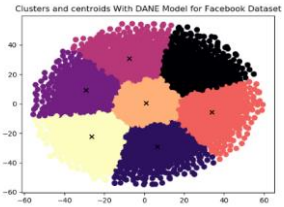

Figure 2d: DANE Clusters

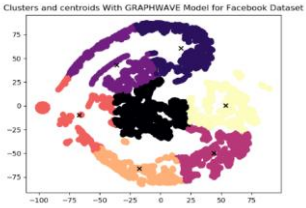

Figure 2e: GRAPHWAVE Clusters

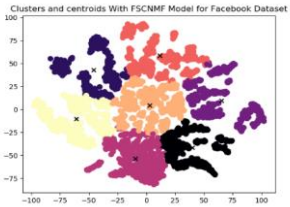

Figure 2f: FSCNMF Clusters

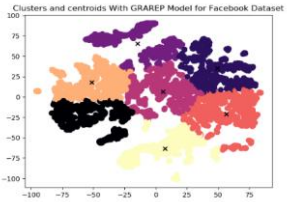

Figure 2g: GRAREP Clusters

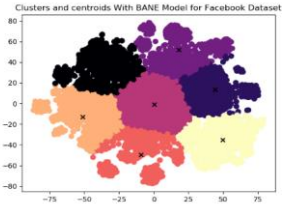

Figure 2h: BANE Clusters

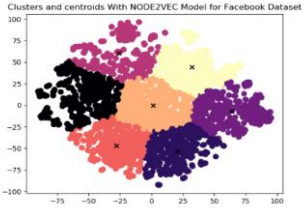

Figure 2i: NODE2VEC Clusters

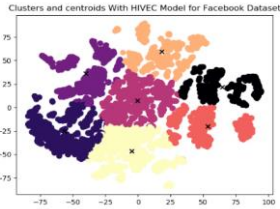

Figure 2j: HIVEC Clusters

Fig 2: Visual Output of Clustering task done on Facebook data set with $k = 7$

6. Discussion and Conclusion

In this paper, we reviewed 10 models for graph embedding and evaluated the models using Community Clustering and Link Prediction task. We compared the results with standard heuristic baselines. The Deep Learning Based Approach performs best with DANE and HIVEC scoring very strongly in the link prediction task; DANE outperforms other models on two of the three datasets. The Random Sampling Approach ranks

next with NODE2VEC and DEEPWALK performing second best behind the DANE model [4]. The Deep learning-based approach also outperforms in clustering task, with DANE scoring the lowest dissimilarity metric index.

References

[1] B. Perozzi, V. Kulkarni, H. Chen, S. Skiena, "Don't Walk, Skip!: Online Learning of Multi-scale Network Embeddings", In proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pages 258-265, (2017).

[2] A.S. Patel, R.M. Ramakrishna, M. Jai, K. Singh, N. Sivadasan, and V.N. Balasubramanian. "HIVEC: A Hierarchical Approach for Vector Representation Learning of Graphs.", (2018).

[3] A. Grover and J. Leskovec, "node2vec: Scalable Feature Learning for Networks", In proceedings of the 22nd ACM SIGKDD, pages 855-864, (2016).

[4] L. Jundong, H. Dani, X. Hu, J. Tang, Y. Chang, and H. Liu. "Attributed Network Embedding for Learning in a Dynamic Environment", *ArXiv* abs/1706.01860, pages 1-10, (2017).

[5] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online Learning of Social Representations", In proceedings of the 20th ACM SIGKDD, pages 701-710, (2014).

[6] S. Bandyopadhyay, H. Kara, A. Kannan, and M.N. Murty, "FSCNMF: Fusing Structure and Content via Non-negative Matrix Factorization for Embedding Information Networks", Conference Submission, arXiv:1804.05313 [cs.SI], (2018).

[7] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality", In Proc. of NIPS, pages 3111–3119, (2013).

[8] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Y. Chang, "Network Representation Learning with Rich Text Information", Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI), (2015).

[9] S. Cao, W. Lu, amd Q. Xu, "GraRep: Learning Graph Representations with Global Structural Information", In proceedings of the 24th ACM International Conference on Information and Knowledge Management, pages 891-900, (2015).

[10] C. Donnat, M. Zitnik, D. Hallac, J. Leskovec, "Learning Structural Node Embeddings via Diffusion Wavelets", In proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 1320-1329, (2018).

[11] H. Yang, S. Pan, P. Zhang, L. Chen, D. Lian, and C. Zhang, "Binarized Attributed Network Embedding", In proceedings of the Twelfth ACM International Conference on Web Search and Data Mining (WSDM), pages 393-401, (2019).

[12] G. Punj, and D.W. Stewart. "Cluster Analysis in Marketing Research: Review and Suggestions for Application", Journal of Marketing Research, 20(2), pages 134-148, (1983).

[13] S. Setyaningsih, "Using Cluster Analysis Study to Examine the Successful Performance Entrepreneur in Indonesia", Procedia Economics and Finance, Vol 4, pages 286 – 298, (2012).

[14] S. Ullman, T. Poggio, D. Harari, D. Zysman, and D. Seibert, "Unsupervised Learning Clustering", Centre for Brains, Minds, and Machines, pages 1-54, (2014).

[15] B. Rozemberczki, R. Davies, R. Sarkar, and C. Sutton, "GEMSEC: Graph Embedding with Self Clustering", Journal of Social and International Networks, ASONAM, (2019).

[16] R.N. Lichtenwalter, J.T. Lussier, and N.V. Chawla, "New Perspectives and Methods in Link Prediction", In proceedings of the 16th ACM SIGKDD, pages 243-252, (2010).

[17] J. Leskovec and A. Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

[18] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei., "LINE: Large-scale Information Network Embedding". In WWW, (2015).

[19] L. Tang and H. Liu., "Leveraging social media networks for classification". Data Mining and Knowledge Discovery, Volume 23, Issue 3, pages 447–478, (2011).