Deep Learning Roles Based Approach to Link Prediction in Networks

Aman Gupta* and Yadul Raghav*

Department of Computer Science and Engineering
Indian Institute of Technology (BHU)
Varanasi, India 221–005

Abstract. The problem of predicting links has gained much attention in recent years due to its vast application in various domains such as sociology, network analysis, information science, etc. Many methods have been proposed for link prediction such as RA, AA, CCLP, etc. These methods required hand-crafted structural features to calculate the similarity scores between a pair of nodes in a network. Some methods use local structural information while others use global information of a graph. These methods do not tell which properties are better than others. With an in-depth analysis of these methods, we understand that one way to overcome this problem is to consider network structure and node attribute information to capture the discriminative features for link prediction tasks. We proposed a deep learning Autoencoder based Link Prediction (ALP) architecture for the latent representation of a graph, unified with non-negative matrix factorization to automatically determine the underlying roles in a network, after that assigning a mixed-membership of these roles to each node in the network. The idea is to transfer these roles as a feature vector for the link prediction task in the network. Further, cosine similarity is applied after getting the required features to compute the pairwise similarity score between the nodes. We present the performance of the algorithm on the real-world datasets, where it gives the competitive result compared to other algorithms.

Keywords: Link Prediction, Deep Learning, Autoencoder, Latent Representation, Non-Negative Matrix Factorization.

1 Introduction

With the surge of the Internet, everyone is almost interconnected via social media platforms (e.g., Facebook, Twitter, Instagram, etc.), professional blogs, and websites. Networks are tools to represent these interconnections in their respective scenarios. For example, an individual profile on Facebook is represented by a node in the network, and relationships between two profiles are represented by the links (edges) between two nodes. Thus, a network can be used to model the communication of a system. Everyday, relationships are changing among individuals, i.e., some new links are formed, and some of them are vanished due to several reasons. This

* These authors have contributed equally.

David C. Wyld et al. (Eds): NLP, JSE, MLTEC, DMS, NeTIOT, ITCS, SIP, CST, ARIA - 2020
pp. 203-222, 2020. CS & IT - CSCP 2020 DOI: 10.5121/csit.2020.101416

behavior makes the scenarios quite complex and dynamic, and dealing with them becomes more challenging. The above scenarios can be modeled using a social or a complex network. Lots of issues exist when dealing with these networks. "Finding missing links or future links in an observed network" is one of the interesting problems which is known as link prediction (LP). Nowell and Kleinberg [1] formally defined the link prediction as follows. Suppose a graph $G_{t_0-t_1}(V, E)$ represents a snapshot of a network during time interval $[t_0, t_1]$ and $E_{t_0-t_1}$, a set of edges present in that snapshot. The task of link prediction is to find set of edges $E_{t_0^{'}-t_1^{'}}$ during the time interval $[t_0^{'}, t_1^{'}]$ where $[t_0, t_1] \leq [t_0^{'}, t_1^{'}]$. Link prediction has been applied in several domain of applications like friendship prediction in Facebook, recommendation system in e-commerce [2], protein-protein interactions (PPI) in bioinformatics [3] etc.

Several authors have presented seminal works on classical methods (indices) of link prediction in networks that are broadly classified into structural similarity-based and maximum likelihood-based methods, which are presented in the reviews [1, 4]. Structural similarity is computed based on the properties of the structure of the networks. These properties are easy to compute, and no need to extract extra information like attribute and other side information. Works on these structural properties are grouped into three categories viz., local, global, and quasi-local. Most classical similarity indexes are based on these three categories. Local indices extract local information (like degree, common neighbors, clustering coefficients, etc.) to compute the similarity between two nodes. Common neighbor [1], Adamic-Adar [5], Resource allocation [6], Preferential attachment [7],etc. are the most well known local indices which are heuristics and used in both supervised and unsupervised settings to show the relevance of other methods. Global indices focus on extracting global properties or information where the whole network is taken into account. These indices are more complex and time-consuming that limits its application for large networks. Most path based indices e.g., Katz [8], shortest path [1], average commute time [9], PageRank [10], Leicht-Holme-Newman Index [11], Random walk with restart [12] etc., comes under global indices. Local similarity indices are simple and efficient in computation, whereas the global are complex and computationally inefficient. A trade-off between them is quasi-local indices, which are as efficient as local indices and not limited to neighborhood information. Sometimes, these methods (indices) take the whole network in computation [13]. Local path index (LP) [14], Local random walk (LRW) [15], Superposed random walk (SRW) [9] etc., are such indices.

Several machine learning [15, 16], and deep learning frameworks [17, 18] also have been explored for the link prediction task. Both supervised and unsupervised models have been used to find missing links in the literature. An unsupervised deep

learning model aims at finding hidden structures or patterns in the data and learns suitable representation that is useful input for several tasks. One of the critical issues with these models is the data representation that extracts useful information from the data. Unsupervised models detect and eliminate irrelevant variability present in the input data. Simultaneously, it preserves the information that is useful to several tasks like detection, prediction, visualization, etc. Some of these models are based on reconstructing the input from the suitable representation (or code) with some desired properties like sparsity, low dimensionality, etc. The autoencoder is one of the unsupervised deep learning models using which we predict missing links in the paper. An autoencoder consists of two parts encoder and decoder. The encoder maps the input data to latent representation or code or feature vector, and this code can further be used in downside the prediction task. Autoencoder can be expressed in another way as consisting of one input layer, one or more hidden layers, and an output layer, as shown in Figure 1. A deep autoencoder consists of more hidden layers. Reasons for the use of the deep autoencoder are two folds first, once the training is complete, computing code (latent representation) takes less time; the relevancy of the extracted information can be checked through the decoder by reconstructing the input. In contrast, link prediction task in social networks can also be treated as grouping the nodes based on the similar structural properties and behaviour. Given a network, we have defined the node-roles relationship between them, with the intuition that two nodes belong to the same role, or we could say they are well connected to each other if they have similar structural behavior or function. Node-Roles relationships help in understanding the underlying behavior in a network and also exploring the interesting data analysis tasks such as sense-making, node-similarity, and prediction tasks [63].

In this paper, we transfer the effects of roles in a network to link prediction tasks, where node-roles relationship act as additional features to find the similarity between the nodes in a network. Without any other information except the network structure, the key problem with the link prediction task is identifying and deciding what structural features are needed that can be derived from the network, which will lead to predicting links in the network. Once we get the desired features, the link prediction task can be well-formulated as finding similarity between the nodes based on these extracted features. Given a dataset, we define the link prediction problem as

- finding the features matrix and node-roles relationship (where the features matrix is obtained by using the latent representation of autoencoder architecture, and the node-roles matrix is determined by applying the non-negative matrix factorization on these extracted feature matrix).
- determining the similarity score between the nodes using both feature matrix and node-roles matrix by applying the Cosine similarity function.

We summarize the main contributions of this paper are determining a set of structural features using the autoencoder architecture and transferring the effect of node-roles relationship to perform the link prediction task.

Organization. Section 2 talks on some literature work on link prediction. The proposed work is presented in section 3. Section 4 discusses an experimental study consisting of an evaluation strategy and the results of several methods against real network datasets followed by a statistical test. Finally, section 5 concludes our work.

2 Related Work

Newman presented a paper on link prediction on collaboration networks in Physics and Biology [19]. In such networks, two authors are considered to be connected if they have at least one paper co-authored by them simultaneously. In the empirical study, the author demonstrated that the likelihood of a pair of researchers teaming up increments with the numbers of different colleagues they have in mutual relation, and the likelihood of a specific researcher acquiring new partners increments with the number of his past teammates. The outcomes give experimental proof in favor of formerly guessed mechanisms for clustering and power-law degree distributions in networks. Later, Nowell and Kleinberg [1] proposed a link prediction model explicitly for a social network. Each node in the network corresponds to a person or an entity, and a link between two nodes shows the interaction between them. The learning paradigm in this environment can be used to extract the similarities between two nodes by several similarity metrics. Ranks are assigned to each pair of nodes based on these similarities, then higher ranked node pairs are designated as predicted links. Further, Hasan et al. [15] expanded this work and demonstrated that there is a significant increase in prediction results when additional topological information about the network is available. They considered different similarity measures as features and performed a binary classification task using a supervised learning approach, which is similar to link prediction in their framework.

The graph embedding is considered as a dimensionality reduction technique in which higher D dimensional nodes in the graphs are mapped to a lower d ($d \ll D$) dimensional representation space by preserving the graph properties as much as possible [20]. These graph properties can be node pair similarity, node neighborhood similarity, substructure similarity, etc. Recently, some graph embedding techniques [21]- [25] have been proposed and applied successfully in link prediction and node classification problems. Deep learning models of graph embedding have also recently been introduced that can be classified in with and without random walk strategies [20]. In the first case, the graph is represented as paths sampled from it, which are inputs to an embedding model and the whole graph as input for a later case. The deep learning model is then applied to these sampled paths in the

framework and encodes to preserve the graph properties (i.e., path properties here). Lots of seminal works based on the first category (i.e., with random walk strategy) are available in the literature like DeepWalk [24], Node2vec [22], HARP [26]. These models are mainly shallow in nature, moreover deep model are based on without random walk strategies like SDNE [27], DNGR [28], VAGE [29], SEAL [30].

Machine learning and deep learning for link prediction. In the literature, most of the well-known link prediction approaches focus on heuristics, which are domain-specific and ignore the evolutionary behavior of the networks. They mainly work on static networks. From the last decade, several machine learning approaches have been applied to improve link prediction performance. In such approaches, the challenging task is to represent features in a format suitable for the application, which vastly affects the performance results. M. A. Hasan et al. [15] proposed a seminal work on link prediction using supervised learning in which three types of features of graph viz., proximity features, aggregated features, and topological features are employed with several classifiers. Likewise, Doppa et al. [16] put forward works based on a supervised approach on link prediction where k-means classifier employed on feature vectors. Recently, deep learning, a new direction in machine learning have been proposed in the literature. The seminal works based on deep models, for examples, stacked denoising autoencoders (SDAE) [17] and convolutional neural networks (CNN) [18] have shown their great potential of representing and learning features in computer vision and natural language processing. One problem of conventional deep learning models is the independent and identically distribution of the input, which cannot model relational data. To overcome this issue, H. Wang et al. [31] employed a Bayesian deep learning framework that learns relational data (network data) effectively. They jointly model high-dimensional node attributes and link structures in their framework and product of Gaussian as an inference approach. Xiaoyi Li et al. [32] introduced a novel deep learning framework, namely Conditional Temporal Restricted Boltzmann Machine (ctRBM), that captures the evolutionary patterns of networks (i.e., dynamic networks). Their framework is based on the joint inferential effects of seed nodes and their local neighbor's influences. [33] proposes a supervised framework of deep learning where two different architectures show their competitive results with the state-of-the-art.

Graph convolutional neural networks (GCNs) [34] are the recent class of deep network approaches used in network embedding, node classification, and link prediction. The model learns representation from a localized first-order approximation of spectral convolutions. Thomas N. Kipf et al. [29] introduced a framework based on GCN that uses a simple inner product decoder and learns node features of structured graph data. In this paper [65], the author considers the link prediction task as a collaborative filtering problem, where they treated the nodes as items and edges

like the rating in the recommendation system and proposed a non-negative matrix factorization approach combined with a bagging technique to predict which nodes are expected to connect. Similarly [64], a unified framework has been proposed for link prediction tasks based on non-negative matrix factorization with coupling multivariate information where they have used both the internal latent feature information and external node attribute information of the network. Different approaches have been used for role extraction in the network. In [63], RolX an unsupervised learning approach has been proposed where it automatically determines the underlying structural roles in a network. It also assigns a mixed-membership of these extracted roles to each node in the network. The author has analyzed different methodologies, research issues, and characteristics that should be considered during the role analysis.

3 Proposed Work

3.1 Network Architecture

The proposed architecture is an unsupervised framework of deep learning model which maps the adjacency matrix of the given graph into the node-features matrix. The architecture does not need any hand-crafted (manual selection) features; rather, it extracts important features automatically. The overall architecture has been divided into two stages. The first stage consists of the autoencoder neural network, which is an unsupervised framework of deep learning that uses backpropagation to update synaptic weights. It mainly consists of two parts encoder and decoder. The encoding layer compresses its input to a lower-dimensional code, known as latent representation. The objective of the decoding layer is to reconstruct the input using this compressed code. Clearly, an autoencoder can be considered as a dimensional reduction technique. The encoder maps the input data to latent representation or feature vector, and this vector can further be used in downside the prediction task. Through this model, we get only the important features that are needed to represent the graph by filtering out the unnecessary details from the graph. We have used this feature vector as a subset of our final node-feature matrix. The second stage uses the given latent representation from the neural network to find out the roles in the graph. Using these roles, we have assigned a mixed-membership of roles to each node in the graph network, giving a node-role matrix, which acts as an additional feature set for our node-feature matrix. In summary, to get the final Node-feature matrix, we have concatenated the feature vector, which we get after the encoder layer from the autoencoder neural network with the node-role matrix. In the proposed architecture, the first hidden layer consists of 16 neurons and the second hidden layer (or latent representation) contains 8 neurons (called latent variables). The learning rate is set to a low value of 0.01 in descent gradient optimization

during the learning process. The workflow of our proposed architecture has been shown in Figure 1.

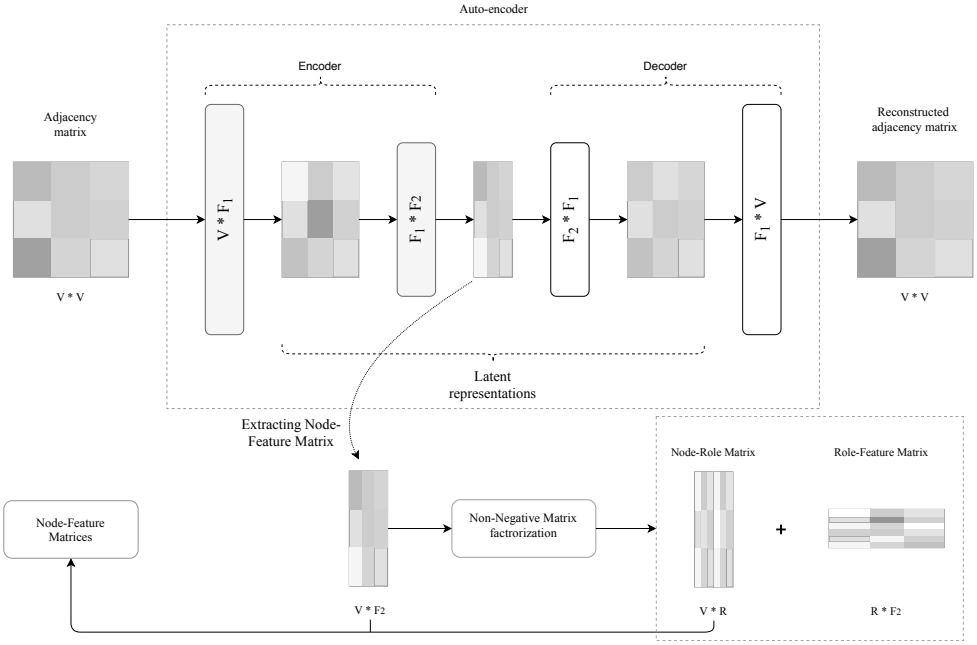


Fig. 1: The Deep Autoencoder Framework.

3.2 Problem characterization for deep autoencoder framework.

Considering a simple undirected network (also applicable to directed and weighted networks), $G(V, E)$, where V is the set of vertices (or nodes) and E is the set of edges. The given graph network can be represented as an adjacency matrix $A \in \mathbb{R}^{|V|\times|V|}$. Inputs to the neural network is a matrix, which is the adjacency matrix A. Aim of the deep autoencoder is to learn low dimensional latent representation $Z \in \mathbb{R}^{|V|\times F_2}$ for the nodes with the constraint of minimization of the reconstruction error (loss). Other half of the architecture aims to find the node-role matrix of dimension $R \in \mathbb{R}^{|V|\times r}$ with the help of feature vector Z. The main focus of the overall architecture is to find the node-feature matrix of dimension $F \in \mathbb{R}^{|V|\times F}$.

3.3 Preprocessing.

Input to the proposed model is a normalized adjacency matrix (A_{Norm}), which is the output of the preprocessing step. Normally, neural architectures use the original

adjacency matrix in a layer-wise propagation function that causes a change in the scale of feature vectors. That is, larger degree nodes have more contribution (i.e., feature value), and smaller degree nodes have lower feature values in the feature representation. The different scales of input feature values are problematic in training those networks that use stochastic gradient descent algorithms. To mitigate this problem, the original adjacency matrix is normalized by taking the average of corresponding neighboring nodes features as described in the paper [34]. The symmetric version of this normalization is expressed as follows

$$A_{Norm} = \tilde{D}^{\frac{-1}{2}} \tilde{A} \tilde{D}^{\frac{-1}{2}} \tag{1}$$

where $\tilde{A}$ (i.e. $A + I$,that enforces to include own features also) is the adjacency matrix of the network and $\tilde{D}$ is the node degree matrix of $\tilde{A}$.

3.4 Roles Extraction.

After getting a latent representation of the graph from the neural network, we have a feature matrix $Z \in \mathbb{R}^{|V| \times F_2}$, the next step of our algorithm is to generate a rank r approximation $PQ \approx Z$, where $P \in \mathbb{R}^{|V| \times r}$ represents the node-roles relationship and $Q \in \mathbb{R}^{r \times F_2}$ define how each identified roles contributes to estimated feature values. To do this, we have used Non-negative Matrix Factorization as it aims to find two non-negative factor matrices which simplify the interpretation of the node-roles relationship. Since we have not defined the number of roles required, we decided to use the Minimum Description Length criterion [37], to find the optimal number on roles r as described in the paper that results in the best compression. Mathematically, it can be written as,

$$minimize \|Z - PQ\|_F^2 \, w.r.t. \, P, Q \, s.t. \, P, Q \geq 0 \tag{2}$$

In the last step, we have added both the feature matrix Z and node-roles matrix P such that $F = Z + P$, where $F \in \mathbb{R}^{|V| \times (F_2 + r)}$ is overall features set that is derived from the dataset. Finally, for each node pair, cosine similarity [35] index is used to find similarity between them. Once the score of non-observed links is available in the sorted order, we can compute the Area under the Operating receiver characteristics (AUROC) and average precision to evaluate the accuracy of our approach.

4 Experimental Study

4.1 Evaluation metrics

Consider a simple undirected graph/network $G(V, E)$ where V characterizes a vertex-set and E, the edge-set. Although a simple graph is considered so, parallel edges and self-loops are not permitted. In a simple graph, a universal set U

contains a total $\frac{|V|(|V|-1)}{2}$ edges, where $|V|$ represents the size of the vertex-set in the graph. $(U-E)$ number of edges is termed as the set of nonexistent links, some of which may be missing that may appear in the future. Finding out such missing links is the aim of link prediction [14]. The accuracy of an algorithm can be tested by partitioning the set of observed links E into two sets. E^T, a training set about which we know at all, and a test set (or validation set), E^P in which there are edges which is not present in the training set. Therefore, $E^T \cup E^P = E$ and $E^T \cap E^P = \phi$ with this strategy, it may be possible that some edges may not ever be chosen in the test set or others could be repeated, which results in statistical bias. This problem is overcome by a procedure of sampling known as K-fold cross-validation. To calculate the accuracy of algorithm, generally, two metrics are used: AUROC [40], and precision [41], [42]. Based on the above definitions, the following observations can be made in a graph:
Total possible links in the graph $= U$,
Existent links $= E$,
Non-existent links $= U - E$,
Observed links $= E^T =$ Training set,
Non-Observed links $= U - E^T$,
Missing links $= E^P =$ Test set.

Area under the Operating receiver characteristics (AUROC) Given a ranking of total non-observed links, the term AUROC is estimated as the likelihood that a chosen missing link is given a higher score than a randomly chosen non-existent link. Each time two edges are selected randomly one from each set and compared their scores. Then, AUROC can be calculated using the following expression:

$$AUROC = \frac{n_1 + 0.5 \times n_2}{n} \tag{3}$$

where, n is total independent comparisons, n_1 is number of times the missing link with a higher score,n_2 is number of times they have same score. The standard value of AUROC should be 0.5 which will be possible under an independent and identical distribution. A score greater than 0.5 represents improved accuracy.

Precision Given the ranking of non-observed links, precision can be characterized as the proportion of relevant items to the number of items chosen i.e.,

$$Precision = \frac{L_r}{L} \tag{4}$$

where, L represents predicted links having top scores, and L_r, the number of predicted links which are correct.

4.2 Datasets

The performance of the proposed method has been evaluated on twelve real-world network datasets collected from diverse areas.

- Karate[1] [43]: A friendship network of 34 members of karate club at a US university.
- Dolphins[1] [44]: A social network of dolphins living in Doubtful Sound in New Zealand.
- Lesmiserables[1] [45]: Co appearance network of characters of the novel LesMiserables.
- Adjnoun[1] [46]: Adjacency networkof common adjectives and nouns in the novel David Copperfield by Charles Dickens.
- Football1 [47]: American football games network played between Division IA colleges during regular season Fall 2000.
- Celegansneural[1] [49]: A neural network of C. elegans compiled by D. Watts and S. Strogatz in which each node refers a neuron and, an edges joining two neurons either by a synapse or a gap junction.
- Netscience[1] [46] is a co-authorship network of scientists working on network theory and experiment compiled by Newman in 2006.
- Political bolgs[1] [5] is a directed network of hyperlinks in political blogs of US election 2004.
- Jazz[2] [48] is the collaboration network of jazz musicians.
- Usair97[3] is an airline network of US where a node represents an airport and an edge shows the connectivity between two airports.
- Facebook[4] [50] is social network of user profiles and network data extracted from 10 ego-networks.
- Ca-GrQc[4] is collaboration network from the e-print arXiv of General Relativity and Quantum Cosmology.

Table 1 shows some basic topological properties of the considered networks datasets.$|V|$ and $|E|$ are the total numbers of nodes and edges of the networks respectively.$\langle D \rangle$ represents node pairs average shortest distance, $\langle K \rangle$ the average degree and $\langle C \rangle$ the average clustering coefficient of the network. r and H are the coefficient of assortativity and degree of heterogeneity respectively.

4.3 Baseline methods

- Common Neighbor(CN). In a given network or graph, the size of common neighbors for a given pair of nodes x and y is calculated as the size of intersection of

[1] http://www-personal.umich.edu/ mejn/netdata/
[2] http://konect.uni-koblenz.de/networks/
[3] http://vlado.fmf.uni-lj.si/pub/networks/data/
[4] https://snap.stanford.edu/data/

Table 1: Topological informations of real-world network datasets

Datasets	$\|V\|$	$\|E\|$	$\langle D\rangle$	$\langle K\rangle$	$\langle C\rangle$	r	H
Karate	34	78	2.337	4.588	0.570	-0.475	1.693
Dolphins	62	159	3.302	5.129	0.258	-0.043	1.326
Lesmiserables	77	254	2.606	6.597	0.573	-0.165	1.827
Adjnoun	112	425	2.512	7.589	0.172	-0.129	1.814
Football	115	613	2.486	10.661	0.403	0.162	1.006
Jazz	198	2742	2.235	27.697	0.620	0.020	1.395
Celegansneural	297	2148	2.447	14.456	0.308	-0.163	1.800
Usair97	332	2126	2.738	12.807	0.749	-0.207	3.463
Political blogs	1490	16718	2.738	22.440	0.361	-0.221	3.621
Netscience	1589	2742	5.823	3.451	0.878	0.461	2.010
Facebook	4039	88234	3.693	43.691	0.617	0.063	2.439
Ca-GrQc	5242	14496	6.049	5.531	0.687	0.659	3.051

the two nodes neighborhoods.

$$S(x,y) = |\Gamma(x) \cap \Gamma(y)| \tag{5}$$

where $\Gamma(x)$ and $\Gamma(y)$ are neighbors of the node x and y respectively.The likelihood of the existence of a link between x and y increases with the number of common neighbors between them. In a collaboration network, Newman calculated this quantity and demonstrated that the probability of collaboration between two nodes depends upon the common neighbors of the selected nodes. Kossinets [52] and Neal [53] investigated a large social network and recommended that two students are more likely to be friends who are having numerous common friends. It has been observed that the common neighbor approach performs well on most real-world networks and beats other complex methods.

– Jaccard Coefficient(JC). The Jaccard coefficient is defined as the probability of selection of common neighbors of pairwise vertices from all the neighbors of either vertex.

$$S(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|} \tag{6}$$

– Resource Allocation (RA). Consider two non-adjacent vertices x and y. Suppose node x sends some resources to y through the common nodes of both x and y then the similarity between the two vertices is computed in terms of resources sent from x to y. This is expressed mathematically as

$$S(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{k_z} \tag{7}$$

– Preferential Attachment(PA). The idea of preferential attachment is applied to generate a growing scale-free network. The term growing represents the incremental nature of nodes over time in the network. The likelihood incrementing

new connection associated with a node x is proportional to k_x, the degree of the node. Preferential attachment score between two nodes x and y can be computed as

$$S(x, y) = k_x * k_y \tag{8}$$

– Node Clustering Coefficient(CCLP). This index is also based on the clustering coefficient property of the network in which the clustering coefficients of all the common neighbors of a seed node pair are computed and summed to find the final similarity score of the pair. Mathematically, this index can be expressed as follows

$$S(x, y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} C(z) \tag{9}$$

where

$$C(z) = \frac{t(z)}{k_z(k_z - 1)} \tag{10}$$

and k_z is the degree of node z and $t(z)$ is the total triangles passing through the node z.

– CARIndex. CAR-based indices are presented based on the assumption that the link existence between two nodes increases if their common neighbors are members of local community (LCP theory) [56].

$$S(x, y) = CN(x, y) \times \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{|\gamma(z)|}{2} \tag{11}$$

where $CN(x, y)$ is common neighbour of (x, y) and $\gamma(z)$ is the subset of neighbors of node z that are also common neighbors of x and y.

– Katz Index. It directly aggregates over all the paths between x and y and dumps exponentially for longer paths to penalize them. It can be expressed mathematically as

$$S(x, y) = \sum_{l=1}^{\infty} \beta^l |paths_{x,y}^{<l>}| \tag{12}$$

where, $paths_{x,y}^{<l>}$ is considered as the set of total l length paths between x and y, β is a damping factor that controls the path weights.

– Node2vec(N2V). This is a node embedding technique where it learns a low dimensional continuous representation of nodes in a graph with the objective of preserving the neighborhood structure.

Table 2: Avg. Precision

	CN	JC	RA	PA	CCLP	CAR	Katz	N2V	ALP
Karate	0.148878	0.130261	0.069634	0.134948	0.162239	0.017582	0.260558	0.041504	**0.653197**
Dolphins	0.179974	0.205618	0.050494	0.082726	0.189028	0.007007	0.110008	0.024493	**0.610708**
Lesmiserables	0.249618	0.366998	0.249818	0.085681	0.256686	0.148050	0.085832	0.146678	**0.701639**
Adjnoun	0.079462	0.076121	0.018148	0.067761	0.084578	0.008218	0.105588	0.011761	**0.242644**
Football	0.459779	0.447029	0.111022	0.092703	0.468967	0.061658	0.178761	0.114445	**0.681193**
Jazz	0.326022	0.367671	0.319647	0.106431	0.328317	0.351645	0.263667	0.086058	**0.691399**
Celegansneural	0.118607	0.134162	0.036424	0.048702	0.146307	0.011771	0.051729	0.018931	**0.743431**
Usair97	0.126467	0.271941	0.231041	0.263769	0.141365	0.204706	0.050021	0.029793	**0.370255**
Political blogs	0.079613	**0.146614**	0.061253	0.015059	0.088426	0.063305	0.021715	0.008462	0.106545
Netscience	0.392733	0.433671	0.128983	0.002171	0.430501	0.108762	0.204778	0.081755	**0.587527**
Facebook	0.244687	0.172462	**0.440321**	0.019037	0.238218	0.235341	0.002152	0.125269	0.314704
Ca-GrQc	0.223600	0.063274	0.133345	0.019163	0.253608	0.217245	0.046797	0.048952	**0.381180**

Table 3: AUROC

	CN	JC	RA	PA	CCLP	CAR	Katz	N2V	ALP
Karate	0.693750	0.628375	0.757500	0.760375	0.656438	0.550500	0.611500	0.721125	**0.897395**
Dolphins	0.745418	0.769799	0.822843	0.726425	0.618911	0.357150	0.826348	0.750734	**0.882697**
Lesmiserables	0.889440	0.871515	**0.934390**	0.699676	0.888291	0.695770	0.912502	0.854812	0.906194
Adjnoun	0.665667	0.568315	0.647079	0.735380	0.653959	0.450198	0.658491	0.613725	**0.743431**
Football	0.873762	0.859288	0.854359	0.252409	0.813651	0.585301	0.854977	0.862271	**0.879024**
Jazz	0.948143	0.959044	**0.963302**	0.789540	0.955104	0.931445	0.452756	0.873276	0.909838
Celegansneural	0.815419	0.792798	0.848494	0.735148	**0.872517**	0.450223	0.416356	0.795693	0.720588
Usair97	**0.958332**	0.914826	0.946785	0.905626	0.957295	0.772429	0.50310	0.884882	0.838057
Political blogs	**0.941012**	0.907954	0.939749	0.934223	0.938549	0.739912	0.345143	0.866696	0.781857
Netscience	0.944599	0.953620	0.944769	0.639099	0.897433	0.532846	0.939401	0.892410	**0.966444**
Facebook	0.991824	0.989581	**0.995177**	0.832809	0.992504	0.944786	0.492362	0.991560	0.894836
Ca-GrQc	0.921563	**0.929400**	0.913091	0.741728	0.892601	0.605524	0.718201	0.908955	0.849303

4.4 Experiments Result Analysis

Accuracy Analysis: Table 2 shows the average precision results of the proposed method ALP with the baseline methods. Best accuracy values are shown in boldface against each network. We observe that the proposed method gives the best result on ten out of twelve network datasets, as shown in the table. Our method performs much better on all the datasets except Political blogs and Facebook dataset; RA is the best performer on Facebook, and JC is best on the Political blogs dataset. However, ALP shows the second-best result in both the datasets.

The AUROC results of the proposed and baseline methods are shown in Table 3. ALP performs best on karate, dolphins, Adjnoun, Football, and Netscience networks. RA best performs on Lesmiserables, Jazz, and Facebook networks, while CN is the best performer index on Usair97 and Political blogs networks. Jaccard (JC)

shows the best result on Ca-GrQc that are collaboration networks of scientists in computer science and general relativity. CCLP is best on Celegansneural network.

Robustness Analysis: Figure 2. shows the robustness measure of the existing and the proposed ALP method. The figure presents the effects of random noise (i.e., links are randomly added to the network) and random removal links. This concept is well explained in the Zhang, P. et al. [58] work on robustness under noisy environments. The parameter "$Ratio''$ on the X-axis defines the fraction of noisy links that are added or deleted to/from the training data as described in the above work. Positive values of this parameter represent a fraction of added links to the training data, and negative values represent a fraction of deleted links from the training data. Figure 2. shows the dependence of AUROC on these fraction of added and removal links.

ALP shows the best robustness with higher accuracy compared to the baseline methods against both added and deleted links on Karate and Dolphin networks [Fig.2a and 2c]. On Lesmiserables data, ALP shows better AUROC after the CN, JC, and CCLP; however, it shows the least fluctuation (highly robust) in the AUROC values [Fig. 2b]. It is the average performing method on Adjnoun, Jazz, Usair97, and Netscience datasets with AUROC values lower than CN, JC, and CCLP on Jazz Usair97 and Netscience [Figs. 2d, 2f and 2h], moreover it shows better robustness for both added and deleted links as shown in the figures. PA and Node2Vec are the best performing indices on Adjnoun and Netscience data, respectively. ALP shows the comparable result on the Football dataset [Fig. 2e]. One thing to note that the fluctuation of the AUROC values for random deleted links is greater than randomly added links, which is similar to the work [58]. In other words, random links deletion are more vulnerable to link prediction. Due to computational issues, robustness results of the remaining datasets are not shown.

Statistical Test: In this experiment, we conduct a statistical test [59] to show the significant difference between the proposed method (ALP) with the baseline methods. We perform the Friedman test [60], [61], to analyze whether there is a significant difference among multiple methods. It is a non-parametric counterpart of the repeated measures ANOVA. If the test result shows a significant difference, we further applied post hoc analysis to check the degree of rejection of each hypothesis. For the post hoc analysis, several methods are available in the literature, and we applied the post hoc counterpart of the Friedman test known as the Post hoc Friedman Conover method.

The Friedman test results for both average precision (Avg. Precision) and area under the ROC curve (AUROC) are shown in Table 4. The observed test values

of the Friedman test for both Avg. Precision and AUROC are 60.222 and 35.956 which are greater than the corresponding χ_2 value (i.e., $\chi_2(c, D_f)$). With the confidence interval α =0.05 and degree of freedom $D_f = 8$, χ_2 value is 15.51, obtained from the χ_2 table available in the literature. This results in the rejection of the null hypothesis, as shown in the last column of the table. This test confirms that there is a significant difference among the methods for both the accuracy measures. The proposed method ALP is considered as the control algorithm in the post hoc analysis.

5 Conclusion

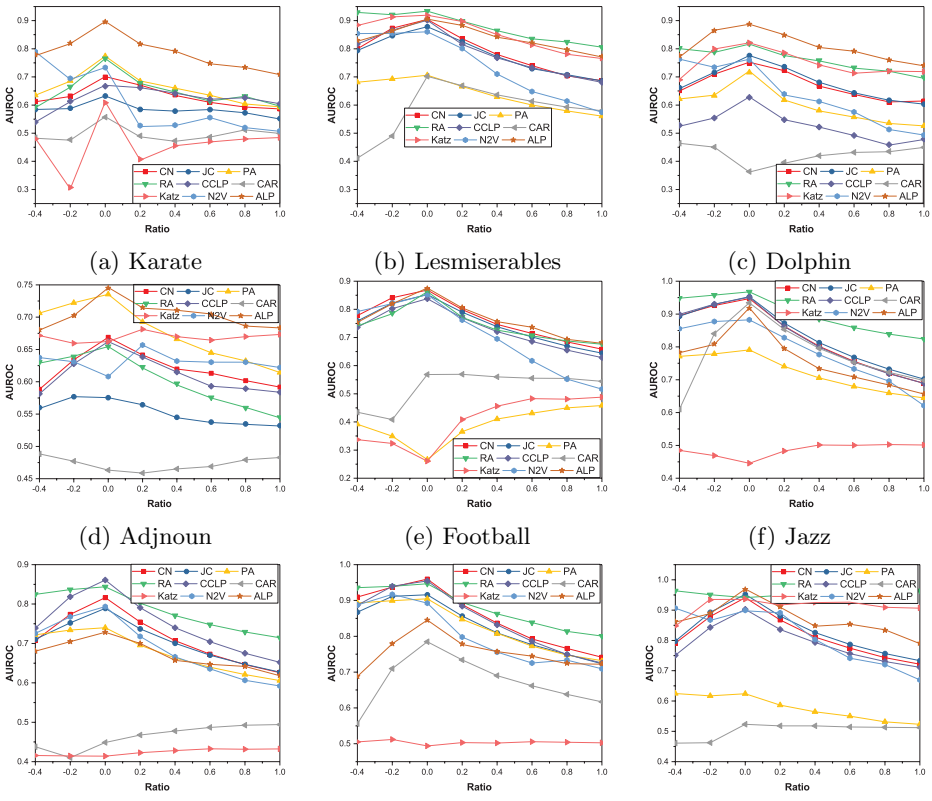
In this work, we incorporate an unsupervised framework of deep learning viz., graph autoencoder for link prediction in networks. Employing this architecture, useful latent representation of nodes is extracted and using these node embeddings, and the similarity matrix is computed for all node pairs. Moreover, missing links are identified based on this similarity matrix. The proposed method (ALP) is evaluated on average precision and AUROC, which show its superiority over the baseline methods. Robustness analysis against the noisy links (added or deleted) is shown, which represents the effectiveness of the proposed method.

6 Acknowledgement

We want to thank Ajay Kumar and Dr.Bhaskar Biswas for helpful discussions and suggestions during the early phases of the research. We would also like to thank the anonymous reviewers for sharing their helpful suggestions and Chaitanya Bhatia and Karan Siwach, for their valuable feedback.

Table 4: The Friedman test on Avg. Precision and area under the ROC Curve

	Dataset	IS-value									Test value	State Result
		CN	JC	RA	PA	CCLP	CAR	Katz	N2V	ALP	F_f	Is $F_f > \chi^2$?
Avg. Precision	Karate	0.148878	0.130261	0.069634	0.134948	0.162239	0.017582	0.260558	0.041504	0.653197	60.222	Null Hypothesis Rejected
	Dolphins	0.179974	0.205618	0.050494	0.082726	0.189028	0.007007	0.110008	0.024493	0.610708		
	Lesmiserables	0.249618	0.366998	0.249818	0.085681	0.256686	0.148050	0.085832	0.146678	0.701639		
	Adjnoun	0.079462	0.076121	0.018148	0.067761	0.084578	0.008218	0.105588	0.011761	0.242644		
	Football	0.459779	0.447029	0.111022	0.092703	0.468967	0.061658	0.178761	0.114445	0.681193		
	Jazz	0.326022	0.367671	0.319647	0.106431	0.328317	0.351645	0.263667	0.086058	0.691399		
	Celegansneural	0.118607	0.134162	0.036424	0.048702	0.146307	0.011771	0.051729	0.018931	0.743431		
	Usair97	0.126467	0.271941	0.231041	0.263769	0.141365	0.204706	0.050021	0.029793	0.370255		
	Political blogs	0.079613	0.146614	0.061253	0.015059	0.088426	0.063305	0.021715	0.008462	0.106545		
	Netscience	0.392733	0.433671	0.128983	0.002171	0.434501	0.108762	0.204778	0.081755	0.587527		
	Facebook	0.244687	0.172462	0.440321	0.019037	0.238218	0.235341	0.002152	0.125269	0.314704		
	Ca-GrQc	0.223600	0.063274	0.133345	0.019163	0.253608	0.217245	0.046797	0.048952	0.381180		
AUROC	Karate	0.693750	0.628375	0.757500	0.760375	0.656438	0.550500	0.611500	0.721125	0.897395	35.956	Null Hypothesis Rejected
	Dolphins	0.745418	0.769799	0.822843	0.726425	0.618911	0.357150	0.826348	0.750734	0.882697		
	Lesmiserables	0.889440	0.871515	0.934390	0.699676	0.888291	0.695770	0.912502	0.854812	0.906194		
	Adjnoun	0.665667	0.568315	0.647079	0.735380	0.653959	0.450198	0.658491	0.613725	0.743431		
	Football	0.873762	0.859288	0.854359	0.252409	0.813651	0.585301	0.854977	0.862271	0.879024		
	Jazz	0.948143	0.959044	0.963302	0.789540	0.955104	0.931445	0.452756	0.873276	0.909838		
	Celegansneural	0.815419	0.792798	0.848494	0.735148	0.872517	0.450223	0.416356	0.795693	0.720588		
	Usair97	0.958332	0.914826	0.946785	0.905626	0.957295	0.772429	0.500310	0.884882	0.838057		
	Political blogs	0.941012	0.907954	0.939749	0.934223	0.938549	0.739912	0.345143	0.866696	0.781857		
	Netscience	0.944599	0.95362	0.944769	0.639099	0.897433	0.532846	0.939401	0.892410	0.966444		
	Facebook	0.991824	0.989581	0.995177	0.832809	0.238218	0.944786	0.492362	0.991560	0.894836		
	Ca-GrQc	0.921563	0.929400	0.913091	0.741728	0.892601	0.605524	0.718201	0.908955	0.849303		



(a) Karate (b) Lesmiserables (c) Dolphin

(d) Adjnoun (e) Football (f) Jazz

(g) Celegansneural (h) Usair97 (i) Netscience

Fig. 2: Robustness

References

1. D. Liben-Nowell, J. Kleinberg, "The Link-prediction Problem for Social Networks," J. Am. Soc. Inf. Sci. Technol, 2007.
2. Z. Huang, X. Li, H. Chen, "Link prediction approach to collaborative filtering," in: Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries, JCDL '05, pp. 141–142.
3. E. Airodi, D. Blei, E. Xing, S. Fienberg, "Mixed membership stochastic block models for relational data, with applications to protein-protein interactions," Proceedings of International Biometric Society-ENAR Annual Meetings (2006).
4. L. Lu, T. Zhou, "Link prediction in complex networks: A survey," CoRRabs/1010.0725 (2010).arXiv:1010.0725.
5. L. A. Adamic, N. Glance, "The political blogosphere and the 2004 U.S.election: Divided they blog," in: Proceedings of the 3rd International Workshop on Link Discovery, LinkKDD '05, ACM, New York, NY, USA,2005, pp. 36–43.doi:10.1145/1134271.1134277.
6. T. Zhou, L. Lu, Y.-C. Zhang, "Predicting missing links via local information," European Physical Journal B 71 (2009) 623–630.doi:10.1140/epjb/e2009-00335-8.
7. A. Barabasi, H. Jeong, Z. Neda, E. Ravasz, A. Schubert, T. Vicsek, "Evolution of the social network of scientific collaborations," Physica A Statistical Mechanics and its Applications 311 (2002) 590–614.doi:10.1016/S0378-4371(02)00736-7.
8. L. Katz, "A new status index derived from sociometric analysis," Psychometrika 18 (1) (1953) 39–43.
9. W. Liu, L. L u," Link prediction based on local random walk", EPL(Europhysics Letters) 89 (5) (2010)58007.
10. S. Brin, L. Page, "The anatomy of a large-scale hypertextual web search engine," in: Proceedings of the Seventh International Conference on World Wide Web 7, WWW7, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, 1998, pp. 107–117.
11. E. A. Leicht, P. Holme, M. E. J. Newman, "Vertex similarity in networks," Phys. Rev. E 73 (2006) 026120.doi:10.1103/PhysRevE.73.026120
12. H. Tong, C. Faloutsos, J.-Y. Pan, "Fast random walk with restart and its applications," in: Proceedings of the Sixth International Conference on Data Mining, ICDM '06, IEEE Computer Society, Washington, DC, USA, 2006,pp. 613–622.doi:10.1109/ICDM.2006.70.
13. V. Mart ınez, F. Berzal, J.-C. Cubero," A survey of link prediction in complex networks," ACM Comput. Surv. 49 (4) (2016) 69:1–69:33.doi:10.1145/3012704.
14. L. Lu, C.-H. Jin, T. Zhou, "Similarity index based on local paths for link prediction of complex networks," Phys. Rev. E 80 (2009) 046122.doi:10.1103/PhysRevE.80.046122.
15. M. A. Hasan, V. Chaoji, S. Salem, M. Zaki, "Link prediction using supervised learning," in: In Proc. of SDM 06 workshop on Link Analysis,Counter terrorism and Security, 2006.
16. J. R. Doppa, J. Yu, P. Tadepalli, L. Getoor, "Learning algorithms for link prediction based on chance constraints," in: Proceedings of the 2010European Conference on Machine Learning and Knowledge Discovery in Databases: Part I, ECML PKDD'10, Springer-Verlag, Berlin, Heidelberg,2010, pp. 344–360.
17. P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," J. Mach. Learn. Res. 11 (2010) 3371–3408.
18. A. Krizhevsky, I. Sutskever, G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in: Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1,NIPS'12, Curran Associates Inc., USA, 2012, pp. 1097–1105.
19. M. E. J. Newman, "Clustering and preferential attachment in growing networks," Phys. Rev. E 64 (2001) 025102.doi:10.1103/PhysRevE.64.025102.
20. H. Cai, V. W. Zheng, K. C. Chang,"A comprehensive survey of graph embedding: Problems, techniques, and applications," IEEE Transactionson Knowledge and Data Engineering 30 (9) (2018) 1616–1637.doi:10.1109/TKDE.2018.2807452.

21. M. Belkin, P. Niyogi, “Laplacian eigenmaps and spectral techniques forembedding and clustering,” in: Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic, NIPS’01, MIT Press, Cambridge, MA, USA, 2001, pp. 585–591.
22. A. Grover, J. Leskovec, “Node2vec: Scalable feature learning for networks,” in: Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’16, ACM, New York, NY,USA, 2016, pp. 855–864.doi:10.1145/2939672.2939754.
23. S. Mehran Kazemi, D. Poole, “Simple Embedding for Link Prediction in Knowledge Graphs,” ArXiv e-prints (Feb. 2018).arXiv:1802.04868.
24. B. Perozzi, R. Al-Rfou, S. Skiena, “Deepwalk: Online learning of social representations,” in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’14, ACM,New York, NY, USA, 2014, pp. 701–710.doi:10.1145/2623330.2623732.
25. S. T. Roweis, L. K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” Science 290 (5500) (2000) 2323–2326.
26. H. Chen, B. Perozzi, Y. Hu, S. Skiena, “HARP: hierarchical representation learning for networks,” CoRR abs/1706.07845 (2017).arXiv:1706.07845.
27. D. Wang, P. Cui, W. Zhu, “Structural deep network embedding,” in:Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’16, ACM, New York, NY,USA, 2016, pp. 1225–1234.doi:10.1145/2939672.2939753.
28. S. Cao, W. Lu, Q. Xu, “Deep neural networks for learning graph representations,” in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI’16, AAAI Press, 2016, pp. 1145–1152.
29. T. N. Kipf, M. Welling, “Variational graph auto-encoders,” CoRRabs/1611.07308 (2016).arXiv:1611.07308.
30. M. Zhang, Y. Chen, “Link prediction based on graph neural networks,” CoRRabs/1802.09691 (2018).arXiv:1802.09691.
31. H. Wang, X. Shi, D. Yeung, “Relational deep learning: A deep latent variable model for link prediction,” in: AAAI, 2017, pp. 2688–2694.
32. X. Li, N. Du, H. Li, K. Li, J. Gao, A. Zhang, “A deep learning approach to link prediction in dynamic networks,” in: Proceedings of the 2014 SIAM International Conference on Data Mining, Philadelphia,Pennsylvania, USA, April 24-26, 2014, 2014, pp. 289–297.doi:101137/1.9781611973440.33.
33. A. Dadu, A. Kumar, H. K. Shakya, S. K. Arjaria, B. Biswas, “A study of link prediction using deep learning,” in: Advanced Informatics for Computing Research, Springer Singapore, Singapore, 2019, pp. 377–385.
34. T. N. Kipf, M. Welling, “Semi-supervised classification with graph convolutional networks,” CoRR abs/1609.02907 (2016).arXiv:1609.02907.
35. G. Salton, M. J. McGill, “Introduction to Modern Information Retrieval,” McGraw-Hill, Inc., New York, NY, USA, 1986.
36. P. V. Tran, “Learning to make predictions on graphs with autoencoders,” CoRR abs/1802.08352 (2018).arXiv:1802.08352.
37. B. Singh, S. De, Y. Zhang, T. Goldstein, G. Taylor, “Layer-specific adaptive learning rates for deep networks,” CoRR abs/1510.04609 (2015).arXiv:1510.04609.
38. T. Schaul, S. Zhang, Y. LeCun, “No More Pesky Learning Rates,” arXive-prints (2012) arXiv:1206.1106arXiv:1206.1106.
39. J. Shu, Q. Chen, L. Liu, L. Xu, “A link prediction approach based on deep learning for opportunistic sensor network,” International Journal of Distributed Sensor Networks 13 (4) (2017) 1550147717700642.arXiv:https://doi.org/10.1177/1550147717700642.
40. J. A. Hanley, B. J. McNeil, “The meaning and use of the area under a receiver operating characteristic (roc) curve,” Radiology 143 (1) (1982)29–36.doi:10.1148/radiology.143.1.7063747.
41. S. Geisser, “Predictive inference: An introduction chapman and hall,” NewYork (1993).

42. J. L. Herlocker, J. A. Konstan, L. G. Terveen, J. T. Riedl, "Evaluating collaborative filtering recommender systems," ACM Trans. Inf. Syst. 22 (1)5–53.
43. W. Zachary, "An information flow model for conflict and fission in small groups," Journal of Anthropological Research 33 (1977) 452–473.
44. D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten,S. M. Dawson, "The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations," Behavioral Ecology and Sociobiology 54 (4) (2003) 396–405.doi:10.1007/s00265-003-0651-y.
45. D. E. Knuth, "The Stanford Graph Base: A Platform for Combinatorial Computing," ACM, New York, NY, USA, 1993.
46. M. E. J. Newman, "Finding community structure in networks using the eigen vectors of matrices," Phys. Rev. E 74 (2006) 036104.doi:10.1103/PhysRevE.74.036104.
47. M. Girvan, M. E. J. Newman, "Community structure in social and biological networks," 99 (12) (2002) 7821–7826.doi:10.1073/pnas.122653799.
48. P. Gleiser, L. Danon, "Community Structure in Jazz," eprint arXiv:cond-mat/0307434.
49. D. J. Watts, S. H. Strogatz, "Collective dynamics of 'small-world' networks," Nature 393 (6684) (1998) 440–442.doi:10.1038/30918.
50. J. G. White, E. Southgate, J. N. Thomson, S. Brenner, "The structure of the nervous system of the nematode caenorhabditis elegans," Philos Trans RSoc Lond B Biol Sci 314 (1165) (1986) 1–340.
51. J. McAuley, J. Leskovec, "Learning to discover social circles in ego networks," in: Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, NIPS'12, Curran AssociatesInc., USA, 2012, pp. 539–547.
52. G. Kossinets, "Effects of missing data in social networks," Social Networks28 (3) (2006) 247 – 268.doi:https://doi.org/10.1016/j.socnet.2005.07.002.
53. J. W. Neal, "Kracking the missing data problem: applying krackhardt's cognitive social structures to school-based social networks," Sociology of Education 81 (2) (2008) 140–162.
54. P. Jaccard, "Distribution de la flore alpine dans le bassin des dranses et dansquelques r egions voisines", Bull Soc Vaudoise Sci Nat 37 (1901) 241–272.
55. "Link prediction with node clustering coefficient," Physica A: Statistical Mechanics and its Applications 452 (2016) 1 – 8.doi:https://doi.org/10.1016/j.physa.2016.01.038.
56. A. Kumar, S. S. Singh, K. Singh, B. Biswas, "Level-2 node clustering coefficient-based link prediction," Applied Intelligence 49 (7) (2019) 2762–2779.doi:10.1007/s10489-019-01413-8.
57. C. V. Cannistraci, G. Alanis-Lobato, T. Ravasi, "From link-prediction in brain connectomes and protein interactomes to the local-community-paradigm in complex networks," Scientific Reports 3 (2013) 1613.doi:10.1038/srep01613.
58. X. Glorot, Y. Bengio, "Understanding the difficulty of training deep feed forward neural networks," in: Y. W. Teh, M. Titterington (Eds.),Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, Vol. 9 of Proceedings of Machine Learning Research, PMLR, Chia Laguna Resort, Sardinia, Italy, 2010, pp. 249–256.
59. P. Zhang, X. Wang, F. Wang, A. Zeng, J. Xiao, "Measuring the robustness of link prediction algorithms under noisy environment," Scientific Reports6 (2016) 18881.doi:10.1038/srep18881.
60. J. Dem˘sar, "Statistical comparisons of classifiers over multiple data sets," J.Mach. Learn. Res. 7 (2006) 1–30.
61. M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," Journal of the American Statistical Association32(200)(1937)675–701.doi:10.1080/01621459.1937.10503522.
62. M. Friedman, "A comparison of alternative tests of significance for the problem of m rankings," Ann. Math. Statist. 11 (1) (1940) 86–92.doi:10.1214/aoms/1177731944.
63. Henderson, Keith and Gallagher, Brian and Eliassi-Rad, Tina and Tong, Hanghang and Basu, Sugato and Akoglu, Leman and Koutra, Danai and Faloutsos, Christos and Li, Lei, "Rolx: structural role extraction & mining in large graphs," Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, 2012.

64. Wang W, Tang M, Jiao P., "A unified framework for link prediction based on non-negative matrix factorization with coupling multivariate information," PLoS One. 2018;13(11):e0208185,2018, doi:10.1371/journal.pone.0208185.
65. Z. Wu and Y. Chen, "Link prediction using matrix factorization with bagging," 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS), Okayama, 2016, pp. 1-6, doi: 10.1109/ICIS.2016.7550942.

Authors

Aman Gupta and **Yadul Raghav**, we received a Bachelor's and Master's degree from the Indian Institute of Technology (BHU), Varanasi, majoring in Computer Science and Engineering. Our interest lies in the field of machine learning, deep learning, and data mining.

Aman Gupta

Yadul Raghav