SOCIAL INTERNET OF THINGS: THEIR TRUSTWORTHINESS, NODE RANK, AND EMBEDDINGS MANAGEMENT

A THESIS

Submitted in partial fulfillment of the

Requirements for the award of the degree of

DOCTOR OF PHILOSOPHY

IN

COMPUTER SCIENCE AND ENGINEERING

BY

ABHILASHA RANGRA

(Enrollment No.: 166205)

Department of Computer Science and Engineering

Jaypee University of Information Technology, Waknaghat

Solan–173234, Himachal Pradesh, INDIA

AUGUST 2021

CANDIDATE’s DECLARATION

I hereby certify that the work, which is being presented here in the thesis entitled, SOCIAL INTERNET OF THINGS: THEIR TRUSTWORTHINESS, NODE RANK, AND EMBEDDINGS MANAGEMENT, submitted in partial fulfillment of the requirements for the award of the degree of Doctor of Philosophy in Computer Science and Engineering and submitted in Department of Computer Science and Engineering, Jaypee University of Information Technology (JUIT), Solan is an authentic record of my own work carried out (July 2016–July 2021) under the supervision and guidance of Prof. Dr. Vivek Kumar Sehgal–Jaypee University of Information Technology, Solan

I hereby declare that this Doctor of Philosophy thesis, my original investigation and achievement has not been submitted by me for the award of any other degree on this work in any other Institution/University.

Abhilasha Rangra

DATE: 4TH AUGUST 2021

ABHILASHA RANGRA
(Enrollment No.: 166205)

We certify that we have read this thesis and that in our opinion; it is fully adequate in scope and quality as a thesis for the award of degree of the DOCTOR OF PHILOSOPHY in Computer Science and Engineering.

Vivek Sehgal

PROF. DR. VIVEK KUMAR SEHGAL
(SUPERVISOR)
Date: 4TH AUGUST 2021

ABSTRACT

The stark decrease and the price per computation afforded by rapidly advancing processors have led to an explosion in the adoption of embedded devices. In contemporary times it seems that everything has embedded systems on chips incorporated into their inner workings. Things we all value so much as embedded device adoption soars there is a natural progression from the use of standalone devices to networking these embedded devices. This is known as the Internet of Things (IoT). IoT provides an enormous basis for data generation potential as these sensors can go where people cannot and never get tired. As a result, these devices are capable of fundamentally changing how we interact with our environments and lifestyle choices. Think of the example of how smartwatches have led many to change their exercise habits as these devices afford health monitoring tools only available to elite athletes not even a decade ago. There exist other opportunities for these IoT devices such as remote control of personal devices regardless of geographical boundaries and the provision of a basis to automate tasks currently seen as mundane.

Advances in integrated Systems-on-Chip (SoC) along with advances in mobile telecommunications have led to ease and networking of these IoT devices together. Provided an avenue for these devices to seek out their peers automatically and lead to the potential for synergistic behavior in their parallel operations. This automated device connection-seeking behavior is referred to as service discovery which enables trustworthy, secure connections that enable secure collaboration between connected IoT devices. This leads to the emergence of a new IoT paradigm, the Social Internet of Things (SIoT). When SIoT was proposed at the beginning of the previous decade, IoT devices were reframed as social objects. Devices that have human-like social relationships with each other by reframing the concept of IoT connections. In this manner existing techniques applied in social media networks could be leveraged in IoT device management providing a pathway for rapid scaling of emerging IoT networks along with a potential basis for yet to be established IoT services and applications. There exist many possible relationships in the context of SIoT as an example the co-location co-work based relation. This is based on the spatial properties derived from the network topology along with the devices location and influence in the network. Another is the social friendship and ownership relation which is based on the relationships between the owners of the SIoT social objects. A third is the social object relation which is an ad-hoc criterion that's established automatically upon the device contact and is dynamically updated based on changing requirements of the network. One of the problems that can be solved using the SIoT concept is the development of efficient edge computing resource allocation and large scale IoT networks. Mobile edge computing is a relatively new distributed computing framework which aims to allocate some computational tasks away from the data center, the traditional home for cloud computing resources. The research team has developed a generic framework to address this problem by exploiting the social connections between IoT devices.

The thesis entitled "SOCIAL INTERNET OF THINGS: THEIR TRUSTWORTHINESS, NODE RANK, AND EMBEDDINGS MANAGEMENT" deals with the analysis of relations/interactions between smart entities of SIoT. The analysis work is done as graphical representation where the link of the graph is information/knowledge, the mapped network structure is the complete graphical algorithm, and similarity measures in graphical components are the similar data points connected with each other. This dissertation presents the traditional methods to implement various machine learning and deep learning algorithms in the graphical representation of SIoT. The various outmoded features are used for node level, link level and, graph level predictions. The compatibilities measures using stable matching algorithm are also discussed. The application of machine learning algorithms on SIoT with large feature vector is very complex as it indulged in large computations. The embedding techniques are used to map high dimensional feature vector into low dimensional spaces for many downstream predictions. The degree of importance of various nodes in a social network of IoT is also calculated.

The aim is to search for active accessible and trustworthy devices that can handle the requested computational tasks and select the devices such that the total service latency is minimized. The concept of the SIoT has the potential to revolutionize how future networks of embedded systems may cooperate in their operation. This has the potential to lead to revolutionary data fusion processes that further amplify the benefits that may be realized from the implementation of these systems. As a result, of the efforts by IoT researchers early results are promising from this, there still are many opportunities for advancing the state of the art. One such application is community detection for disaster management, discussed in this dissertation.

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to thank all the persons in my life for their support and encouragement for my research. First of all, I would like to thank my advisor Prof. Dr. Vivek Kumar Sehgal for constant guidance, encouragement and support. Without guidance and motivation from him, my research would not have been possible. His deep knowledge of Internet of Things, or other subjects have always helped me, not only towards my research but also in other domains of life. His dedication towards his subjects and work has inspired me to remain dedicated towards the goals for life. I would also thank my babaji Mr Yogiraj Swami Amarjyoti, my family members Mrs Anita Rangra, Mr Shakti Rangra, Brother Mr Abhilaksh Singh Rangra, Sister-in-law Mrs Meenakshi Rangra, Nephew Mr Hanumant Singh Rangra, and all my friends Ishani, Shashi, Atul, Jatin, my grandparents and all my relatives for continuous support.

Date: 4th August 2021

[illegible]

Signature:

Abhilasha Rangra (166205

SPECIAL THANKS TO

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY,

WAKNAGHAT, SOLAN, HIMACHAL PRADESH, INDIA

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 *Problem Statement*

IoT includes a huge number of smart objects that communicate with one another automatically using several communication protocols. They also jointly interact with their neighbors to achieve common objectives. When the number of devices connected to the internet grows, finding the right device to deliver the desired service becomes increasingly important. SIoT includes social networking ideas into the Internet of Things. Smart objects may use only local knowledge to locate preferred resources between their friends in a decentralized manner. The main goal of the SIoT is to connect the two layers one is of people and one is of smart objects. The most important responsibilities are to consider the best strategies for the development and organization of social associations between items. According to a survey by 2025, it is expected that each human will have 3 to 4 smart devices as shown in Figure1.1. Since each device is IP enabled. The network created by these smart devices will exactly replicate the social network between humans. The exponential growth of the smart devices in IoT applications attract the researchers to address various open research issues. By drawing inspiration from social computing, the SIoT aims to solve the problems of the IoT, such as resource discovery, trust, and scalability [1]. This thesis work encourages the authors to separate the two levels of peoples and things.

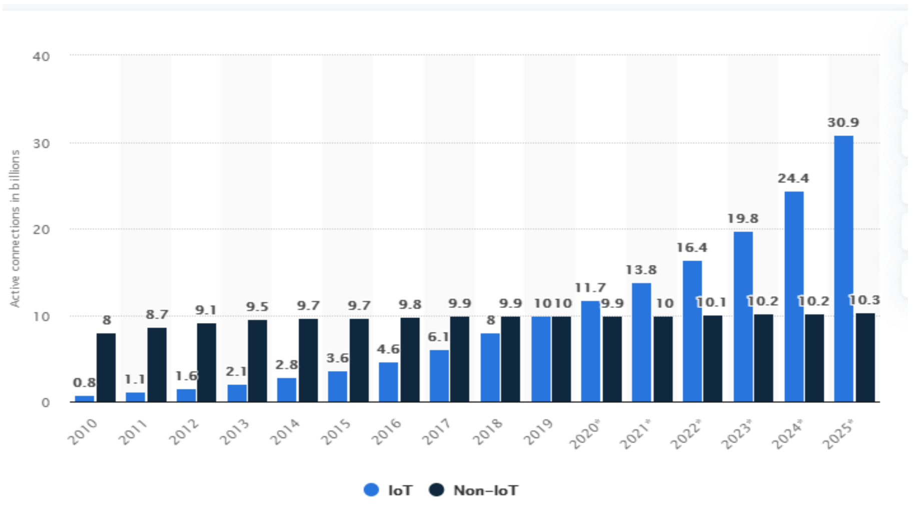


Fig 1.1 Internet of Things (IoT) and non-IoT (Humans) Active Device Connections Growth from Source: https://www.statista.com/statistics/1101442/iot-number-of-connected-devices-worldwide

The literature survey reveals that Social Network Analysis offers an opportunity to see the world through a network through a network paradigm, where IP enabled smart objects establish their own network according to the application domains like: event graphs, pandemic network, cyber physical systems, economic network, molecules, and knowledge graphs as shown in Figure 1.2.

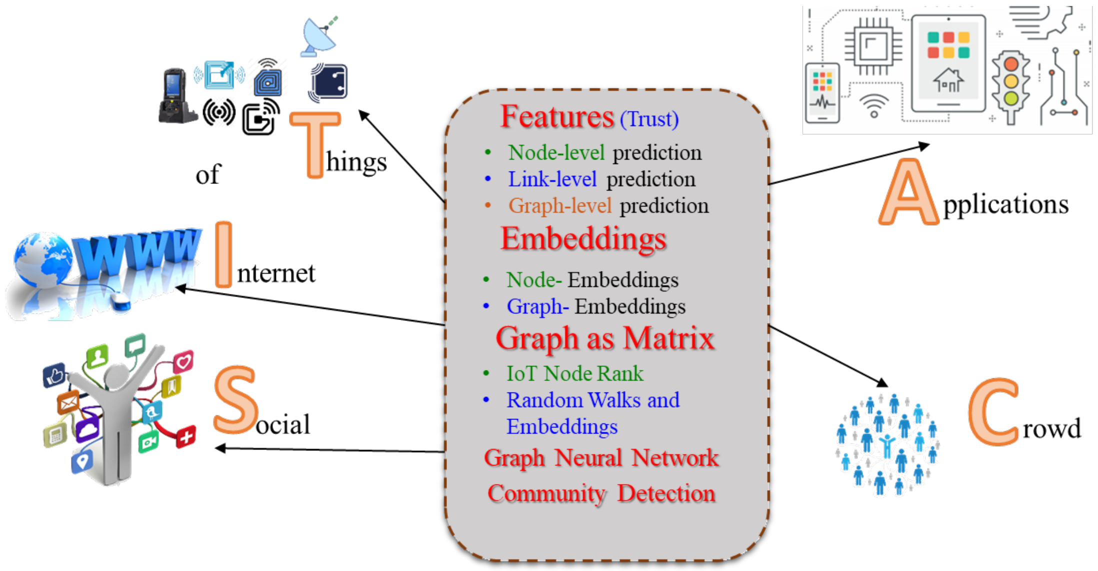


Fig 1.2. Social Internet of Things

1.2 *Contributions*

This thesis includes several key contributions to the SIoT: their trustworthiness, node rank, and embeddings management. The identified research problems are mapped into the following research objectives:

1. Trustworthiness computation using:
    a. Node-level prediction.
    b. Link-level prediction.
    c. Graph-level prediction.
2. To map nodes and graphs into an embedding space used for many downstream predictions.
3. Node rank solution using power iteration of the stochastic adjacency matrix (G) with Markov Chains.
4. Community detection for disaster management.

Stage I

i. Characterization of node level prediction tasks and features

ii. Characterization of link level prediction task and features

iii. Graph kernels and graph level predictions
iv. Trustworthiness Measure using Stable Matching

Stage II

i. Nodes Embedding
ii. Node2vec
iii. Feature learning framework
iv. Biased random walk
v. Spectral clustering

Stage III

i. Social IoT Node/Page Rank Flow Model
ii. Power Iteration
iii. Brin-Page Rank equation

Stage IV

i. Modularity clustering
ii. Louvain method for community detection
iii. Natural disasters management

This thesis embodies the subject matter resulting out of this study and is arranged in seven chapters.

1.3 *Thesis Outline*

The thesis has seven chapters out of which Introduction is covered in CHAPTER 1. CHAPTER 2 presents literature of the SIoT, trustworthiness as function of nod, link and, graph prediction. The embedding management with the rank of importance of a node is also discussed in the literature. The preliminary notations are introduced to keep the clarity of usage throughout the thesis. CHAPTER 3 presents the node level task to place the smart things in network, link level task to communicate between smart things in network, and graph level features that characterize the structure of an entire graph. CHAPTER 4 presents mapping nodes and graphs into an embedding space used for many downstream predictions. Partitioning via spectral clustering is also discussed in this chapter. CHAPTER 5 presents the Node rank solution using power iteration of the stochastic adjacency matrix (G) with Markov Chains. CHAPTER 6 presents the community detection for disaster management by using modularity clustering and Louvain method. Finally, CHAPTER 7 highlights the thesis conclusion, which is supported by the results of experiments and simulations, as well as the research work's future scope.

CHAPTER 2

PRELIMINARIES AND BACKGROUND

This chapter represents the organized review of literature on Social Internet of Things. It is further categorized into four segments. These segments are as follows:

i. Features for trust
ii. Embedding's
iii. Graph as a Matrix
iv. Community detection

2.1 *Features for Trust*

The Internet of Things (IoT) includes a huge amount of smart objects that communicate with one another automatically using several communication protocols. They also work together with their neighbors to achieve shared objectives. When the number of devices connected to the internet grows, finding the right device to deliver the desired service becomes increasingly important. The Social Internet of Things (SIoT) includes social networking like Facebook, Twitter, Instagram etc. ideas in the direction of Internet of Things. Smart objects may use only local knowledge to locate preferred resources between their friends within decentralized way. **Arjunasamy *et al.* [1]** applied for improving complete network navigability, a heuristic for efficient friend selection. An average path length, huge component, the average degree of connections, and local clustering coefficient analysis are executed. The Internet of Things includes us with a plethora of previously unthinkable interests. The agreeable independent interchange between IoT and social systems is growing which is taking advantage of the current positive approach of SIoT. The main goal regarding SIoT is to divide the two layers of things and people to encourage things to have their relational relationship, only access the result of self-governing in the middle question relations occurring in the protest's informal culture allowing people to urge rules to ensure their security and. Even the most magnificent protest will have little effect; social issues will. The most important responsibilities are to consider the best game plans for the development and organization of social associations between items. To depict a potential roadmap as long as IoT that combines features for making tasks easier to relate to one another. The SIoT perspective's goals are obvious: to easily approach the slow down result of self-administering allying question affiliations happening on articles social framework and to enable individuals to drive guidance to ensure their protection [2]. The constant movement based on objects which establish transient social connections is one based on major difficulties in SIoT. Non-performance within chain of links inside network navigability, SIoT, as well as tough in service composition and discovery, all occurred by relationship topology dynamics. Other models planned to be the same as SIoT but take a dissimilar direction in design and structure include Mobile Social Network (MSN) and the Social Internet of Vehicles (SIoV). **Esfahani *et al.* [3]** establishes a modern concept we call the Mobile Social Internet of Things (MSIoT). This definition incorporates SIoV, MSN, SIoT, trends and removes the difficulties represented by the object's continuous mobility. Because of its extension, trends united into a standard framework and architecture. Since objects in SIoV, MSN, SIoT, play dissimilar roles, we propose the latest classification for objects in these systems to incorporate these definitions. We then implement the new MSIoT framework and identify two key components, which are Service Forwarder (SF) and Mobility Management (MM). There are many advanced movement-friendly social connections represented, each encompassing broad cordial social links in such a way that covers wide range regarding social situations. Lastly, we investigate how the offer social relationships and new elements act in MSIoT using a framework. In recent decades, the Internet of Things (IoT) model has gotten a lot of notice in industry and academia. The incorporation of social networks and IoT recently been introduced as a means to encourage further growth. This article established on the challenges, solutions, applications, of SIoT over collective edge computing, which takes care of the interests of both social relationships and mobile edge computing among SIoT users. The use of streaming data processing, caching, distributed offloading is demonstrated first, led by examples of solutions such as federated learning, alliance games, auctions. Lastly, many research issues discussed. **Dong *et al.* [4]** contribution on this given article: 1) we explain why the three methods presented should be used in SIoT; 2) we define the role of social relations in traditional IoT applications and how they affect person selection; and 3) we explore how the discussed problems can help to improve intelligent SIoT frameworks, stable, robust in the future. By drawing inspiration from social computing, the SIoT aims to solve problems of the IoT, such as resource discovery, trust, and scalability. **Vahdat *et al.* [5]** purposed for this article is to look at SIoT research from two aspects: integration with new computing models, application domain. The 2-dimensional structures are suggested, along with designs also examined respectively. 1-dimension categorizes along with examines accessible investigate out of an implementation state viewpoint, while 2nd dimension does the same from the perspective of integrating latest evaluating architecture. Its main goal is for expressing Social Internet of Things in technical terms, identify relevant research, promote the circulation of new, as well as discuss upcoming testing regulations within this area.

The IoT belongs to state-of-the-art world in which billions of objects smartly interact with one another. The important field in this area is the latest model of the SIoT, which incorporates IoT along with social networks inside a creative way. SIoT is a simulation of object-to-object and human-to-human social networks. Human establishes their social network to solve problems like productivity, success, improving functionality to provide the services they need. **Rad *et al.* [6]** proposed a detailed analysis evaluation out of SIoT method for discussing along with evaluating latest research inside field. As a result, we focused our research on the SIoT features, parameters, and challenges, key components. They include the final SIoT framework based on the scholarly contributions and a scientific taxonomy for investigation. SIoT links the social network well alongside IoT becoming summery topic because of networking services and of its ability to help novel IoT applications in more productive ways.

Fu *et al.* [7] contributed for this paper is to quantify in what way to explore work on social relationships inside IoT. To begin, personally note so as browser acts simultaneously conduit between IoT and the social system, and then they suggest a Search SIoT architect of SIoT development. Second, they use different types of efficiency benchmarks: average size, network stability, degree distribution, system density, system diameter, betweenness regarding users. It shows in Search SIoTs, the degree distribution maintains an accelerated mean interval shrink, power law, system density, system diameter; network security, betweenness regarding users all are higher. Finally, they demonstrate quantitatively a browser speeds up a spread of vicious source in SIoTs.

The increasing difference between Big Data analytics and users necessitates the development of innovative tools that solve the velocity, volume, variety of big data. As a result, selecting features and analysing from such a large amount of data becomes computationally inefficient. **Ahmad *et al.* [8]** explained a machine architecture that uses an Artificial Bee Colony to pick features (ABC). A Kalman filter used in the Hadoop ecosystem to extract features. In addition, conventional Map Reduce with ABC is employed to improve processing speed. Furthermore, a four-tier framework is proposed that easily categorizes data, removes redundant information, analyses information considering suggested Hadoop-based ABC method. Personally executed the suggested network considering Hadoop-Map Reduce accompanied by ABC method to check the productivity regarding suggested methods accomplishment inside a suggested system structure. The ABC techniques applied for a pick characteristic; duration Map Reduce relies on parallel method to process large amounts of data quickly. The framework is implemented in near-real-time using the Map Reduce tool on top of Hadoop parallel nodes. Furthermore, using ten separate data sets, the method is compared to Swarm approaches and examined in terms of throughput, performance, accuracy. The experiments demonstrated that the proposed scheme is more effective for selecting candidates.

Significant problems related to the development and design of things services must be resolved considering IoT to progress. **Atzori *et al.* [9]** suggested a potential solution to such problems. They present the SIoT, a latest model for the "social network of intelligent objects" based on social connections between objects. A proposed framework for the implementation of SIoT described after the specification of a specific social structure among objects. The ability of devices for locating and human, select, utilize items via belonging to utilities within IoT enhanced by SIoT paradigm.

Atzori *et al.* [10] findings of this article in this context: a) we establish proper strategies for managing and establishing social connections among objects inside a method because perhaps corresponding to social system is passable; b) they identify potential IoT structure that involves its utility designed for managing items inside social system; c) they identify the effect of the SIoT network structure using simulations.

The issue of understanding how the information given by members of social IoT must be handled to create a reliable framework due to the actions of the objects is the subject of such a case. **Nitti *et al.* [11]** starting with the proposed solutions for social networks and P2P, we identify two approaches for trustworthiness management. Mostly the subjective model, every node calculates the trustworthiness from its friends based on its own opinions and experience of friends with whom it shares mutual interests. The information about every node is stored and distributed using a distributed hash table structure an objective model so that any node can access the same data. Simulations demonstrate what the experimental results can easily isolate almost any suspicious nodes in the network while increasing network traffic through feedback exchanging.

Ding *et al.* [12] Social Networks, IoT, the Internet all are clustered together in this article. The growth of social structures along with IoT would be supported by clustering. Many macro components within human society can be summarized and monitored using the clustered platform. As a result, scientists can easily examine the actions of people and objects as information. This article also describes the clustered platform's potential implementations.

Big data collection of data is just an easy way to gather knowledge that the collector is involved in. Even so, there is no guarantee that the data provided by the users is accurate. The trustworthiness of users who participated in the collection becomes crucial because the collector cannot verify the authenticity of any piece of information. **Yu *et al.* [13]** developed an improved method for calculating user data collection trustworthiness in the context of big data. They split trustworthiness into two categories: similarity and familiarity, and investigate the effects of user behavior on trustworthiness. They also design a security queue to record user's previous trust information, so that we can identify malicious users with great accuracy. This involves protecting users from increasing their trustworthiness and presenting misleading information that could deceive final results. The outcomes of results illustrate its suggested method will withstand malicious user behavior with sensitivity.

Azad *et al.* [14] illustrates a different technique build on updating and computing the trustworthiness of SIoT network members in such a self-enforcing manner, without the need for a trusted third party. In a decentralized setting, homomorphic encryption is used to protect the identity of SIoT participants. To establish self-enforcing properties, each device's trust score is gradually modified based on its former trust score and the most current count of votes cast by its members of the network, using zero-knowledge proofs (ZKPs) to ensure that each participant observes the procedure honestly. By prototyping the system's key features, we test the performance of the illustrated scheme and current evaluation standards. The efficiency findings demonstrate, as the network grows larger, the system's computation and coordination overheads grow linearly. Besides that, they demonstrate the proposed system's privacy, protection, correctness, in a malicious adversary model.

Boudagdigue *et al.* [15] suggested a model for complex trust management that is appropriate for industrial settings. They also suggest transforming IIoT networks in manufacturing facilities from their conventional centralized architecture to a hybrid architecture consisting of a collection of modern industrial relationship principles. This work's evaluation plan is split into two sections. They compare their proposed design to the conventional architecture of the plant's IIoT network in the first section. The findings of such an analysis indicate that their architecture is better suited for simplifying IIoT system confidence management. The capacity, resiliency, adaptability of ours suggested trust model toward alterations of IIoT nodes in malicious environments were demonstrated in the second section.

Khan *et al.* [16] contributed to this article are highly significant. The basics of trust and SIoT principles in SIoT, and also the differences and similarities between SIoT and IoT. We include a thorough overview of the trust management strategies presented in this research for SIoT over the last six years. Then, using modified investigation with regard to the trust direction method, they evaluate advanced trust direction techniques invented as long as SIoT. They define and address the requirements and difficulties regarding different type of SIoT, as well as difficulties to make and accessing reliability between communicating social objects.

Sherchan *et al.* [17] the first systematic analysis of computer science and social literature on trust in social networks is represented in this paper. We begin by reviewing existing trust constructs and defining social trust throughout the realm of social networks. Following that, we'll go through some recent research on three dimensions of social trust: trust dissemination, trust assessment, trust data collection. At last, they try comparing the research in recognizing areas in which more research is needed in the field of social trust.

Kourtellis *et al.* [18] suggested a new method for identifying nodes accompanied by large betweenness consistency. They presents k-path consistency, new metric, distributed method because of calculating, experimentally demonstrates which nodes regarding large-path consistency often hold large node-betweenness consistency. A randomized technique takes $O(k^3n^{2-2\alpha}\log n)$ time to run and returns an approximation of each vertex v, k-path consistency till an additional mistake of $\pm n^{\frac{1}{2}+\alpha}$ including probability$1-\frac{1}{n^2}$. As compared to existing randomized algorithms, the proposed technique on synthetic and real social links illustrates enhanced precise in determining huge betweenness consistency nodes, substantially decrease the implementation duration.

The index of betweenness centrality is crucial in social network analysis, although it is time-consuming to calculate. The quickest algorithms currently available involve $O(n3)$time and $O(n2)$space, where n represents the size of participants in the system. **Brandes *et al.* [19]** discussed a new algorithms for betweenness, which is motivated by the rapidly increasing need to compute centrality indices on massive, but sparse, networks. On weighted and unweighted networks, they need to run $O(nm)$, $O(nm+n\,2\log n)$time, and $O(n+m)$space, respectively, where m represents the number of connections. This significantly expands the number of networks for which centrality analysis is possible, according to experimental proof.

Abbasi *et al.* [20] investigates whether writers with different levels of centrality have different levels of preferential attachment in scientific authorship networks. They demonstrate that the betweenness centrality of an existing node is a significantly good predictor of favorable connection by new companies than a degree of closeness centrality, and use a comprehensive database for the scientific specialty of steel structure analysis. Preferential attachment changes from degree centrality for betweenness centrality as just a global measure as a network grows.

Geisberger *et al.* [21] This article, it suggest the structure to count the amount based on smallest links moving between node using betweenness, as it will be used for approximation algorithm. This will yield great new schemes and approximations even for unimportant nodes than ever occurred before for many real worlds inputs.

As per the assumption, the information is spread usually on the shortest paths, the betweenness measure calculates the fraction of the shortest paths between the node pairs which relaxes the assumption. Betweenness measure gives more priority to the shortest path using matrix methods but still, it also contributes to all the essential paths [22].

Alamsyah *et al.* [23] categorizes the social network analysis on its graph representation. Social structure investigation follows three approaches: Graph Presentation, Content Extracting, along with Semantic Examinations. For first approach symbolizes for investigating social system topology, system modelling, bind-strength, community detection, group cohesion visualization, along with metrics calculations.

The SIoT, which combines the IoT and Social Networks model, has been developed to create a network of smart nodes able of forming social connections. SIoT is the most dependable service provider for reducing the risk of being exposed to malicious nodes since it provides a social trust model based on node centrality and similarity metrics to extract trust behaviors among social network nodes. The SIoT model starts with a flexible bipartite graph, and then uses the Hellinger distance to construct social networking among the service requestor nodes. **Aalibagi *et al.* [24]** finds that SIoT is resistant to a variety of network assaults and can effectively identify the most appropriate and trustworthy service provider.

Apart from promoting its digital and intelligent level, Ubiquitous Power Internet of Things (UPIoT) brings some uncertain social factors for cyber risk brewing. So understanding and evaluating the cyber-security risk of UPIoT is an important guarantee for Energy Internet Construction. **Zhao *et al.* [25]** experiments results proves that using AHP algorithm we can identify key risks in the network as we as it also reflects the security status of the network using.

Malik *et al.* [26] investigated centrality as just a heuristic for selecting a new object relation, as well as network parameters including average degree, average path length, clustering coefficient. The test results for different forms of centralities were also examined.

Mehmood *et al.* [27] evaluates the explanation applicability within IoT area, a prophecy structure construct on measures of having social structure analysis techniques, including the different type of centralities: degree centrality, betweenness centrality, eigenvector centrality is suggested. For validating an accuracy regarding proposed model, a puzzled matrix introduced.

Riquelme *et al.* [28] investigates the feasibility of using the linear threshold model to define centrality measures for labelled and weighted social networks in this article. They suggest a new centrality metric, the Linear Threshold Rank (LTR), for ranking network users, as well as a centralization measure, the Linear Threshold Centralization, for determining how centralized the entire network is (LTC). We equate our new measures to two centrality measures based on another centrality measure and relevance criteria based on the independent cascade model in four separate social networks. The findings indicate that the metrics can be used to network and rate actor's recognizable way.

Borgatti *et al.* [29] suggests a network flow typology characterized by two dimensions of variation: the method of spread and the types of trajectories that traffic can take. The types of flows that each measure of centrality is suitable for are then balanced. The relationship between the mode of flow and the relative value of nodes in terms of main measurements including flow frequency and reception speed is investigated using simulations. It is demonstrated the centrality measure formulas are completely valid just for the particular flow processes for which they were developed, and that when they are applied to other flow processes, they produce the "false" result. A key argument made in this article is that, given implicit models of how centrality measures traffic flows, can be regarded as determining an absolute value for certain types of node outcomes, and that this offers a useful pattern of learning about centrality.

A practical, authentic suggested structure modelled, where social trust-modelling problem is being addressed formed on customer's characteristics in a Social structure. **Davoudi *et al.* [30]** proposed for predicting classification as long as customized suggested structure construct on similarity, consistency, and social connections. The initial results on the real-world dataset show better results from suggested system in future enhancing correct regarding classification forecast within social suggested networks. IoT along with underlying Big Data technology can be used for traffic analysis of different parts of cities and quickly identify the hotspots. This model will be at par with modern-day systems for traffic management. This will surely help in advancing our telecom infrastructure and smart city planning and development [31]. For better delivery fulfilment of messages in scarce Mobile Ad hoc Networks (MANETs), SimBet Routing suggested. It uses an interchanging regarding early assessment 'betweenness' centrality measure or nearby established social 'similarity' up to final node. This routing mechanism better than Epidemic Routing as it offers less overhead. It also performs better than Routing especially when connectivity is less between sender and receiver nodes [32].

Niwattanakul *et al.* [33] uses for getting better information through search engines, a similarity measurement method is being proposed to calculate the similarity between words by using Jaccard Coefficient. A measure to compare the similarity between sets of data has been made with Jaccard implemented with the Prolong programming language. Precision, recall, and F-measure was used for better performance of this proposed measure.

SIot a next step towards IoT which combines networking concepts with IoT. **Aljubairy *et al.* [34]** proposed a framework with has 3 stages: i) gathering based on raw motion information regarding IoT schemes, ii) manufacturing brief order structures regarding SIoT, iii) projecting connections between IoT schemes which is possibly to happen. Supervised machine learning techniques have been made on the concepts of structured-based analysis which falls under Social network analysis (SNA). **Kumari *et al.* [35]** help in predicting the chances of establishing the links in the future. Feature vector has been prepared after considering different based on structured similarity measures for each non-existing link inside structure.

An industrial Internet of Things (IIoT) now being joined with social networks. **Qiu *et al.* [36]** suggested for using the decision tree ensembles, a directed edge weight prediction model

(DEWP). Local similarity values expanded to DWNs, but each edge receives a sequence of similarity indices among nodes via attributes. These attributes were utilized to create mixed regression architect that included random forest, gradient boost decision tree, extreme gradient boosting, along with a light gradient boosting machine. DEWP offers better speculation validity along with the robustness when compared with other existing algorithms.

Kumar *et al.* [37] proposed a latest similarity index SHOPI (Significance of Higher-Order Path Index) that minimizes restriction of data issue in between the similar neighbors because of adding a penalty to them. Discriminating features based on higher-order paths are used to penalize larger links accessible among seed node link. At early the consequences show that SHOPI gives better performance than the baseline methods.

Peng *et al.* [38] suggested a different motif-established application graph CNN because of graph division, whatever is based on deep learning-based graph embedding approaches. Such a network could grasp additional particular along with a well-off graph attributes. To better preserve the spatial information, a motif-matching guided subgraph normalization method is developed. Graph classification performance is observed to be better than traditional graph kernel methods and recent deep learning approaches, based on initial data.

Utilizing kernel random attributes inside the graphlet architecture, the theoretical connection with a mean kernel metric is established. This technique is too expensive for a typical random attribute, we can employ optical random attributes that can calculate in fixed duration. The resulting algorithm is better than the graphlet kernel in terms of speed and accuracy [39]. Nowadays encoding of data is mostly done in the form of graphs. In physical chemistry biochemical compounds are represented in the form of graphs where atoms are represented by vertices and edges denote the chemical bonds between the two atoms. **Wongsriphisant *et al.* [40]** by using the method of graph classification they find the functions of molecular of a biochemical compound. An algorithm was introduced to extract substructure primitives of compounds and then creating a graph that can be provided with a support of a vector machine using kernel graphs. The performance of this graph is giving an accuracy of 0.847 which is better than solely graph kernels.

Table 2.1 Literature for, Features for Trust

Referenced Articles	**Main Topic**	**Strengths**	**Weakness**	**Future Scope**
Dipraj *et al.* (2020) [2]; Social Internet of Things.	IoT, SIoT	-Comprehensive systematic literature overview of SIoT -Brief background on SIoT	There is no statistical analysis	-Comparative analysis -New components and social relationships behave
Vahdat *et al.* (2020) [5]; Social Internet of Things and New Generation Computing.	Social networks (SNs), Social objects			
Nitti *et al.* (2014) [11]; Trustworthiness Management in the Social Internet of Things.	Trustworthiness Management,	Establishment and the management of social relationships, between objects	Limited Features are used	Development of secure, robust, and intelligent SIoT frameworks.
Boudagdigue *et al.* (2014) [15]; Trust Management in Industrial Internet of Things.	Trustworthiness familiarity and similarity,			
Geisberger *et al.* (2008) [21]; Better approximation of betweenness centrality.	Different ways to model importance: •Betweenness centrality •Closeness centrality	Social Objects, Social Relationships based on node features	A few Importance-based features has been discussed	Importance-based features and Structure-based features can be used for trust calculation
Borgatti *et al.* (2005) [29]; Centrality and network flow.	Different ways to model importance: •Eigenvector centrality •Ego centrality •Katz centrality			
Qiu *et al.* (2021) [36]; A Directed Edge Weight Prediction Model.	Predict new links established on: •Distance-based feature	Social Objects, Social Relationships based on Link features	Trustworthiness graph with link prediction is not discussed	Link Prediction Task for trust calculation
Ajay *et al.* (2020) [37]; Link prediction in complex networks.	Predict new links based on: •Local neighborhood overlap •Global neighborhood overlap			
Ghanem *et al.* (2020) [39]; Fast Graph Kernel with Optical Random Features.	features that characterize the Structure of an entire graph.	Kernel methods to measure the similarity between two graphs	Trustworthiness for graph is not discussed	Compatibilities between two subgraphs using stable matching

2.2 *Embedding's*

Goldberg *et al.* [41] introduced by word2vec software of Tomas Mikolov and colleagues1. To understand the equation which worked on the neural network's language –modelling the crowd was a bit difficult. Then the equation of negative sampling was explained which was found in the "Distributed Representations of Words and Phrases and their Compositionality".

Perrozi *et al.* [42] presented a new approach as DeepWalk which was used to learn vertices representation in the form of latent in a network. It encoded the continuous vector space in the form of social relations in which statistical models were used for exploitation. The sequence of words to graphs is used to advance in modelling languages and deep learning. After representing its work on the multi-label classification of networks for networking socially. The results show that it can outperform the view of the network which is done globally. The result shows the score which is 10% higher than the data that is labelled is sparse. DeepWalk's method is performing better even after using 60% less method of training. The qualities of these methods are scalable, results are increasing, and parallelizing trivially which makes it suitable for applications such as network classification and anomaly detection.

Node2vec was proposed in which a framework of algorithmic is used for learning the representation of a node in a network. **Grover *et al.* [43]** suggested an algorithm, in which we learn nodes mapping to space of low dimensional that maximizes the probability of network preserved of nodes neighborhood. It is used to design a procedure that explores neighborhood diversity. The demonstration of node2vec is done on top of extant advanced methods upon multi-label categorization, then it is linked with real-world networks from domains of diversity. So the new way is introduced in which state of art task is represented independently in complex networks.

Dong *et al.* [44] studying the representation problem of learning in the heterogeneous network. Learning models are represented in the form of two scalable models, metapath2vec and metapath2vec++. Metapath2vec model is used to formalize the walks that are random for constructing a heterogeneous neighborhood of a node and then the maximum advantage is taken from a skip-gram heterogeneous model, which is used for the performance of node embedding. The model metapath2vec++ is used for the modelling of semantic correlation and structure in heterogeneous networks. These two models perform better in the modern model inside heterogeneous structure task of extracting.

Haija *et al.* [45] proposes novel model attention on the transition matrix in which power series is applied, in which the random walk is guided for the optimization objective of an upstream. This method is proposed to use the parameters on the traversal data. In this, we produce the graph which is best preserved so we can see the hidden information. It is used to improve the results of the state-of-the-art by reducing the errors by 20% to 40%. The automatic learning of the parameter can vary per graph compared to the hyper parameter which is used on existing methods manually.

Tang *et al.* [46] proposed an embedding problem of large information into the vector space of low dimensional that is used in various methods like visualization, prediction of link, and classification of nodes. It also illustrates an embedding method of a network called LINE which is suitable for directed, undirected, or weighted types of network information. This method is used to preserve the structure of global and local networks. Then the new edge sampling algorithm is introduced which is used to address the limitations of descent of stochastic gradient classical and increase the effectiveness and efficiency of inference. It is a very efficient algorithm which is used in networks of the real world like language, social and citation network. The efficiency of this algorithm is that it can be embedded with billions and millions of vertices and edges on a single machine in a limited time.

Chen *et al.* [47] introduced a novel method named HARP which is used for learning embedding of low dimensional nodes graph which helps to prevent structural feature of higher-order. HARP method finds a graph that is smaller so that it can approximate the input global structure which can help in learning initial representation sets in the detailed graph. Then the graph is decomposed into levels of series and the hierarchy of the graph is embedded from coarse to the original graph. HARP improves all the state-of-the-art algorithms like DeepWalk, LINE, and Node2vec. HARP achieves a performance gain of up to 14%.

Goyal *et al.* [48] introduced the embedding task which faces the problems of scalability, dimensionality choices, and preserved features and solutions that are possible. Then three categories of approaches with algorithm representative examples were based on factorization methods, random walks, and deep learning and their performance is analyzed on tasks. The evaluation of the methods of state of art is done on a few common datasets and performance is compared which concludes some potential applications and future scopes. Graph Embedding Method is finally presented that combines the interface of the presented algorithm to facilitate research on the topic.

A neural network is introduced that operates directly on graphs which allows learning of pipelines prediction end to end whose inputs are arbitrary size and shape of graphs. The standard feature of the molecular extraction method is generalized based on circular fingerprints. It is a feature of data-driven that is more interpretable and a variety of tasks have better predictive performance [49].

Li *et al.* [50] proposed the features of the techniques of learning for the input structure of graphs. Firstly they modified the Graphical Neural Networks (GNN) together with utilizing blockaded recurrent segments along with new optimized methods also next the sequences of result are extended. A problem of graph-structured is solved with the group regarding neural network structures which favored experimental biases related entirely to the sequence-constructed architectures. The program verification has subgraphs that are described as data structures to solve the problem by achieving the performance of state-of-the-art.

The prominent result surges when the task of the entire graph is represented due to the CNN learning on graph-organized information. CNN demonstrated performances regarding state-of-the-art in a graph classification task. The discoveries of graph objects, anonymous walks are used to design the independent task algorithms for learning the representation of the graph in a distributed and explicit way. **Ivanov *et al.* [51]** representing a state-of-the-art learning graph with a new scalable unsupervised representation on the entire graph.

Cai *et al.* [52] introduced the new effective method for resolving graph systematic difficulty is known as Graph Embedding. With the help of this technique, it can preserve the graph structure information and properties of graph maximum as it is converting graph data into low dimensional space. Firstly the definition and related concepts of Graph embedding are done than 2 categories regarding graph embedding whatever relates to all that kind regarding difficulties exists and how to find their solutions. The 4 favorable investigation commands stated as to regarding computation planning, issue settings, and methods, along with an application criteria's.

Yue *et al.* [53] analyzing biomedical network by using the traditional technique which is matrix factorization which is a type of graph embedding method that has shown promising results, so there is no need for a recent graph of embedding methods according to the potential and usability to further the state of the art. The experiment is performed on 11 representative graph methods of embedding and comparison is conducted on 3 tasks of biomedical relationship projection such as drug-disease association (DDA) prediction, drug-2 interaction (DDI) projection, protein-2 interaction (PPI) projection, along with assignment regarding two node organization: medical label semantic like categorization, protein function projection. The result of these predictions is because of latest graph embedding techniques are providing great output along with the future scopes are there in biomedical graph analysis. After the analysis, the result provided is that the general guidelines are provided for determining graph-embedding techniques along with hyper-parameters are put because of dissimilar biomedical jobs.

The web is now presenting a new paradigm for the information retrieval (IR) community by generating new challenges and growing interests are also attracted from around the world. An example of the challenges is to manage a large collection of text and evaluate the hyperlinks uses which are contained within them [54].

Qiu *et al.* [55] according to them, all the models mentioned earlier with the negative sampling are being merged intent to framework of matrix factorization alongside the concluded formation. The result of the analysis revealed that DeepWalk produces a transformation rank which is very low of a normalized network of Laplacian matrix; distinct demonstration regarding DeepWalk is Line once vertices measurement is put to 1; PTE is an addition regarding Line which is seen for joint factorization regarding Laplacians various networks; node2vec is doing factorization of matrix coupled alongside issue of stationary, probable change regarding second-order of random walk. Finally, the network embedding is computed with the help of the NetMF method and approximation algorithm, which leads to the improvement on DeepWalk, LINE as long as system convention task of mining. Hence, the skip-gram-based system embedding techniques tends to a good consideration regarding learning presentation for latent system.

Koren *et al.* [56] demonstrates by the Netflix Prize competition that the nearest neighbor technique is better than the matrix factorization model for recommending the production of a product, which allows information like implicit feedback, temporal effects, and confidence levels.

The physical characteristics of White matter (WM) fiber are getting rich insights from Diffusion tensor imaging (DTI) which is used to represent the traffic pathway of the brain. It helps in investigating changes in the brain arising from pathology or aging. Graph embedding and non-negative matrix factorization are used to know the patterns of sub-network which are connecting the non-negative decomposition into the basis of the reconstruction, as well as additional basis sets, are represented on various sources in population like pathology and age. Diffusion-based connectivity is studied in subjects that are related to autism which can record the changes related to the variation in development and pathology [57].

Pan *et al.* [58] proposing an adversarial graph embedding framework that is related to the graph of data. It is encoding the structural topology and content of the node in a graph to represent a decoder that is trained to reconstruct the structure of a graph. A disruptive teaching schemes used for applying hidden representation to content earlier issue. Robust embedding is learned by using two variants of adversarial approaches are developed that are adversarial regularized graph auto encoder (ARGA) and adversarial regularized variation graph auto encoder (ARVGA). The real-world graph is validating our design and algorithms are outperforming with a wide margin in the prediction of link, clustering of a graph, and graph visualization task.

Ou *et al.* [59] proposed the scheme of maintaining asymmetric transitivity next to using large-arrangement proximity. An innovative graph-embedding method, Large Arrangement Proximity maintained Embedding, which is used as long as the preservation of large arrangement closeness regarding huge-measure graphs. It is also providing a high constrained about Root Mean Squared Error. An experiment is performed about manufactured data-set along with 3 actual-world data-sets alike an approx. a large arrangement proximities with help regrading HOPE which is good besides new technique related to reconstruction, prediction regarding link, and recommendation of the vertex.

Methods for graph embedding strive to map each vertex into a low-dimensional vector space while preserving certain structural links between the vertices in the original graph. Several recent research, such as Node2Vec, Line, Deep Walk, have suggested learning embedding based on sampled pathways from the graph. However, even if the underlying graph is undirected, their approaches barely maintain sequential togetherness, that may inadequate in several situations. Furthermore, they possess a conceptual understanding of the relationships that they maintain in their embedding space. **Zhou *et al.* [60]** present a random walk with restarting asymmetric proximity preserving (APP) graph embedding approach that captures both high-order similarities and asymmetric in-between node pairs. They show that their results inherently maintaining Rooted PageRank scoring the almost same 2 vertices in a theoretical study Researchers performed the experiments on node recommendation tasks and link prediction on online recommendation services as well as open-source datasets in the Alibaba Group, where the instructing graph above 290 million vertices along with 18 billion edges, demonstrating the effectiveness and scalability of their method.

Table 2.2 Literature for, Embedding's

Referenced Articles	**Main Topic**	**Strengths**	**Weakness**	**Future Scope**
Wongsriphisant *et al.* (2014) [41]; word2vec Explained.	Maps word to labels (Sentence)	-Discover embedding based on nodes inside *D-dimensional* time which maintains similarity. -Grasp node embedding in such as way that neighboring nodes close simultaneously inside system.	Not an advanced ways to obtain Graph embedding's. -Performing is highly-costly -Encapsulated amount above nodes offers O ($\|V\|^2$) complication!	Frame the aim for maximizing likelihood optimization difficulty, individualistic for Downstream prediction jobs.
Perozzi *et al.* (2014) [42]; Deepwalk: Online learning of social representations	Map nodes into an embedding space.			
Grover *et al.* (2016) [43]; node2vec: Scalable feature learning for networks	Trustworthiness Management,	The notion of network neighborhood $N_x(u)$of node u is flexible, resulting in rich node embedding's	-Optimization based on only 1-hop and 2-hop random walk. -Cannot on the (sub)graph use a conventional graph embedding methods	It is stay embedding The weight of the edge is not taken into account in the process , We can try to put the weight into the network training
Goyal *et al.* (2018) [48]; Graph embedding techniques, applications, and performance	Trustworthiness familiarity and similarity,			
Hongyun et *al*. (2018) [52]; A comprehensive survey of graph embedding: Problems, techniques, and applications	-A Thorough investigation of graph embedding. -divide graph embedding tasks into categories established on problem setting	Encapsulation a graph moved inside 2D space with various level of granularity	How can the variety of linked patterns seen in graphs be captured?	-Node Proximity Matrix Factorization. -Graph Laplacian Eigen maps
Jiezhong *et al*. (2018) [55]; Network embedding as matrix factorization	Mathematical basis of skip-gram established system embedding approaches, resulting in a depth study for latent network classification tasks.	Deep Walk's hidden matrix along with graph Laplacians have a theoretical link.	A better knowledge of their fundamental relationships is lacking.	Discovering just how generated graph connects towards the input graph
Koren *et al*. (2009) [56]; Matrix factorization techniques for recommender systems				

2.3 *Graph as a Matrix*

In current years, neural models in information retrieval have grown in popularity. Because of their strategic strategy in semantic matching, they provide efficient methods for product search. However, applying graph-based features in these neural techniques, despite their utility in the IR literature, is difficult. In this research, researchers use recent improvements in graph embedding techniques to enable neural retrieval models to automatically extract features from graph-structured data. **Zhang *et al.* [61]** suggested technique can indeed resolve the lengthy difficulty regarding click-through information, but it can in addition help to improve search results by incorporating external heterogeneous information. Thorough simulations upon the real-world e-commerce set of data show that their suggested technique, both even as an attributes utilized inside studying-to-rank and as an individual retrieval model framework outperforms various strong baselines.

PageRank originally devised although method as long as estimating by classifying regarding Web pages beside browser. PageRank is a crucial tool for tackling basic problems in mainly because of big data structures containing many several number of nodes, and general graphs, which based on relationships in interconnected networks. It is established on random walks while a geometric total, proposed by Brin along with Page which is represented as it. They look at a PageRank idea constructed on heat kernel that can state as an exponential sum of random walks in this study. A heat kernel fulfils a heat equation and is used to investigate a variety of aspects of random walks in a graph. Alongside concentrating about cuts defined next to sequential command regarding vertices applying heat kernel PageRank's, a local Cheeger condition proved, implying that the output division inside quadratic factor regarding optimum. Also the amount of little piece divided alongside deletion is limited up to certain goal value. As a result, the running duration of a graph-partitioning algorithm which proportional measure regarding selected amount [62].

ML over graphs are significant along with common job along applications varying medicine construct to social network relationship suggestion. A major problem inside the community for developing the mechanism for encode, represent, network structure in a method that machine-learning models can easily exploit. Earlier, machine learning algorithms features extracted encoding structural information about a graph using user-defined algorithms. There has been a surge in methods that use nonlinear dimensionality reduction and deep learning techniques to automatically train to encode network structure into low-dimensional embedding. They give a conceptual overview of significant developments inside area regarding graph presentation learning, including graph neural networks, random-walk-based algorithms, and matrix factorization-based approaches. Their strategies for approaches for embedding complete and embedding individual nodes (sub) graphs. **Hamilton *et al.* [63]** creates a unifying framework to define these current techniques, as well as a number of key applications and future research areas.

A notion that perhaps issuing regarding every layer's inputs varies throughout learning because of values of previous layers change affects Deep Neural Network training. This makes it infamously difficult to train models with saturating nonlinearities because it necessitates cautious parameter initialization and lower learning rates. Internal covariate shift is the term used to denote such phenomena, and they then solve the problem by normalizing layer inputs. The efficiency of their strategy comes from executing normalizing for every training mini-batch and including normalization within the model architecture. Batch Normalization enables others to provide considerably greater learning rates with initialization, and it can even remove the requirement for Dropout. Batch Normalization, when applied to a cutting-edge image classification model, obtains the very same performance using fourteen duration less training pace along exceeds a previous concepts for big amount. The result improved on ImageNet classification by utilizing a group regarding batch-normalized system, achieving 4.82 percent top-5 test error [64-65].

Based on an efficient variation of convolutional neural networks that operate directly on graphs, researchers describe a modular strategy especially regarding semi-supervised technique over graph-network information. The constrained 1st order estimation regarding spectral graph convolutions used to justify a choice of our convolutional design. **Kipf *et al.* [66]** proposed an algorithm grasps private layer presentations which encode all nodes attributes along with scales linearly and local network structure in total regarding graph edges. They show how their technique surpasses related methods by a large margin in a variety of trials on a knowledge graph dataset and citation networks.

Graphs found in huge variety of applications, from biology along with social research computer vision. Graphs' special capability allows them to record the structural patterns and relationships in data, allowing them to extract more insights than studying data separately. However, solving learning problems on graphs can be difficult because (1) underlying connectivity patterns for graph-organized data are often diverse and complex, (2) many forms of data, such as text and photos, aren't typically structured as graphs. On the other hand, representation learning has had a lot of success in a lot of different fields. As a result, one available alternative is to discover how to represent graphs within low-dimensional Euclidean space while maintaining graph features. Despite significant efforts to resolve the graph presentation-learning issue, several regarding all of them experience because of shallow learning methods. In ML, deep learning structures using graphs have recently shown improved performance in a variety of applications. Despite the fact that there are several different forms regarding GNN, we focus for burgeoning area regarding graph convolutional networks (GCN), it's the very popular deep learning systems. **Zhang *et al.* [67]** illustrates that the model divide already present GCN architectures into 2 classification established on various form of convolutions used along with then underline a few GCN systems in greater depth. At last, researchers divide into many GCN into categories based on their application areas. Lastly, we explore many open difficulties in this field as well as prospective research possibilities.

Consistent networks is an effective representations regarding large-arrangement tensors that have shown particularly useful in mathematics and physics applications. By employing matrix product condition to framework non-linear kernel learning systems, we show in what the techniques as long as an optimizing consistent networks may extended up to a supervised learning problems. They get less than 1% test set classification error with the MNIST data set. The extra architecture supplied because of consistent network toward learned system is discussed and interpreted [68].

Defferrard *et al.* [69] wants to extend CNNs in distinct to low-dimensional orderly matrix, wherein audio, video, image all described, in distinct to high-dimensional asymmetrical realm, which is words' embedding, brain connectome, or social networks, which are depicted by graphs. They provide a spectral graph theory formulation of CNNs, which offers the essential efficient numerical approaches and mathematical basis for designing rapid localized convolutional filters on graphs. The suggested methodology, in addition to applying to any graph structure, has the same constant learning complexity and linear computing complexity as classical CNNs. Simulations on 20NEWS and MNIST show that this unique deep learning system is capable of learning compositional, stable and local, graph features.

Graphs are used to represent information in a variety of applications. Traditional ways deal with graphical data structures by transforming the graphs into a set of flat vectors during the pre-processing step. However, essential topological information will be lost in this process, and the results may be significantly reliant on the pre-processing stage. **Gori *et al.* [70]** introduces a novel neural model, the graph neural network (GNN) that can analyses graphs directly. GNNs is an extension of recursive neural networks that can be used on a wide range of graphs, including undirected, directed, cyclic, and labelled graphs. For GNNs a learning algorithm is described, along with various experiments that evaluate the model's properties.

Learning from graph data can be used to solve a variety of significant challenges. For arbitrary graphs, we provide a method for learning convolutional neural networks. Directed, discrete, and undirected, edge properties and continuous nodes are all possible in these graphs. **Niepert *et al.* [71]** describes a wide address to withdrawing locally linked neighborhood coming from graphs which are the same as in-order-to image-based convolutional networks which work upon locally linked neighborhood regarding input. It illustrated that learned feature representations more corresponding with advanced graph kernels, even the calculation is more well organized using existing benchmark data sets.

Several machine learning tasks, ranging from video processing and image classification to natural language and speech recognition understanding, have been transformed by deep learning in recent times. Typically, the information for jobs is presented inside Euclidean time. Ever, information produced because of non-Euclidean areas is increasing being presented by graphs alongside intricate interactions along with interrelationship among items. Existing machine learning methods have been severely hampered by the complexity of graph data. Many works on expanding deep learning algorithms for graph data have recently been published. **Wu *et al.* [72]** presents a detailed overview of graph neural networks (GNNs) in the machine learning and data mining domains in this paper. Spatial-temporal, Convolutional graph, Repetitive graph, graph auto-encoders neural networks are the four categories into which we propose a new classification to separate state-of-the-art graph neural networks. They also go over the applications of graph neural networks in many disciplines, as well as the graph neural network model evaluation, benchmark data sets, open-source code. Finally, they make some suggestions for future research on this rapidly expanding topic.

Table 2.3 Literature for, Graph as a Matrix

Referenced Articles	**Main Topic**	**Strengths**	**Weakness**	**Future Scope**
Zhang *et al.* (2019) [61]; Neural IR meets graph embedding: A ranking model for product search	Not only can the suggested methodology work to alleviate the long-tail difficulty regarding click-through information, but it can also be used to include additional heterogeneous knowledge to support search queries	Determine node importance via random walk (PageRank) -Obtain node embedding's via matrix factorization -View other node embedding's (e.g. Node2Vec) as MF	Solution for the Dead ends is not discussed	-Expected transitions using power iteration of the stochastic adjacency matrix to converge final state
Chung *et al.* (2007) [62]; The heat kernel as the page rank of a graph.				
Hamilton *et al.* (2017) [63]; Representation learning on graphs	A powerful platform for exploring different GNN designs and tasks		The similarity of recalling along with link prediction jobs is highlighted,	
Jiaxuan *et al.* (2020) [64]; Design space for graph neural networks	Specific architectural designs of GNNs	A comprehensive set of guidelines	The input graph is not predetermined and evolves throughout the learning procedure	Modularized GNN implementation
Marco *et al.* (2005) [70]; A new model for learning in graph domains	Extended recursive neural networks	The model appears to be quite effective, according to preliminary experimental data.		
Niepert *et al.* (2016) [71]; Learning CNN for graphs	The learned feature representations are competitive with state of the art graph kernels	A methodology for training graph presentations, which is particularly useful when used with CNNs.	Non-linear attributes paired together to learn by CNNs and whose weights are distributed among receptive areas.	Alternative neural network structures, like RNNs, are being utilized.
Zonghan *et al.* (2020) [72]; Graph neural networks.				

2.4 *Community Detection*

A network's modularity measures how many vertices group towards community cluster in comparison to zero model network. They establish two-way modularity by using the zero model applicable regarding two-way networks. The modularity of bipartite networks represented by the modularity matrix B; several fundamental aspects of B's Eigen spectrum are utilized for explaining an algorithm to discover modules inside two-way networks. **Barber *et al.* [73]** developed the method upon a presumption that modules inside 2 segments regarding the network are interdependent, along with every portion using the vertices regarding another part to induce within modules. They demonstrate so as an algorithm favorably recognizes the modular form of two-way networks using real-world network data.

Understanding and characterizing complex systems necessitate the identification of module or community structures. While stochastic models for regeneration and modularity maximization models for discrimination are built with different goals in mind, both types of models strive for low-rank embedding to best serve as well as reconstruct network structure. However, because the mapping achieved through such embedding is real-network and linear, has a variety of nonlinear characteristics, these models are less useful in practice. **Yang *et al.* [74]** propose a unique nonlinear reconstruction approach that uses deep neural networks for representation, inspired by their tremendous representation capacity. They then incorporate pairwise constraints among network nodes to turn the approach into a semi-supervised community discovery system. The novel methods surpass most state-of-the-art techniques for community discovery, according to extensive experimental data on actual and synthetic networks.

Community detection is a research hotspot in difficult networks, which are commonly used in each element regarding mankind society. The aim function of many algorithms is modularity, which might help to simplify the algorithm. **Shang *et al.* [75]** present the community discovery technique established upon the modularity improved genetic algorithm. It employs prior information to make an algorithm further selected, increase accuracy, durability regarding community detection. It also employs modularity Q while an objective function, in this way simplifies the method. MIGA uses the genetic algorithms approach as a local explore technique, in this way it may tweak to upgrade capacity regarding local exploration. Four real-world networks and simulation findings on computer-generated demonstrate the effectiveness of MIGA when compared to futuristic techniques.

Meantime, community discovery algorithms are primarily concentrated on unweighted networks. Real-world networks, on the other hand, are constantly complicated, and unweighted networks are insufficient to capture the relationships between real-world objects. **Li *et al.* [76]** provides a deep sparse auto encoder-based community detection approach. First, the nodes' second-order neighbors are discovered, and the track load matrix considering node's 2nd-order neighbors is generated. Researchers obtain the similarity matrix by combining the track load matrix along node's weighted neighboring pathways; therefore, it reflects never barely similarity connections between linked nodes inside network topology even the similarity connections between 2nd -order neighbors and nodes. The attribute matrix that has a higher capacity to indicate the attributes regarding system may then create for developing the infinite scattered auto encoder using the unsupervised deep learning method. Finally, to generate the community structure, and for grouping low-dimensional attributes matrix for K-means algorithm is used. Their experimental findings show that the suggested approach can more reliably identify community structures than four other community detection techniques. Furthermore, parameter experiments reveal that the cluster network recognized alongside WCD technique for this study is very appropriate than cluster network obtained alongside K-means approach, it is quickly apply large-dimensional adjacency matrix.

Difficult networks are commonly employed in biological and social studies. The key to understanding complicated networks is to examine real community structure in networks. Popularly utilizes the strategies within community detection is modularity maximization. However, because of its greedy nature, it produces more communities and a huge number of improper partitions than are present. To overcome the following challenge, existing techniques use modularity as a Hamiltonian at limited temperature. Despite this, modularity not codified as a statistical model in the technique, which prevents and limits the usage of several statistical inference methods [77].

The task of community discovery in network analysis is a difficult and classic one. The similarities between the modularity maximization mode and auto encoder in respect to low-dimensional estimation as far as modularity matrix were the inspiration for this project. **Pan *et al.* [78]** proposed a novel-community-detection approach using a deep auto encoder. The modularity first sent into the auto encoder, provides a non-linear infinite presentation regarding the network. Even, node attribute connection has used the framework was subsequently and updated to create pairwise constraints on nodes to include graph regularization. Extensive practical evaluations on synthetic and real-world networks show that the proposed framework outperforms existing community discovery algorithms.

Chaudhary *et al.* [79], Blondel *et al.* [80] proposed a novel method for revealing community structure in huge complex networks. The suggested community discovery approach is based on network modularity optimization, which is a well-known concept. The method achieves this by using a cosine similarity metric based on shared links. This similarity measure aids in the efficient identification of node similarity in big networks. In comparison to previous similarity measures, it additionally takes into account the network's sparse nodes with low complexity. The approach then picks pair-wise togetherness within nodes regarding difficult networks once similarity has computed. It also identifies communities using a procedure inspired by the popular Louvain method, which effectively optimizes modularity esteem. Results conducted for demonstrating their techniques outperforms different methods, insignificantly improves experiments regarding previous techniques resulting in within-firm outcomes. Network's execution analysis regarding approaches done in community quality, modularity esteem, communities.

Table 2.4 Literature for, Community Detection

Referenced Articles	**Main Topic**	**Strengths**	**Weakness**	**Future Scope**
Barber *et al.* (2007) [73]; Modularity and community detection.	Modularity and community Detection in object based networks.	Nodes and links connected two communities can be used for disaster management	Resolution based modularity	Determining compatible and non-compatible communities
Yang *et al.* (2016) [74]; Community Detection with Deep Learning	Identifying overlapping communities as well as hubs and outliers via nonnegative matrix factorization	Identification of overlapping communities, hubs, and outliers	Performance in terms of clustering quality	Community-aware learning process to characterize community information
Shang *et al.* (2013) [75]; Community detection based on modularity and an improved genetic algorithm		Detection method based on modularity and an improved genetic algorithm (MIGA)	Consensus clustering methods.	Clustering dynamics of a network
Jin *et al.* (2020) [77]; Markov Random Field method.	Modularity optimization	Markov Random Field (MRF) method	Due to its greedy characteristic, it leads to a large number of incorrect partitions and more communities than in reality.	Introduce method outperforms the state-of-the-art methods.
Pan *et al.* (2020) [78]; Learning Deep Embedding for Community Detection	Using a deep auto encoder, a unique community detection framework has been developed	For the network, a non-linear hierarchical presentation is used.		
Laxmi *et al.* (2020) [79]; Modularity and similarity measures in SN.	Decomposing SN into dynamic user clusters “interest groups”	Ultimate ranking model (UltRank)	The clique problem.	All the benchmark approaches can also be include
Blondel *et al.* (2008) [80]; Fast unfolding of communities.	Extract the community structure of large networks.	The accuracy of our algorithm is also verified on ad hoc modular networks.	Optimal prediction	

CHAPTER 3

TRUSTWORTHINESS COMPUTATION USING GRAPH FEATURES PREDICTIONS

3.1 *Introduction and Motivation*

All the interacting entities are well expressed with the graph language. IoT objects are smart enough, as they are with small processing elements to execute the algorithm burned in processing controller chips, as well as can interact with each other like a social network. This means that rather than thinking of a given domain as a set of isolated sensor or actuator-based data points, we think of it in terms of networks and relations between these entities. This means that there is an underlying graph of relations between the entities, and these entities are related to each other [2]. According to these connections or the structure of the graph and many types of data that can naturally be represented as graphs and modeling, these graphical relations, this relational structure of the underlying domain, allows us to build much more faithful, much more accurate models of SIoT as shown in Figure 3.1.

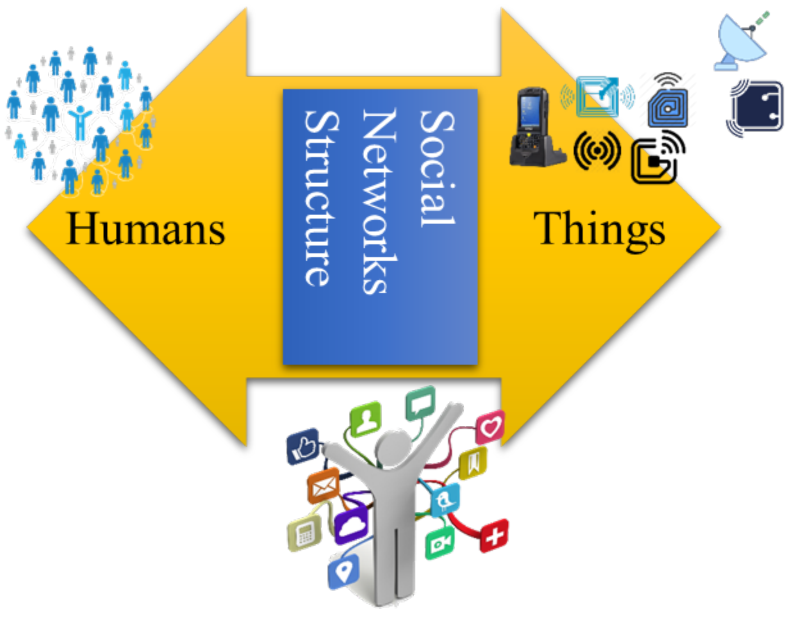


Figure 3.1: Social Interaction Between Humans and Things

In the SIoT graph machine learning, different types of tasks can be formulated on the level of individual nodes, edges, and subgraphs of nodes. The established IoT network in social format is used as input graph structure data in machine learning, deep learning, and the representation learning for output prediction as:

1. Node-level prediction.
2. Link-level prediction.
3. Graph-level prediction.

In this Chapter, three levels of tasks have been discussed as node-level prediction tasks, link-level or edge-level prediction tasks that consider pairs of nodes and tries to predict whether the pair is connected or not, and graph-level prediction, where a prediction for an entire graph can be made [13].

3.2 *Node-Level Prediction*

In a semi-supervised case, we are given a network, in there are given a couple of nodes that are labelled with different labels, so the goal is to predict the characteristics of unlabelled nodes in a way that each unlabelled node get labelled. So, the main idea of how we going to do this is by checking the nodes in a network according to their edge's adjacency with other nodes. So, for example, a node with one edge adjacency will be labelled in one group, likewise, a node with two-edge adjacency will be labelled with the second group. With the help of centralities, we do node-level prediction. Important structural features for Node-level prediction are as follows:

1. Node degree
2. Node centrality
3. Clustering coefficient
4. Graphlets

Different ways to model importance:

1. Eigenvector centrality
2. Betweenness centrality
3. Closeness centrality
4. many others…

These features are mostly used for characterization of structure and position of a node in the network as shown in Figure 3.2.

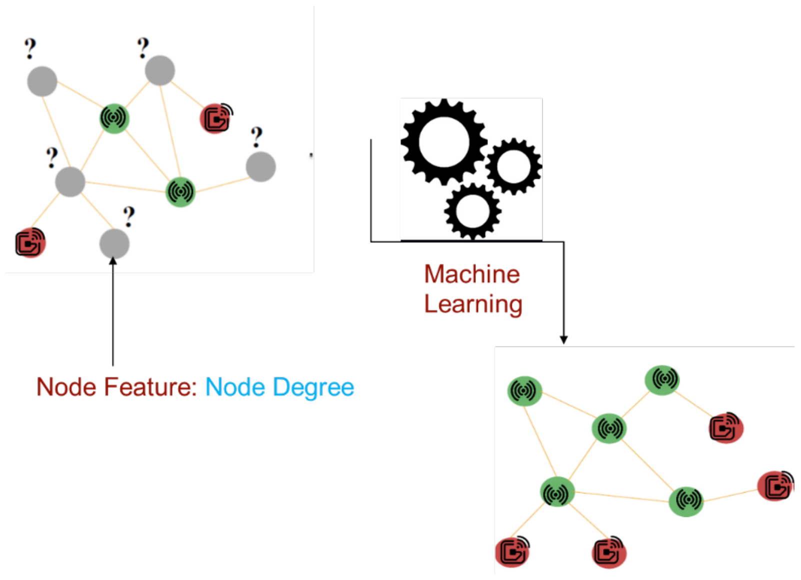


Figure 3.2: Node Degree as Structural Feature for Node Classification

3.2.1 *Node Degree:*

The number of edges linked to a node determines its degree. For counting neighbouring nodes without capturing their importance. Centrality of nodes C_V considers the relevance of nodes in a graph. We just focused on Node Centrality and Clustering Coefficient. We are predicting the nature of nodes on the basis of their degree for instance the node with single degree we provided with RF-id (Radio Frequency Identification) and node with degree two or more is provided with Wi-Fi.

3.2.2 *Node Centrality:*

The node centrality measures, try to capture or characterize this notion of how important is the node in the graph? There are many different ways how we can capture um, or model this notion of importance [27].

3.2.2.1 *Betweenness Centrality:*

It's a way of figuring out how frequently a node affects the information flow in a graph. It's even utilized to find nodes that serves as a connector between two graph components $\sigma_{st} = 12$.

$$C_B(v) = \sum_{s \neq v \neq t \epsilon V} \frac{\sigma_{st}(v)}{\sigma_{st}} \tag{3.1}$$

$\sigma_{st}(v)$ = No of shortest/geodesic paths from s to t that passes Through vertex v

σ_{st} = No of shortest/geodesic paths from s to t

1. Betweenness centrality is a measure of a vertex's ability to stay on pathways between other vertices.
2. In a network, vertices with high betweenness can have a lot of power due to their edge over other vertices conveying information.

The betweenness centrality of different nodes shown in Figure 3.23 can be calculated from Table 3.1.

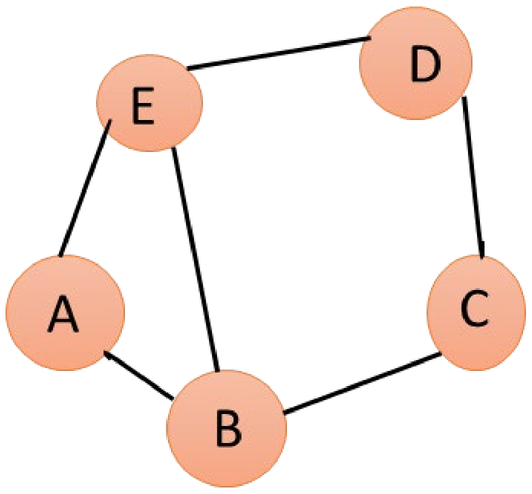


Figure 3.3: Betweenness Centrality

Table 3.1: Betweenness Centrality

$C_B(A)$=0

$C_B(B)$=0.125

$C_B(C)$=0.0417

$C_B(D)$=0.0417

$C_B(E)$=0.125

Source Node (s)	Target Node (t)	Node between Path (v)	Path
A	B		AB
A	C	B	ABC
A	D	E	AED
A	E		AE
B	C		BC
B	D	C or E	BCD/BED
B	E		BE

For the data set taken from:

Dataset Source: https://data.world/esimone/iot-platforms

Number of Nodes 88

Number of Edges 121

Graph Density: 0.032

The betweenness centrality is simulated and calculated in Figures 3.4 and 3.5

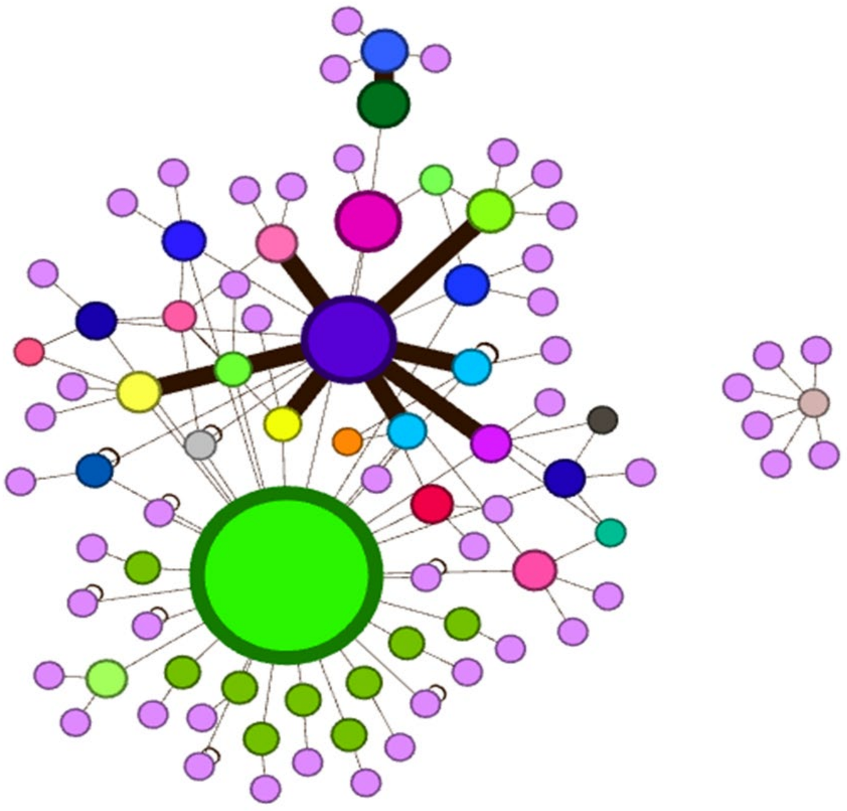


Figure 3.4: Visualization of Betweenness Centrality Distribution

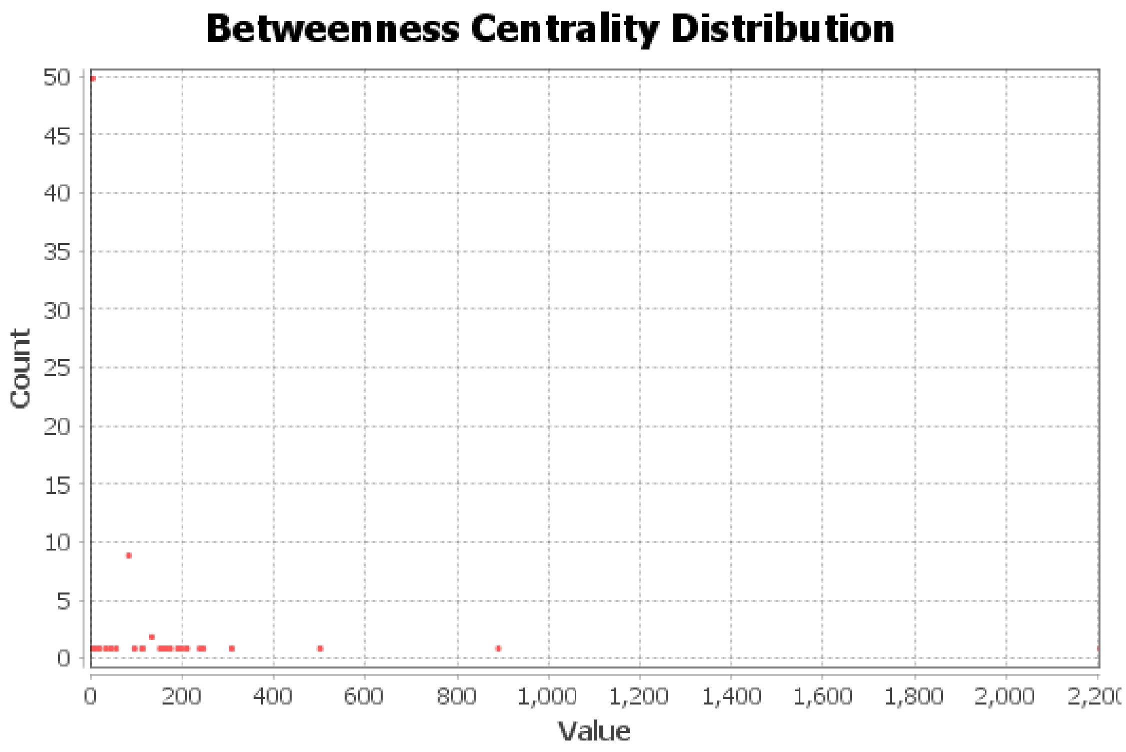


Figure 3.5: Betweenness Centrality Distribution

3.2.2.2 *Closeness Centrality:*

It's a method for locating nodes that can efficiently transfer data across a graph. Closeness centrality is used to calculate a node's average distance from all other nodes. The shortest distances exist between nodes with a high closeness score as shown in Figure 3.6.

$$C_v = \frac{1}{\sum_{u \neq v} shortest\ path\ length\ between\ u\ and\ v} \tag{3.2}$$

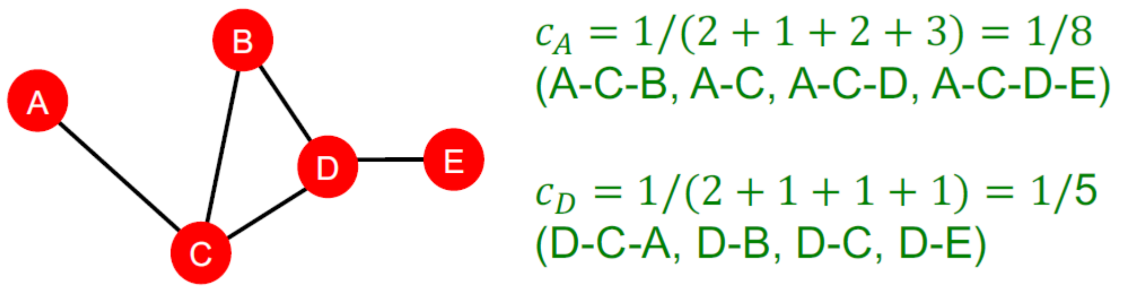

A
B
C
D
E
$c_A = 1/(2+1+2+3) = 1/8$
(A-C-B, A-C, A-C-D, A-C-D-E)
$c_D = 1/(2+1+1+1) = 1/5$
(D-C-A, D-B, D-C, D-E)

Figure 3.6: Closeness Centrality

This is used to calculate the closeness of a node from the centre of the network. The closeness centrality is simulated and calculated in Figures 3.7 and 3.8.

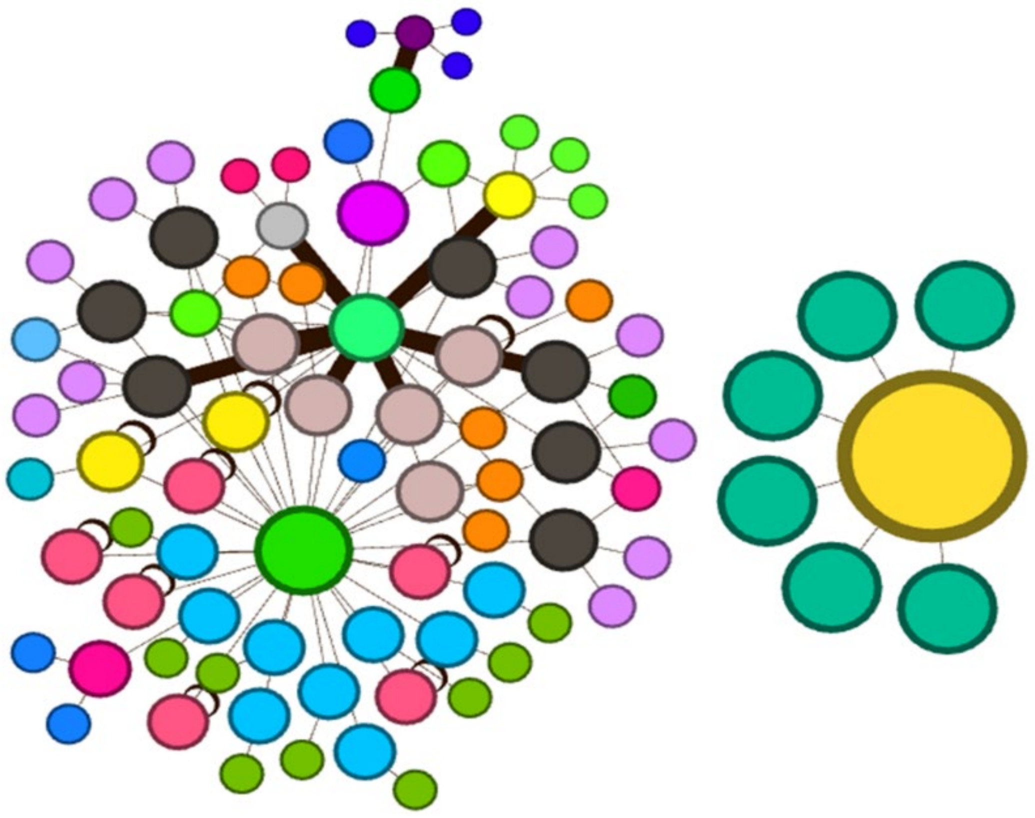


Figure 3.7: Visualization of Closeness Centrality Distribution

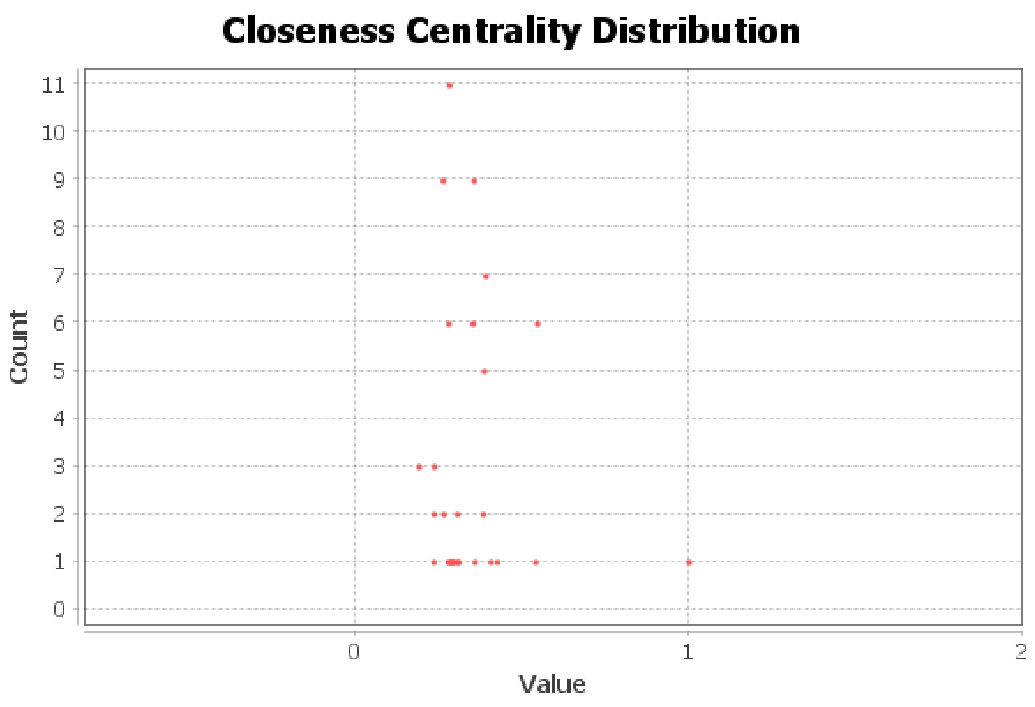


Figure 3.8: Closeness Centrality Distribution

3.2.2.3 *Eigenvector Centrality:*

Having more neighbours can't guarantee about the node trustworthiness so it's the problem in Degree Centrality. To overcome this problem, we use Eigenvector Centrality. The calculation of the centrality of a node that is linked to the majority of central nodes is eigenvector centrality. The value of the neighbours is taken into account in eigenvector centrality, which is a generalization of degree centrality (undirected). It could use outgoing or incoming edges in directed graphs as shown in Figure 3.9.

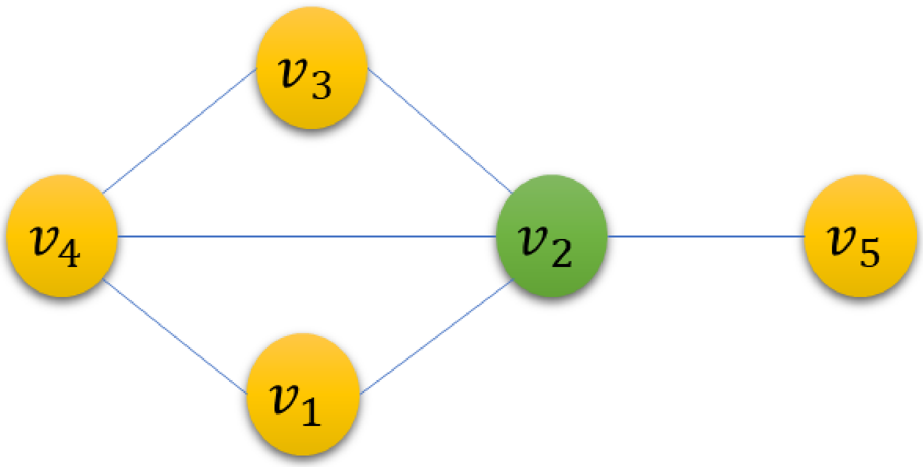


Figure 3.9: Eigenvector Centrality

$$C_e\,(v_i) = \frac{1}{\lambda_{max}} \sum_{j=1}^{n} A_{ji} C_e(v_j) \tag{3.3}$$

$$\lambda C_e = A^T C_e \tag{3.4}$$

$\lambda \boldsymbol{c} = \boldsymbol{Ac}$

• $\boldsymbol{A}$: Adjacency matrix

$A_{ji} = 1$ if $i \in N(j)$

• $\boldsymbol{c}$: Centrality vector

• λ : eigenvalue

$$\lambda = [-1.7491 \quad -1.2713 \quad 0.0000 \quad 0.3349 \quad 2.6855]$$

$$\lambda_{max} = 2.6855$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix} \quad C_e = \begin{bmatrix} 0.4119 \\ 0.5825 \\ 0.4119 \\ 0.5237 \\ 0.2169 \end{bmatrix}$$

Algorithm-3.1: Eigenvector Centrality

```
Input: adjacency matrix $A$
Output: eigenvector corresponding to $\lambda_{max}$
    1. Compute eigen values of A.
    2. Select largest eigenvalue $\lambda$.
    3. Find corresponding eigenvector $\lambda$ is $C_e$
    4. Based of PFT( Perron Forbenius theorem), $C_e$ will be positive corresponding
    5. Components $C_e$ are eigenvector centralities of the network
```

The distribution has been made on the basis of Eigenvector Centrality so you can see nodes having the higher Eigenvector Centrality compared to other nodes that's why the eigenvector centrality near to one. We used Gephi open source tool. The eigenvector centrality is simulated and calculated in Figures 3.10 and 3.11.

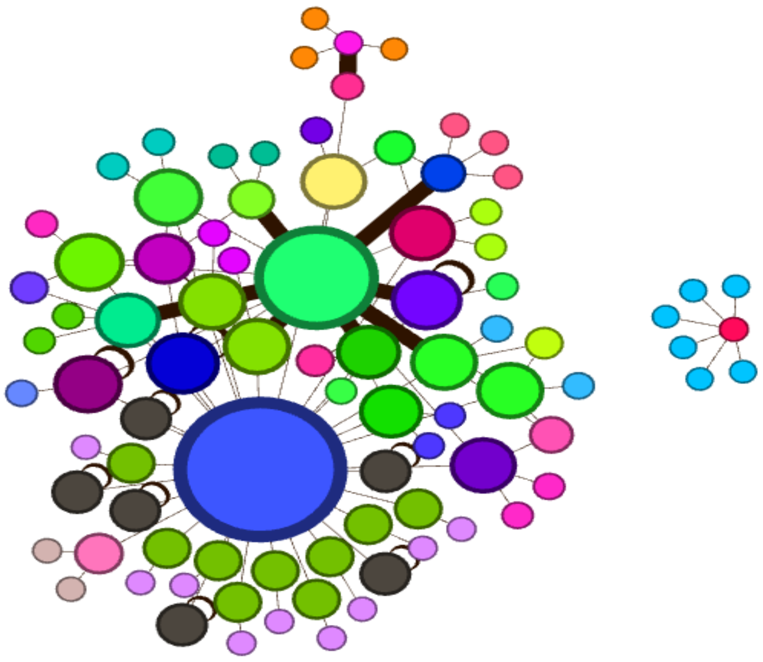


Figure 3.10: Visualization of Eigenvector Centrality Distribution

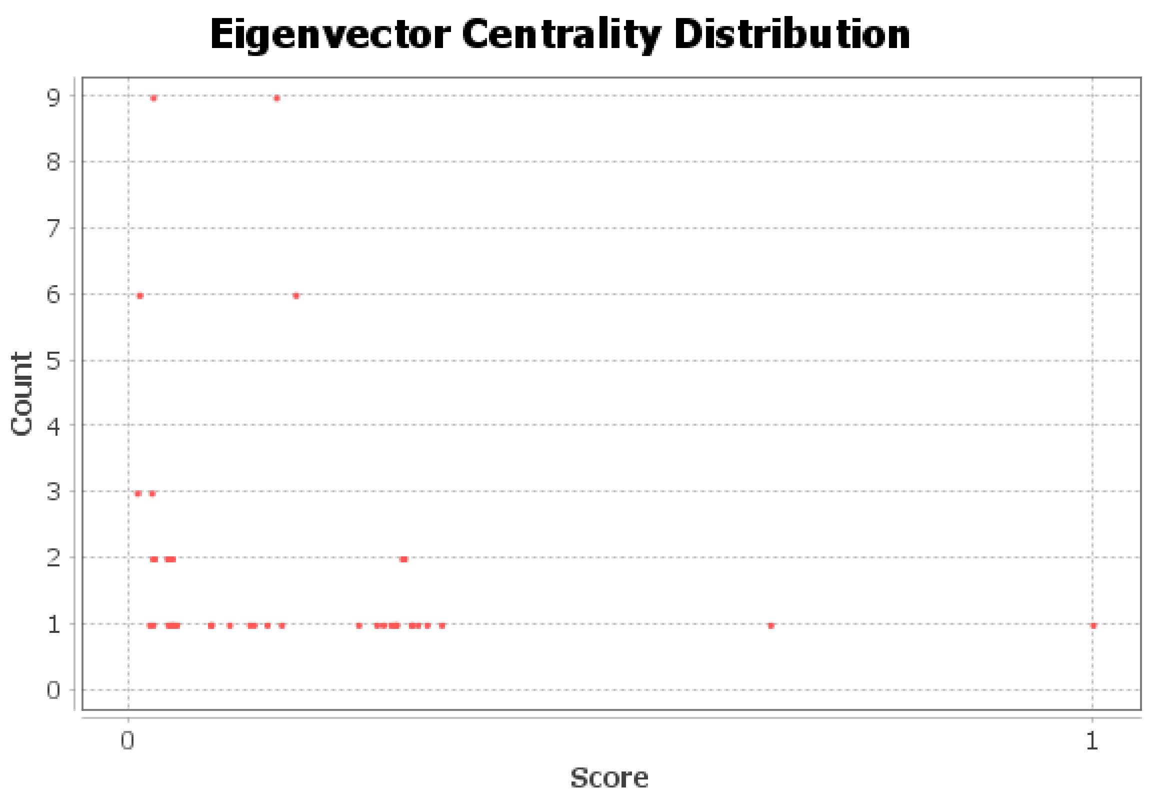


Figure 3.11: Eigenvector Centrality Distribution

3.2.2.4 *Katz Centrality:*

When dealing with directed graphs, eigenvector centrality has a significant flaw. In certain cases, such as when a node is in a directed acyclic graph, centrality only moves over outgoing edges, and in others, for when a node is in a directed acyclic graph, centrality is zero. To fix the issues, they introduce a bias term β to all node centrality values in equation 3.4 as shown in Figure 3.12.

$$C_{Katz}(v_j) = \underbrace{\boxed{\alpha \sum_{j=1}^{n} A_{ji} C_{Katz}(v_j)}}_{\text{eigenvector centrality}} + \beta \tag{3.4}$$

$\alpha \rightarrow Controlling\ term \quad \beta \rightarrow Bias\ term$

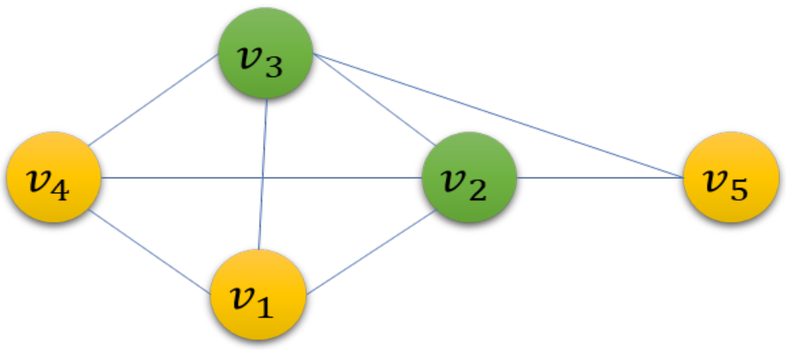


Figure 3.12: Katz Centrality

$$C_{Katz} = \alpha A^T C_{Katz} + \beta 1 \qquad \lambda = [-1.68 \quad -1.0 \quad -1.0 \quad 0.35 \quad 3.32]$$

$$C_{Katz} = \beta(I - \alpha A^T)^{-1}.1 \qquad \alpha = 0.25 < {}^{1}/_{3.32}, \ \beta = 0.2$$

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \qquad C_{Katz} = \beta(I - \alpha A^T)^{-1}.1 = \begin{bmatrix} 1.14 \\ 1.31 \\ 1.31 \\ 1.14 \\ 0.85 \end{bmatrix}$$

```
Algorithm-3.2: Katz Centrality
Input: adjacency matrix A, controlling factor α, bias term β
Output: $C_{Katz}$
    1. Compute $(I - \alpha A^T)$ [-Det$(I - \alpha A^T) \neq 0$]
       $C_{Katz} = \alpha A^T C_{Katz} + \beta 1$
    2. Find $C_{Katz}$ for $\alpha < {}^{1}/_{\lambda_{Max}}$
    3. Components $C_{Katz}$ are Katz centralities of the network
```

3.2.2.5 *Betweenness Utility Calculation:*

Based on egocentric assessments, betweenness centrality does not fully correlate to socio-centric measurements. In this network, however, seen evaluating nodes focused on 2 metrics of betweenness are equal. That shows that 2 nodes can compare respectively locally computed between values and decide which node has the greater betweenness score. Node contacts represented in symmetric adjacency matrix as shown in Figure 3.13.

$$A_{ij} = \begin{cases} 1 \ if \ there \ is \ a \ contact \ between \ i \ and \\ 0 \qquad\qquad otherwise \end{cases} \tag{3.5}$$

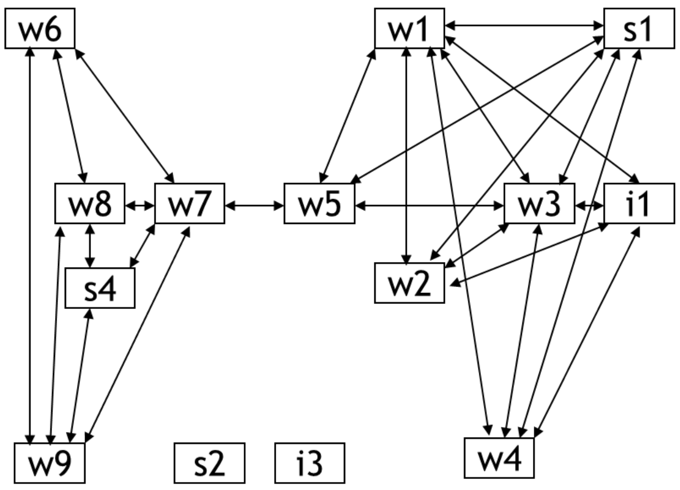


Figure 3.13: Egocentric Betweenness Similarity Utility [32]

Ego betweenness is given as the sum of the reciprocals of

$$A^2|1 - A|_{ij} \qquad (3.6)$$

$$w8 = \begin{array}{c|ccccc|} & w8 & w6 & w7 & w9 & s4 \\ w8 & \cancel{0} & 1 & 1 & 1 & 1 \\ w6 & 1 & \cancel{0} & 1 & 1 & 0 \\ w7 & \boxed{1} & 1 & \cancel{0} & 1 & 1 \\ w9 & 1 & 1 & 1 & \cancel{0} & 1 \\ s4 & 1 & 0 & 1 & 1 & \cancel{0} \end{array}$$

$$w8^2[1\text{-}w8] = \begin{array}{c|ccccc|} & w8 & w6 & w7 & w9 & s4 \\ w8 & * & * & * & * & * \\ w6 & * & * & * & * & 3 \\ w7 & * & * & * & * & * \\ w9 & * & * & * & * & * \\ s4 & * & * & * & * & * \end{array}$$

All non-zero values just above the diagonal must be examined because the matrix is symmetrical. As a result, 3 is the reciprocal of 0.33, giving the number the egocentric between values for such node. That whenever a new node is met, it transmits a list of the nodes it has come across.

3.2.2.6 *Similarity Utility Calculation:*

Indirect Node contacts learnt during a node encounter is represented in and additional matrix

$$w8 = \begin{array}{c|ccccc|c|c|} & w8 & w6 & w7 & w9 & s4 & & w5 \\ w8 & 0 & 1 & 1 & 1 & 1 & & 0 \\ w6 & 1 & 0 & 1 & 1 & 0 & & 0 \\ w7 & 1 & 1 & 0 & 1 & 1 & & 1 \\ w9 & 1 & 1 & 1 & 0 & 1 & & 0 \\ s4 & 1 & 0 & 1 & 1 & 0 & & 0 \end{array}$$

Node similarity is a simple count of common neighbours

Table 3.2: Egocentric Betweenness

Node	Egocentric Betweenness
w1	0.83
w2	0.25
w3	0.83
w4	0.83
w5	4
w6	0
w7	4.33
w8	0.33
w9	0.33
s1	0.25
s2	0
s4	0
i1	0
i2	0

3.2.2.7 *Clustering Coefficient:*

A clustering coefficient is calculated of how closely nodes in a network cluster together in graph theory.

$$CC(v) = \frac{2N_v}{k_v(k_v-1)} \tag{3.7}$$

$$CC(v) = \underbrace{Average}_{v\epsilon G}\ CC(v)$$

v: *is a node*

k_v: *It's degree*

N_v: *number of links between neighbors of v*

Fractions of possible outcomes

$$0 \leq CC(v) \leq 1$$

Star Clique: Network Density=1

$$\text{Network Density}=\frac{Connected\ links}{No\ of\ possible\ links} = \frac{m}{\frac{n(n-1)}{2}} \tag{3.8}$$

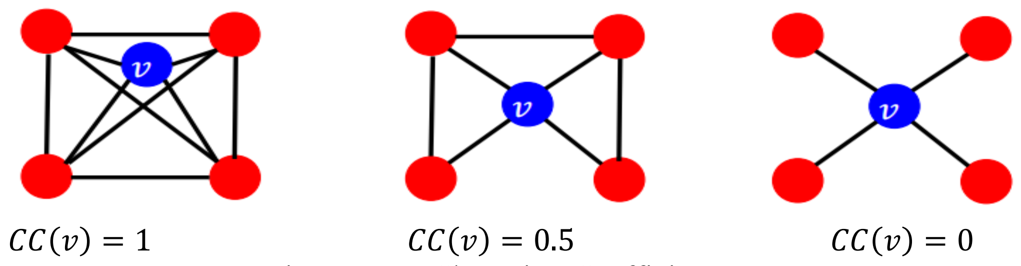


Figure 3.14. Clustering Coefficient

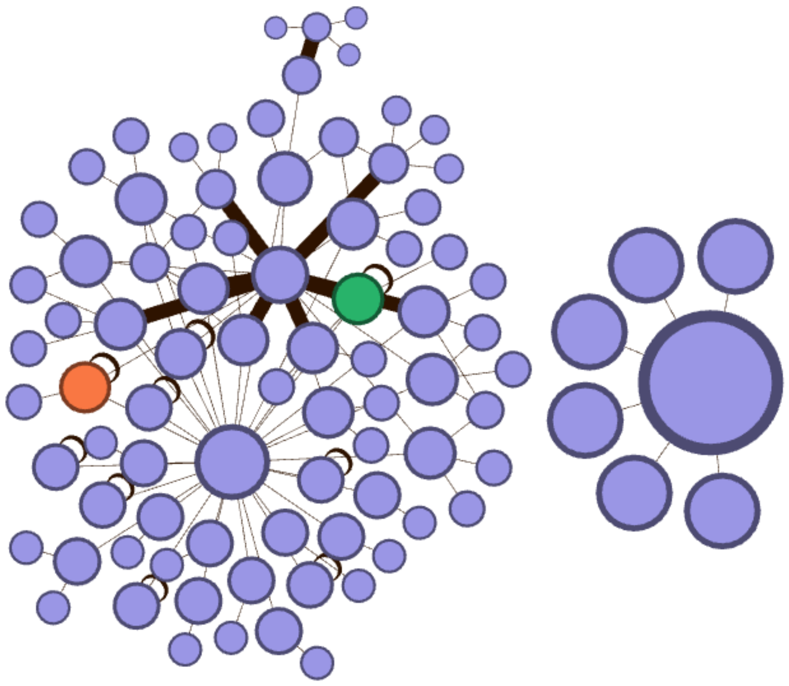


Figure 3.15: Visualization of Clustering Coefficient Distribution

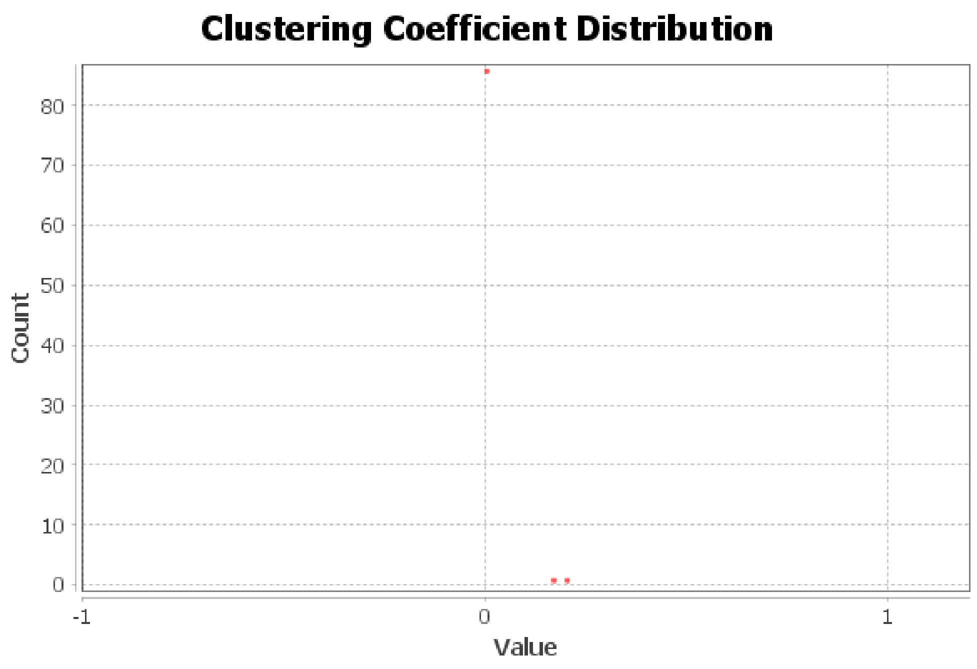


Figure 3.16: Clustering Coefficient Distribution

3.3 *Link-Level Prediction*

Methods indicate a connection between the 2 nodes that haven't yet been coupled. Over time, networks that are ideal for dynamic networks emerge, and some of the existing interconnections fade away.

A relationship establishment between two complex network communities depends upon:

1. Distance-based feature
2. Local neighbourhood overlap
3. Global neighbourhood overlap

These features can be used to predicting the existence of a link between two complex communities. The graph depicted a missing value adjacency matrix. The goal is to fill in the missing values in the matrix [33].

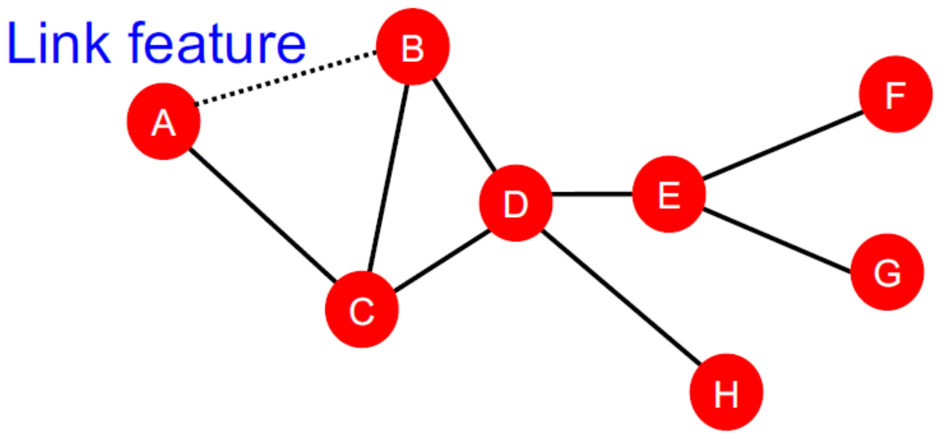


Figure 3.17: Link Prediction

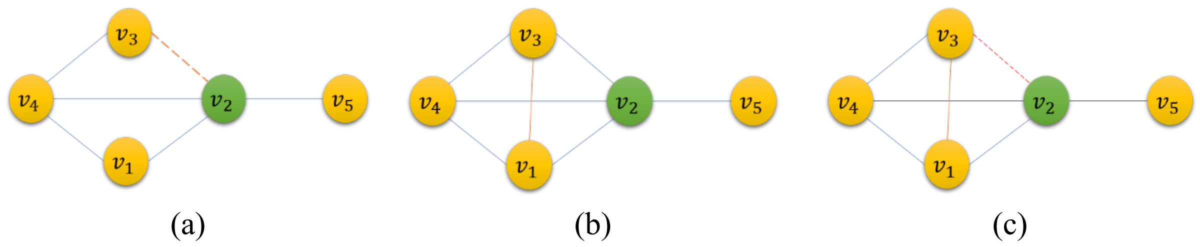


Figure 3.18: Link Level Task: (a) Removing Links (b) Adding Links (c) Adding and Removing Links

3.3.1 *Distance-Based Feature:*

It takes the shortest link in between 2 nodes but ignores how the neighbourhood overlapping. To overcome this problem, we use neighbourhood overlap which is further categorized into two different parts which is local and global neighbourhood overlap.

3.3.2 *Neighbourhood-Based Link Prediction*:

1. Define $N(i)$ as a set of neighbor nodes.
2. Only those link prediction systems are built to calculate a score for any 2 nodes that have not yet been joined.
3. When any latest connection is established, the node pair with the highest positive link prediction score is picked to be linked.

3.3.2.1 *Local Neighbourhood Overlap:*

how many neighbours do you have in common? What is the number of common nodes between a pair of nodes? And this all this captured by the notion of local neighbourhood overlap, which captures the number of neighbouring nodes shared between two nodes v_a and v_b. There are three ways which are as follows:

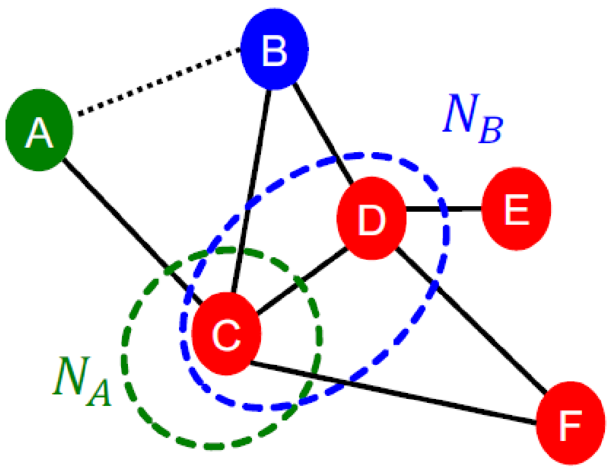


Figure 3.19: Local Neighbourhood Overlap

3.3.2.1.1 *Common Neighbours:*

Neighbouring nodes shared between two nodes v_a and v_b.

$$|N(v_1) \cap N(v_2)| \tag{3.9}$$

$$|N(A) \cap N(B)| = |\{C\}| = 1$$

3.3.2.1.2 *Jaccard's Coefficient:*

It examines two sets of nodes to see which ones are common and which are unique. It is a percentage measure of how near two sets of data are to each other, ranging from 0% to 100%. The closer the two groups are, the higher the proportion.

$$\frac{|N(v_1) \cap N(v_2)|}{|N(v_1) \cup N(v_2)|} \tag{3.10}$$

$$\frac{|N(A) \cap N(B)|}{|N(A) \cup N(B)|} = \frac{(\{C\})}{(\{C, D\})} = \frac{1}{2}$$

3.3.2.1.3 *Adamic-Adar Coefficient:*

It was developed by Lada Adamic and Eytan Adar in 2003 to estimate social network links based on the number of mutual links between two nodes. It's calculated by adding the inverse logarithmic degree centrality of the node's mutual neighbours.

$$\sum_{u \in N(v_1) \cap N(v_2)} \frac{1}{log(k_u)} \tag{3.11}$$

$$\frac{1}{log(k_u)} = \frac{1}{log 4}$$

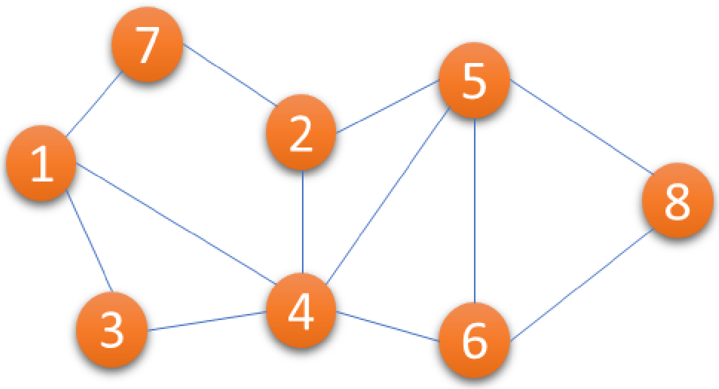


Figure 3.20: Local Neighbourhood Overlap for Adamic-Adar index and Jaccard's coefficient

Table 3.3: Degree of Neighbour Nodes

Nodes	Neighbours	log(Degree)	1/log(Degree)
1	{3,4,7}	0.4771	2.0959
2	{4,5,7}	0.4771	2.0959
3	{1,4}	0.301	3.3222
4	{1,2,3,5,6}	0.6989	1.4308
5	{2,4,6,8}	0.4771	2.0959
6	{4,5,8}	0.4771	2.0959
7	{1,2}	0.301	3.3222
8	{5,6}	0.301	3.3222

Table 3.4: Jaccard's coefficient of Each Node

Nodes Pairs (I,j) (not connected)	**Intersection of Neighbor Set**	**Union of Neighbor Sets**	**JAC(I,j)**
1,2	{4,7}	{3,4,5,7}	2/4=0.50
1,5	{4}	{2,3,4,6,7,8}	1/6=0.16
1,6	{4}	{3,4,5,7,8}	1/5=0.2
1,8	{}	{3,4,5,6,7}	0/5=0
2,3	{4}	{1,4,5,7}	1/4=0.25
2,6	{4,5}	{4,5,7,8}	2/4=0.5
2,8	{5}	{4,5,6,7}	1/4=0.25
3,5	{}	{1,2,4,6,8}	0/5=0
3,6	{4}	{1,4,5,8}	1/4=0.25
3,7	{1}	{1,2,4}	1/3=0.33
3,8	{}	{1,4,5,6}	0/4=0
4,7	{1,2}	{1,2,3,5,6}	2/5=0.40
4,8	{5,6}	{1,2,3,5,6}	2/5=0.40
5,7	{2}	{1,2,6,8}	1/4=0.25
6,7	{}	{1,2,4,5,8}	0/5=0

Table 3.5: Adamic-Adar index of Each Node

Nodes Pairs (I,j) (not connected)	**Common Neighbours**	**AAI(i,j)**
1,2	{4,7}	4.753
1,5	{4}	1.4308
1,6	{4}	1.4308
1,8	{}	0
2,3	{4}	1.4308
2,6	{4,5}	3.5267
2,8	{5}	2.0959
3,5	{}	0
3,6	{4}	1.4308
3,7	{1}	2.0959
3,8	{}	0
4,7	{1,2}	4.118
4,8	{5,6}	4.118
5,7	{2}	2.0959
6,7	{}	0

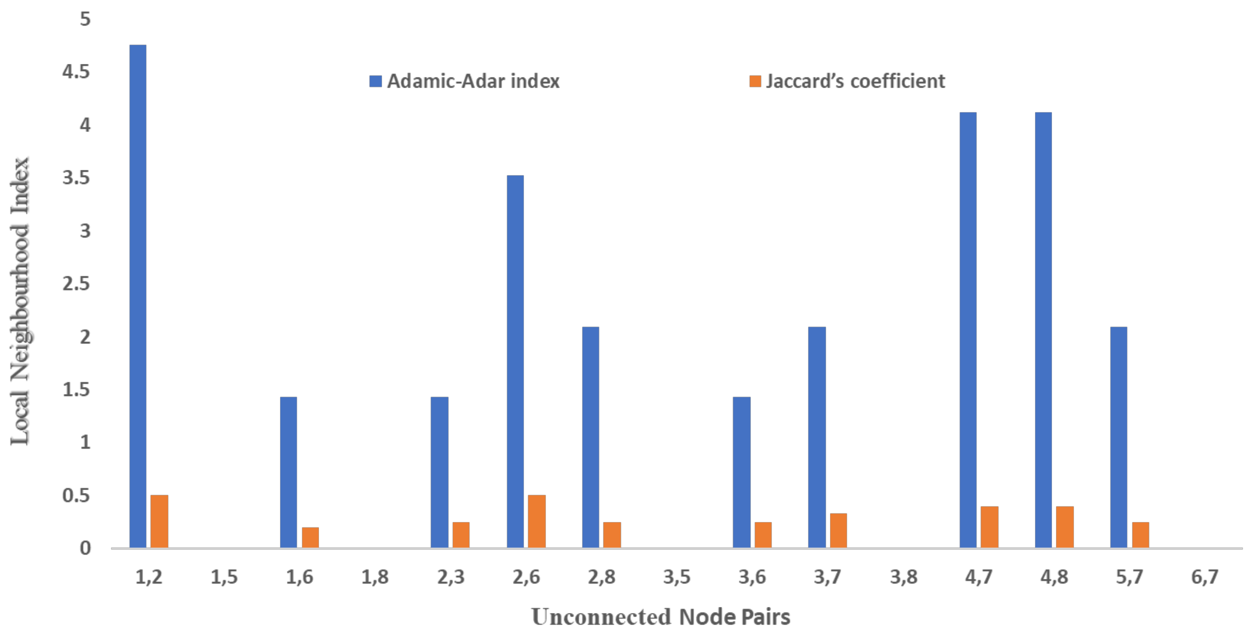


Figure 3.21: Node Prediction Using Local Neighbourhood Overlap for Adamic-Adar index and Jaccard's coefficient

Limitation Of local neighbourhood overlap:

- If 2 nodes haven't any neighbors in association, then the metric is indeed 0.

$$N(v_{\mathrm{a}}) \cap N(v_{\mathrm{b}}) = \emptyset$$

$$|N(v_{\mathrm{a}}) \cap N(v_{\mathrm{b}})| = 0$$

- Nevertheless, 2 nodes might be interconnected in the future.

3.3.2.2 *Global Neighbourhood Overlap:*

It overcomes the constraint by taking into account the full graph. Using global graph structure to score two nodes. The matrix we are going to address is katz index. Katz index counts the number of all paths between a pair of nodes with differing lengths. So, for figuring out the two things which are firstly, how do you compute number of paths of a given length between two nodes? This can actually be very elegantly computed by using power of the graph adjacency matric. Let G is simple graph V(G)= {1,2,3,4,5} shown below.

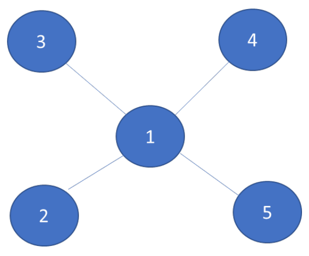


Figure 3.22: Global Neighbourhood Overlap

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} \text{No of walk=2, No of path=1}$$

Each pair vertices G find paths of length 2 between them

Definition: path alternating sequence of vertices and edges of form $v_1\, e_1\, v_2\, e_2$ ------------ $v_{k1}\, e_{k-1}$ where e_i edge joining v_i to v_{i+1} , all vertices and all edges are distinct.

$$A^3 = \begin{bmatrix} 0 & 4 & 4 & 4 & 4 \\ 4 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 \end{bmatrix} \text{No of walks=3,}$$

No path possible in this as according to the definition we can’t use the same edges or vertices twice.

3.4 *Graph-Level Prediction:*

At the graph level, Graph-Neural-Network that forecasts a single value for an entire graph referred to as this. Mostly used for categorizing complete graphs or calculating graph similarities. The focus of this study is on node-level outputs.

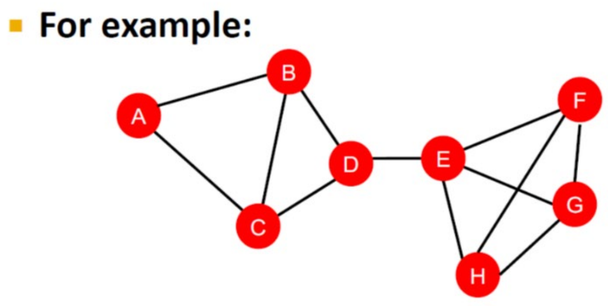


Figure 3.23: Graph Level Features

3.4.1 *Kernel Methods:*

Pattern analysis algorithms are a collection of several sorts of algorithms. The linear classifier is used to address non-linear problems. In SVM, a common kernel is utilized to offer a window through which data can be modified. Polynomial kernels, Gaussian kernels, and so on are examples of its applications. kernel between graphs G, and G', returns a real value, and measures a similarity between these two graphs, or in general, measure similarity between different data points. Kernel matrix is then a matrix where simply measures the similarity between all pairs of data points, or all pairs of graphs. this kernel matrix, uh, has to be positive semi-definite, which means it has to have positive eigenvalues [39].

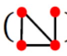
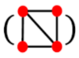
$\emptyset(.)$ is feature presentation for: $K(GG')$=$\emptyset(G)^T\emptyset(G')$

Graph Kernels: Measure similarity between two graphs:
Weisfeiler-Lehman Kernel and Graphlet Kernel both employ a packet of graph representation. But there are some **problem in Graphlet Kernel:**

- If G and G' are different shapes and sizes, the result will be substantially affected.
- Because the subgraph dynamic capabilities test is NP-hard, it is inevitable under the difficult scenario.
- If you count size k graphlets for a graph of size n, you'll get n^k.
- If the graph has a degree of d, then the order of degree will be $o(nd^{k-1})$.

3.4.2 *Weisfeiler-Lehman Kernel:*

It evaluates the correlation between 2 graphs simultaneously an inner product regarding their histogram vectors after performing multiple rounds of the Weisfeiler-Lehman algorithm. Kernel gathers total times a color appears within graph in each redundancy in certain histogram vectors. 2 feature vectors are same, kernel returns a maximum correlation for two isomorphic graphs.

$\phi($  $) = \phi($  $)$

Algorithm-3.3: Weisfeiler-Lehman Kernel

```
Given: A set of nodes V in a graph G.
Step1:  Assign each node v with initial color $c^{(v)}$.
Step2:  Use Iteration for node color refinement:
            $c^{(\kappa+1)}(v) = HASH\left(\left\{c^{(\kappa)}(v), \left\{c^{(\kappa)}(u)\right\}_{u\varepsilon N(v)}\right\}\right)$
        where HASH denotes two inputs to diverse color
Step3:  Aafter K steps of color enhancement, $c^{(\kappa)}(v)$ recapitulates the assembly of K-hop neighbourhood
```

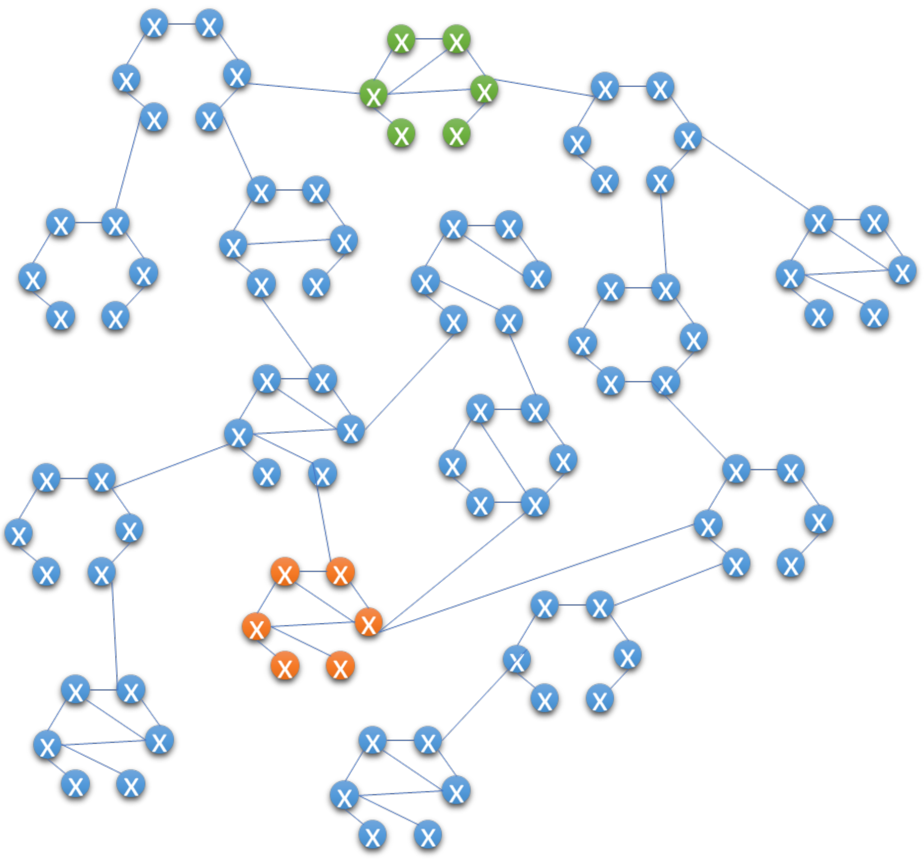


Figure 3.24: Connected Subgraph Network

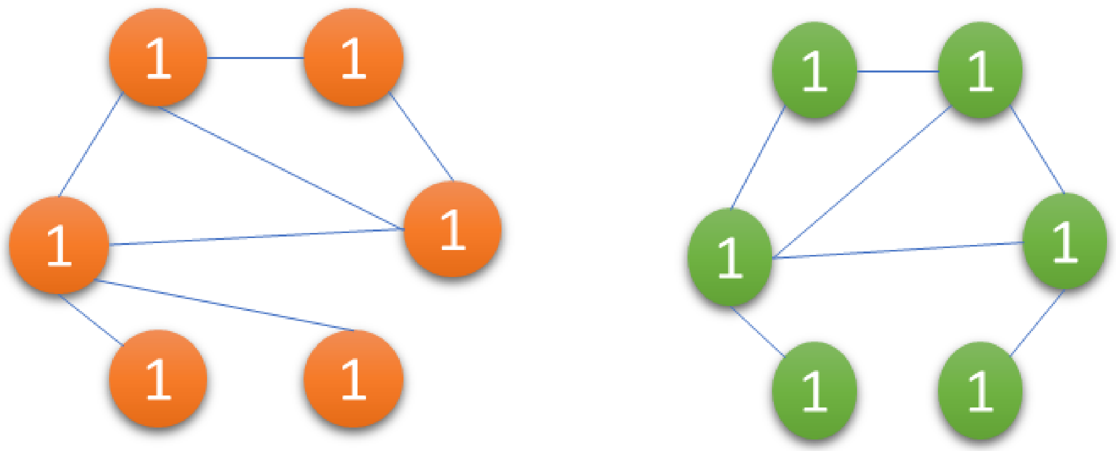


Figure 3.25: Color Refinement given Two Graphs

Step1:

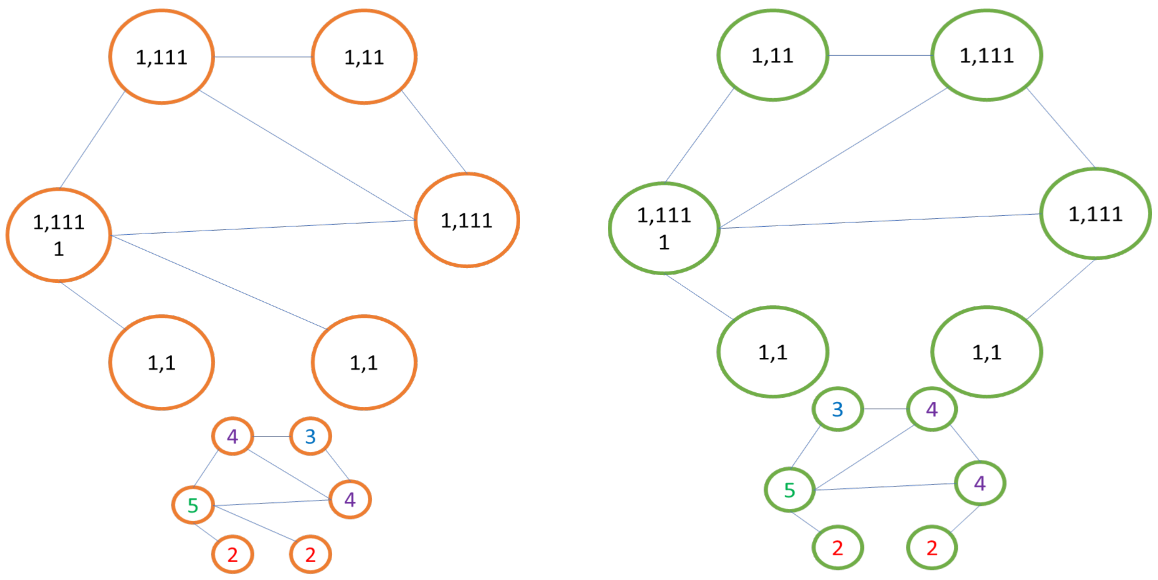


Figure 3.26: Aggregated Colors and Hash Aggregated Colors Step-1

Table 3.6: Hash Table for First Iteration

Hash table	
1,1	2
1,11	3
1,111	4
1,1111	5

Step2:

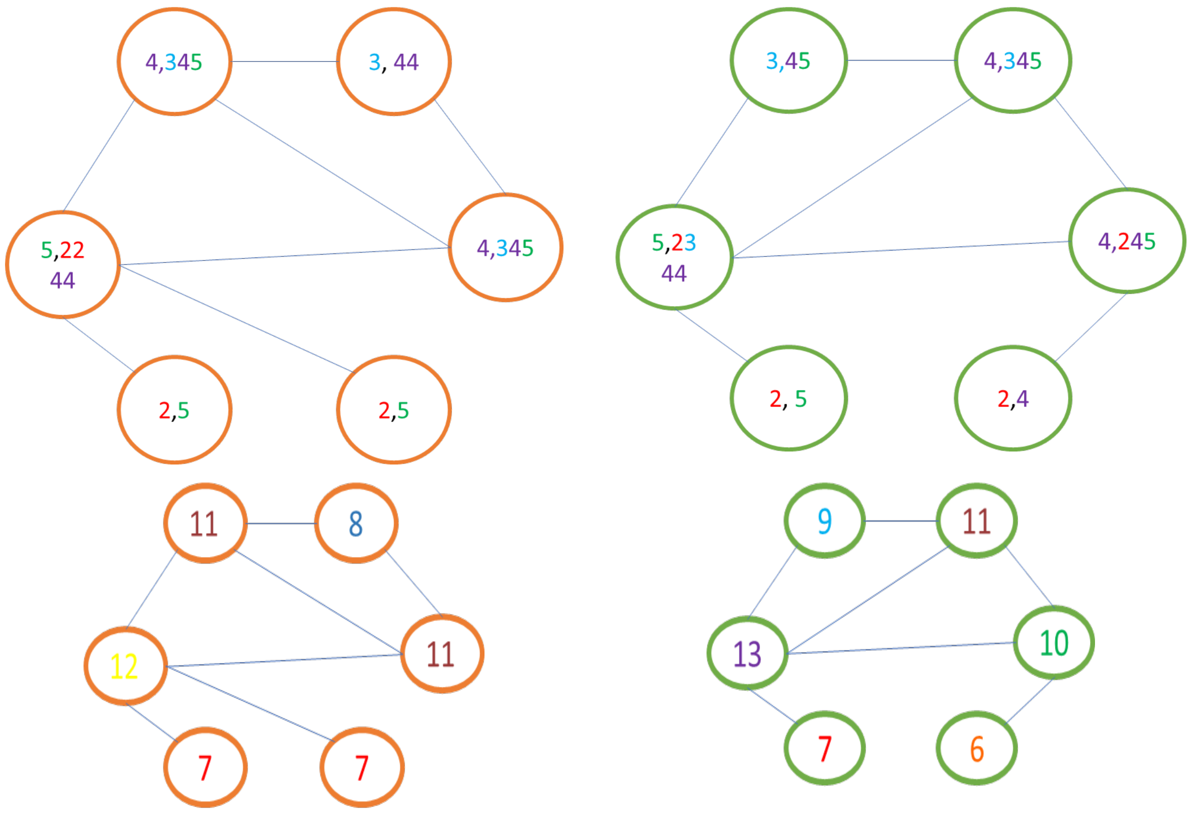


Figure 3.27: Aggregated Colors and Hash Aggregated Colors Step-2

Table 3.7: Hash Table for Second Iteration

Hash table	
2,4	6
2, 5	7
3, 44	8
3,45	9
4,245	10
4,345	11
5,2244	12
5,2344	13

Table 3.8: Combined Hash Table

Hash table	
1,1	2
1,11	3
1,111	4
1,1111	5
Hash table	
2,4	6
2, 5	7
3, 44	8
3,45	9
4,245	10
4,345	11
5,2244	12
5,2344	13

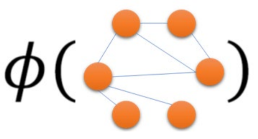
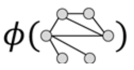


Table3.9: Kernel Counts Number for 

Colors	1	2	3	4	5	6	7	8	9	10	11	12	13
Counts	6	2	1	2	1	0	2	1	0	0	2	1	0

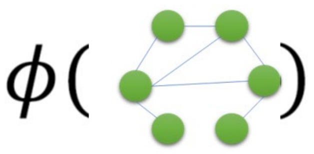
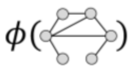
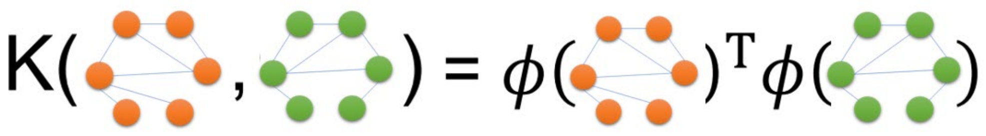


Table3.10: Kernel Counts Number for 

Colors	1	2	3	4	5	6	7	8	9	10	11	12	13
Counts	6	2	1	2	1	1	1	0	1	1	1	0	1

The WL kernel value is computed by the inner product of the color count vectors:



=6*6+2*2+1*1+2*2+1*1+0*1+2*1+1*0+0*1+0*1+2*1+1*0+0*1

=36+4+1+4+1+0+2+0+0+0+2+0+0=**50**

3.4.3 *Trustworthiness Measure using Stable Matching*

The Gale Shapely Algorithm namely the result to stable matching difficulty and we must understand the problem to fully understand the solution. In the stable matching problem we want to make a set of input together utilizing two sets regarding nodes of different subgraphs. With every set presenting within every pair, each Graphlet can once be together as well as such solution would be straight 1-to-1 grouping inside 2 sets regarding Graphlets. Individual tastes are what add to the problem's complexity. Every Graphlet has a preference for who they would want to interface with from the other set. The objective is to produce a set of matchings in which no two Graphlets interface with each other than their present interfaces. To aid understanding of the challenge and the procedure that delivers a solution to it here is an example we have two Graphlets. The first is M-Graphlet which is M-1, M-2, M-3, M-4, M-5 and another N-Graphlet which is N-1, N-2, N-3, N-4, and N-5. We need to pair all of them in a purely single interface and due to the constraints of the problem. As, we can see each Graphlet has a choice list next to their name containing all Graphlet from the other set in order of preference. So, for instance M-1 very popular to be interfaced along with N-4 go after by N-2 and likewise. It is influential that this ranking is complete so that each Graphlet has a specified preference for each Graphlet in the opposite set. Note that it is not necessary for preferences to be reciprocated. For example while M-5 top choice is N-4 and whereas M-5 is the least favourable match from N-4 perspective. The Gale Shapely Algorithm itself fairly simple to begin it is expected that each Graphlet has a strict ranking members of the opposite group. One of the two sets is chosen to make what we will call proposals while choosing either set will produce a stable matching. Next part of the algorithm is a looping section. One Graphlet from the opposite group in our case Graphlet M who is not interfaced will propose to its most preferable choice who has not yet rejected it. The Graphlet N that has been proposed must decide whether to accept or reject the proposal. If Graphlet N really hasn't got an offer earlier, Graphlet N will accept it. If Graphlet N has indeed received an offer, but the initial offer is still preferred, Graphlet N will reject it. If Graphlet N had previously received an offer, but the current offer is more favourable, Graphlet N will either guilt or reject the prior offer and accept the new one. It's worth noting that the sequence in which the Graphlet M make their proposals is irrelevant. Because the stable matching will be the same, this phase will loop until all of the Graphlets in the proposed group are interfaced, and therefore all of the Graphlets in the other group. The pairs have now been finished, and the Graphlets in the pairs have been interfaced.

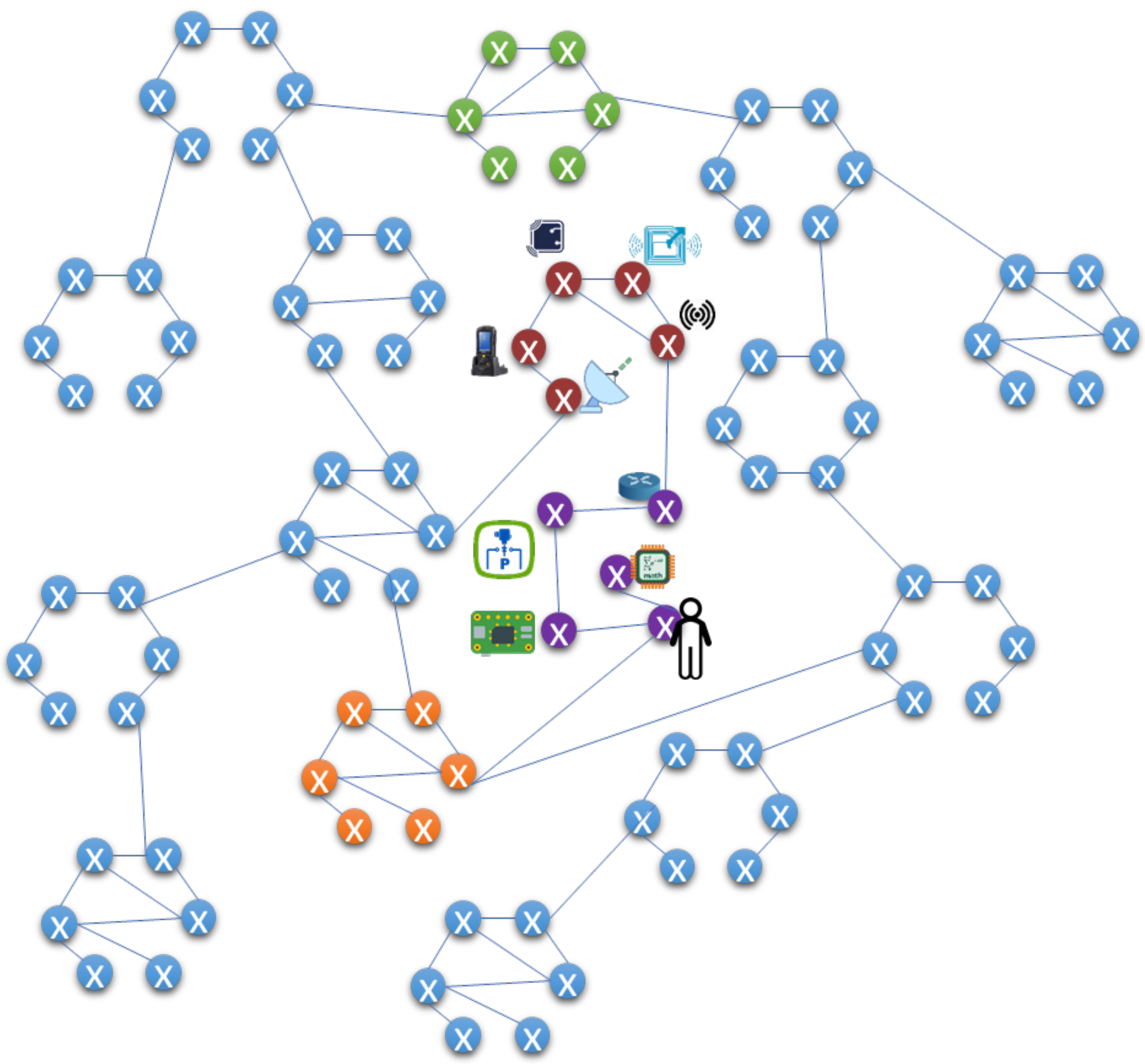
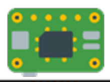
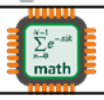


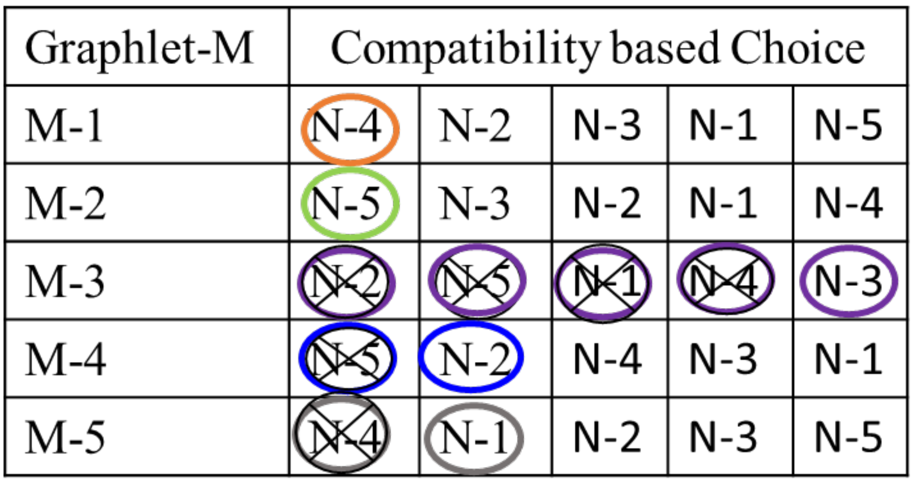
Figure 3.28: Stable Matching Between Two Nodes of Different Subgraphs

Table 3.11: Graphlet Node Details

Graphlet-M		Graphlet-N	
M-1		N-1	
M-2		N-2	
M-3		N-3	
M-4		N-4	
M-5		N-5	

Here is an example of the algorithm working in practice. Firstly M-1 proposes to its most preferable option N-4 as N-4 has no better options for the time being N-4 will accepts and M-1 and N-4 are interfaced. Next M-2 proposes its most preferable option N-5 and as N-5 has not received any better offers so N-5 also accepts. M-3 proposes N-2 and N-2 also accepts due to lack of any other offer. M-4 then proposes to N-5 however N-5 has already received a better offer from M-2 and so, N-5 rejects M-4. Finally, M-5 proposes to N-4 but as N-4 has already received a better offer from M-1 so it reject it too. At this point M-1, M-2 and M-3 are already interfaced. So, do not attempt any more proposals. M-4 still has no one to interface with and so proposes to its next most preferable choice who has not rejected it. M-4 prefers N-2 to its previous offer from M-3 and so N-2 rejects M-3 and is now interface with M-4. M-5 proposes N-1 and so it has not previously received an offer so it accepts. M-3 is now the only one who has yet to find a pair after being rejected by N-2. Then M-3 proposes to N-5 got rejected then again M-3 proposes to N-1 got rejected again then again M-3 proposes to N-4 got rejected again as they all have better options then at last it finally proposes to N-3 who accepts as it has not received any offers. Thus, all of Graphlet M are now interfaced and as well as there is no need for any more proposals the current pairings are interfaced with the final pairs being M-1, N-4; M-2, N-5; M-3, N-3; M-4, N-2; M-5, N-1. All Graphlets have been successfully interfaced and while certain Graphlet do not get that preferred choices. The matching is stable, we know this because if a Graphlet M preferred Graphlet N to his current match it would already have proposed to Graphlet N. If we do proposal from Graphlet N choice list then the result of the interfaced best choice will be different not same. The complexity of the algorithm is $O(N^2)$.

Table 3.12: Graphlet M Choice Table



Graphlet-M	Compatibility based Choice				
M-1	N-4	N-2	N-3	N-1	N-5
M-2	N-5	N-3	N-2	N-1	N-4
M-3	~~N-2~~	~~N-5~~	~~N-1~~	~~N-4~~	N-3
M-4	~~N-5~~	N-2	N-4	N-3	N-1
M-5	~~N-4~~	N-1	N-2	N-3	N-5

Table 3.13: Graphlet N Choice Table

Graphlet-N	Compatibility based Choice				
N-1	M-4	M-2	[✓ M-5]	~~M-3~~	M-1
N-2	M-2	M-1	[✓ M-4]	~~M-3~~	M-5
N-3	M-1	[✓ M-3]	M-5	M-4	M-2
N-4	M-4	[✓ M-1]	~~M-3~~	M-2	~~M-5~~
N-5	[✓ M-2]	M-5	M-1	~~M-3~~	~~M-4~~

Best interface pairs are: M-1,N-4 M-2,N-5 M-3,N-3 M-4,N-2 M-5,N-1

Algorithm-4: Gale-Shapley Algorithm for Stable Matching

```
Initialize  $\forall M_i \in$ Graphlet-M and $\forall N_j \in$ Graphlet-N to free, $S = \emptyset$
While $\exists M_i \in$ Graphlet-M who does not have a Interface do
        Let $N_j$ be the highest-ranking node in Graphlet-N subgraph in $M_i$ Choice list, to whom $M_i$ has not yet interfaced
        Now $M_i$ interfaced to $N_j$
  If $N_i$ is free then
        $(M_i, N_j)$ has best compatible interface (add $(M_i, N_j)$ to S)
  else
        $N_j$ already interfaced to $M_{i\pm q}$
If $N_j$ interfaced to $M_{i\pm q}$ instead of $M_i$ then
        $M_i$ remains free for interface
else
        $N_j$ interfaced to $M_i$ instead of $M_{i\pm q}$
        $M_{i\pm q}$ remains free for interface (remove $(M_{i\pm q}, N_j)$ from S)
        $(M_i, N_j)$ has interfaced (add $(M_i, N_j)$ to S)
end
Return the set S of interfaced nodes pairs
```

3.5 *Results and Discussions*

The graph feature prediction at the level of node, link and graph itself address the various entities like node degree, node centrality, clustering coefficient and local and global neighborhood overlap are the major component for feature engineering used as graph input for any machine leaning algorithm. The use of stable matching algorithm with Weisfeiler-Lehman Kernel gives the improved compatibility of two subgraphs.

CHAPTER 4

NODES AND GRAPHS EMBEDDING FOR DOWNSTREAM PREDICTIONS

4.1 *Introduction and Motivation:*

NODE EMBEDDING: The purpose of embedding is to encode nodes in such a way that similarity in the dot product embedding space approximates similarity in the original network. These techniques are divided into three stages:

1. Nodes to embedding mapping.
2. It measures the similarity of nodes in the original network, which describes how the vector space relationships transfer to the original network relationships.
3. Optimize the encoder's parameters so that node similarity in the network approaches the dot product of node embedding.

GRAPH EMBEDDING: In some cases, we want to incorporate a whole graph GG. Graph embedding can accomplish in a variety of ways:

1. The basic idea is to apply a typical graph embedding technique to the sub-graph, and then add the node embedding in the sub-graph.
2. Using a conventional graph embedding technique, create a "virtual node" to characterize the sub-graph.
3. Anonymous walk embedding can be used. We can enumerate all potential anonymous walk steps and count them; they characterize to understand graph embedding, use the graph as probability distribution across so many paths.

Using traditional learning when the graph is used as input, only some node-link or graph-level features that describe the topological structure of the network, either around the node or around a particular link or the entire graph, can be extracted. Then, the extracted topological information can be used to compare and combine it with the attribute-based information to then train a classical machine-learning model like support vector machine or logistic

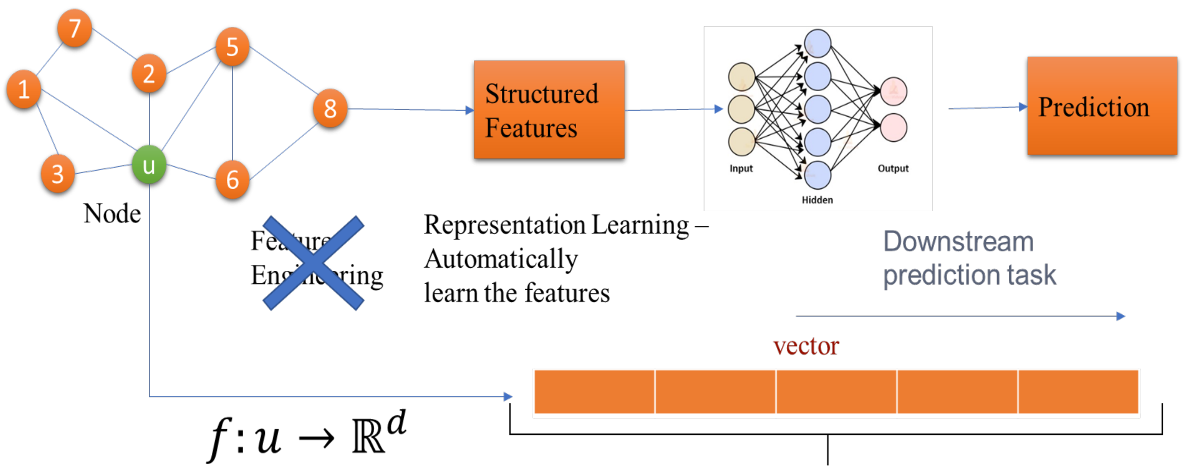


Figure 4.1: Feature Representation, Embedding of Graph

regression to make predictions. We are given an input graph then from that input graph we are creating structured features of the graph in that we can apply learning algorithms and make predictions. In addition, generally, most of the effort goes into feature engineering, in which scientists, humans, engineers try to figure out how to best describe, the particular network, which is more useful for the downstream prediction task [41-42].

Feature engineering is not done because with the increase of feature the length and dimension of the feature vector is also increased which include a high degree of computation so instead of having all the features, we concentrate only on the structural features which can be used with the Machine Learning algorithm to predict the node, link, and graph. Every single time for every different task we won't automatically want to learn the structure of the feature of the network is called representation. So, an idea behind graph representation learning is that we want to alleviate the need for manual feature extraction for every time and every different task, we want to automatically learn the features, the structure of the network, in which we are interested in and this is called representation learning so that no manual feature engineering is necessary anymore. Therefore, The goal will be to use graphs to do effective task-independent feature learning for machine learning. Therefore, for example, if we are doing at the level of individual nodes, so for every node we want to learn how to map the node in d-dimensional space $f: u \to R^d$and it as a vector of d numbers. R^d Vector of d numbers as feature representation or embedding. Therefore, the goal is that the mapping happens automatically and the vector captures the structure of the underlying network we are interested in or analysing or making predictions over. So the structure information is used for many kinds in different downstream prediction tasks.

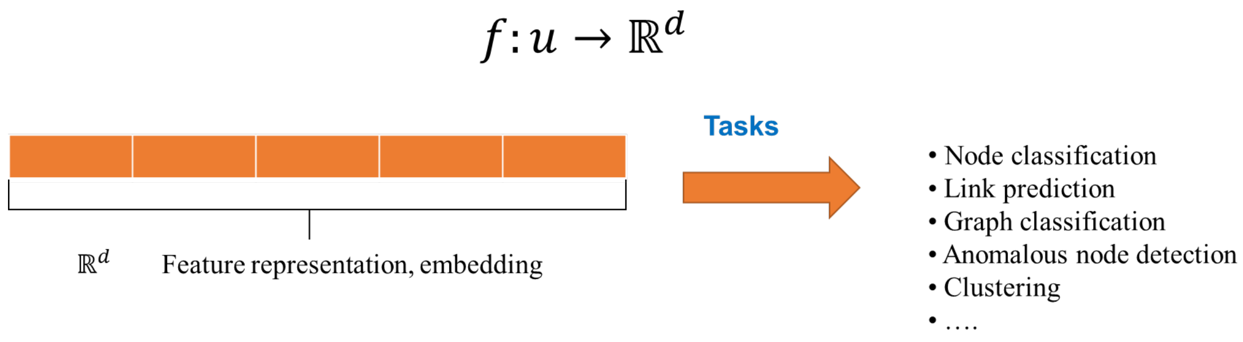


Figure 4.2: Structure Information for Downstream Prediction

4.2 *Feature Learning Framework:*

Given the graph G', on nodes, V' and an edge set E', i.e. $G' = (V',\ E')$. Our focus is to understand how to sketch out a route from the nodes, $f: u \to R^d$ to their embedding $f(u) = z_u$ and we are going to the maximum like hood objective. Our goal is to find the function, the mapping so basically finds the coordinates z of the nodes such that, the summation over the nodes, of log probabilities that given the node u, enhances the log probability of nodes in its local, random-walk environment. Basically, to sum out-to maximize the sum, which means we want to make nodes that are visited in the same random walk to be kind of embedded, close together. Therefore, we like to develop feature representations that could predict the nodes within every random walk neighborhood N.

1. Beginning with node u throughout the graph, we'll execute a brief static-length random walk using little random walks algorithm X.
2. To every node u we'll gather N of u and that is a collection of nodes explored in such a random walk beginning with node u. Multiset meaning that the same node can appear multiple times in the neighborhood because it may be visited multiple times.
3. Then we are going to optimize an optimization problem and optimize the embedding, so that given node u want to be able to predict who are the nodes that are in its neighborhood and defined again by the random walk. So, we are going to maximize the objective as follows:

$$\max f \sum_{u \epsilon V} \log \Pr\big(N_x(u) | f(u)\big) \tag{4.1}$$

We adopt two typical assumptions for making the problem with optimization tractable:

Conditional Independence: To factor the likelihood by assuming that, given the source's feature representation, the likelihood of identifying a nearby node is unaffected by the number of neighbors likelihood of viewing any other neighborhood node:

$$\Pr\big(N_x(u)|f(u)\big) = \prod_{n_i \epsilon N_x(u)} \Pr\big(n_i|f(u)\big) \quad (4.2)$$

Symmetric in Feature Space: A neighboring node and even a starting node within the feature space having a symmetric impact on each other. In the conclusion, we define the conditioned likelihood of every starting-neighborhood node combination as a SoftMax component restricted by both a scalar product regarding its features:

$$\Pr\big(n_i|f(u)\big) = \frac{exp\big(f(n_i) \cdot f(u)\big)}{\sum_{v \epsilon V} exp\big(f(v) \cdot f(u)\big)} \quad (4.3)$$

With the given assumptions, Eq. 4.1's goal can reduced to:

$$\max f \sum_{u \epsilon V}\Big[-\log Z_u + \sum_{n_i \epsilon N_x(u)} f(n_i) \cdot f(u)\Big] \quad (4.4)$$

For big networks, each-node partition function,
Z_u *is equal to* $\sum_{v \epsilon V} exp\big(f(v).f(u)\big)$ Expensive to compute, thus we use negative sampling to approximate it.

4.2.1 *Node2vec:*

To optimize the embedding for a given random walk, X the uniform random walk where it runs static-length unbiased every node starts a random walk. There is a similarity issue here, as there can be limits in many circumstances. As a result, to solve an issue, we must utilize more expressive, richer random walks which can fine-tune those embedding. As a result, node2vec is used, which embeds nodes with comparable network neighborhoods, narrowing a

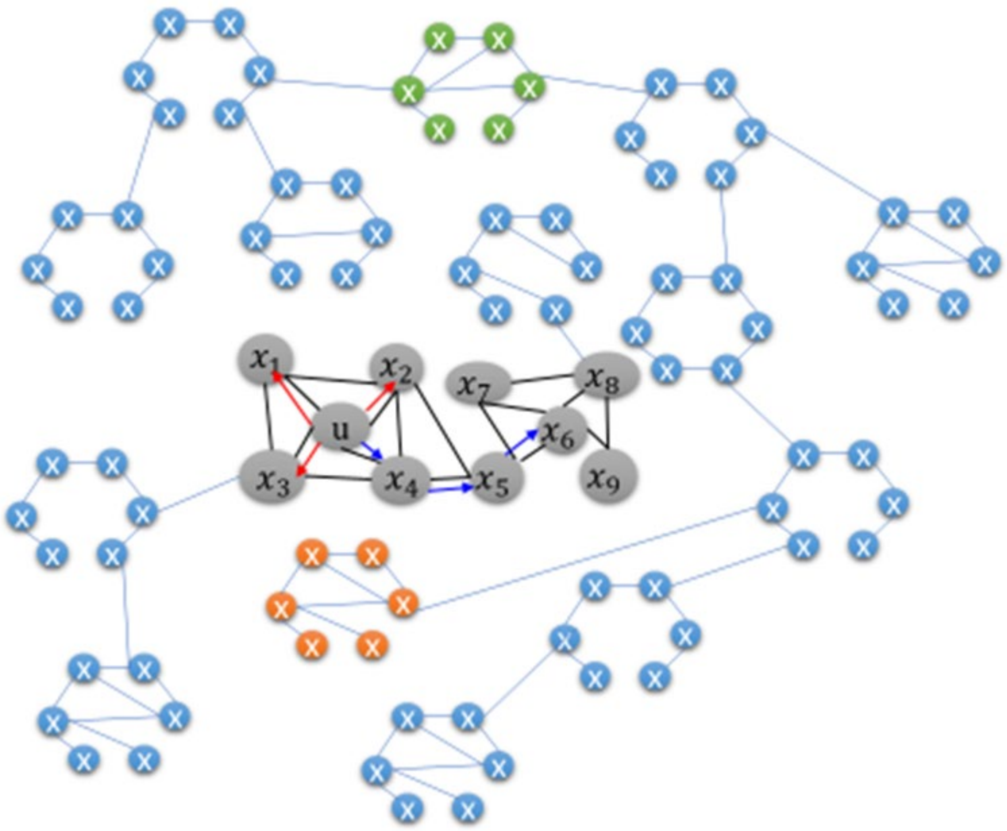


Figure 4.3: Two Connected Subgraphs with Structural Similarity

feature space, which leads too much richer, node embedding. It creates the network neighborhood N using a second-order random walk X, where it applies the same optimization problem as shown in Figure 4.3. Text can represent as linear words that propagate forward as you go in time and to get vector representation of this word. Let's say you can define sliding window and representation for this word depends on what are the words that occur in the context so under skip-gram given this word you want to predict what are the nearby words that occur to be in that window to that is how you context to natural language but network are not linear so you need some mechanism to kind of convert that non-linear structure that might have hierarchy as well as that is where the sampling strategies come into the pictures and how you do it again defines the richness of representation that you are trying to learn which again what we saw as in case of BFS and DFS in terms of homophily and structural symmetry [43]. It employs bias random paths to trade-off for both global as well as local network perspectives, flexible. We consider the challenge of sampling a source node's neighborhood's to be a type of local search. Figure 1 depicts a graph in which we intend to produce (sample) its neighborhood $N_s(u)$given a source node u. Importantly, we will limit the measurement regarding neighborhood set N_s in-order-to k nodes in order to meaningfully compare different sampling procedures S, after then, for a node u, sample numerous sets. There will be two different sampling procedures for constructing neighborhood set (s) N_s in-order-to k nodes:

1. **Breadth- First Sampling (BFS):** It used to calculate the shortest link between two nodes with unit weight edges. A neighborhood N_x is limited to nodes that are directly adjacent to the start. For instance, in Figure 4.4, BFS samples nodes x1, x2, and x3 considering a neighborhood regarding measurement k = 3.
2. **Depth-First Sampling (DFS):** The neighborhood consists of nodes that have already been collected in increasing order of range from the starting node. Samples of DFS x4, x5, and x6 shown in Figure 4.4.

Two traditional ways for defining a of node u neighborhood $N_x(u)$. Walk of size 3 ($N_x(u)$ regarding size 3):

$N_{BFS}(u) = \{x_1, x_2, x_3\}$ Local microscopic view $\rightarrow$ BFS
$N_{DFS}(u) = \{x_4, x_5, x_6\}$ Global macroscopic view $\rightarrow$ DFS

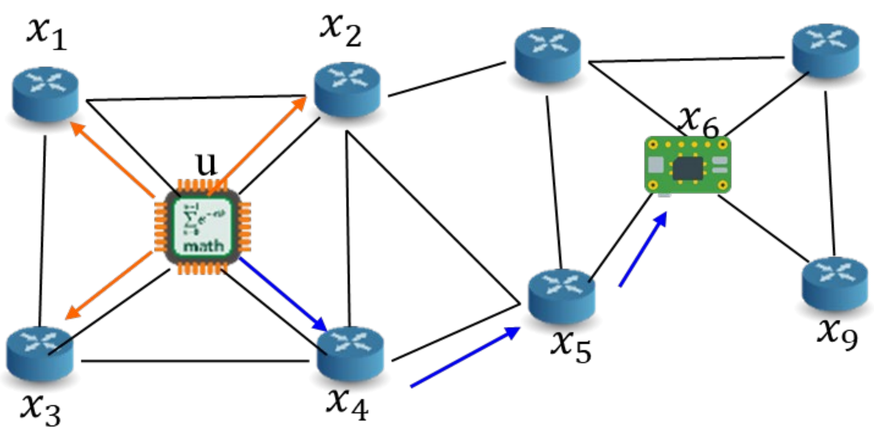


Figure 4.4: Local and Global Views of Network

BFS Prediction tasks involving nodes within networks, in particular, frequently switch between two types of similarity: structural and homophily equivalence. According to homophily theory, nodes that are connected as well as belong to similar network clusters or communities should be embedded close together. But from another side, the structural equivalence hypothesis suggests all nodes in networks with identical structural responsibilities should group collectively. Whereas homophily, organized equivalence somehow doesn't focus heavily on connection; nodes in a network might be far apart and still serve a similar functional function.

4.2.2 *Biased Random Walk:*

The easiest method to influence our random walks is to use the static edge weights to sample the next node. That, though, precludes us from optimizing the network architecture and targeting different ways of network areas using our search strategy. A second-order random walk is defined as one with two guiding parameters, p or q. Walker has crossed the edge (x_1, u) and is now at u. What should I do next?

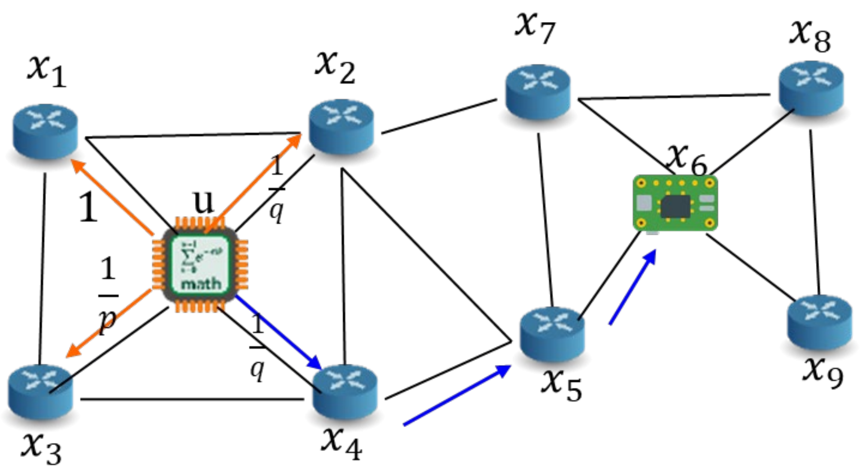


Figure 4.5: Biased Random Walks with Probabilities Distributions

p, q model transition probabilities BFS-like walk: Low value of p. DFS-like walk: Low value of q

p … *return parameter*

q …" walk *away" parameter*

$N_x(u)$ are the nodes that the biased walk passes through?

Table 4.1: Probabilities Distributions

Target t	Prob.	*Distance* (x_i, t)
x_1	$\frac{1}{p}$	0
x_2	1	1
x_3	$\frac{1}{q}$	2
x_4	$\frac{1}{q}$	2

Algorithm-5: node2vec

```
1. Calculate the probabilities of a random walk.
2. Start for every node u construct r random walks of length l
3. Use Stochastic Gradient Descent to improve the node2vec objective.
4. Linear-time complexity
```

Dataset Source: https://data.world/esimone/iot-platforms

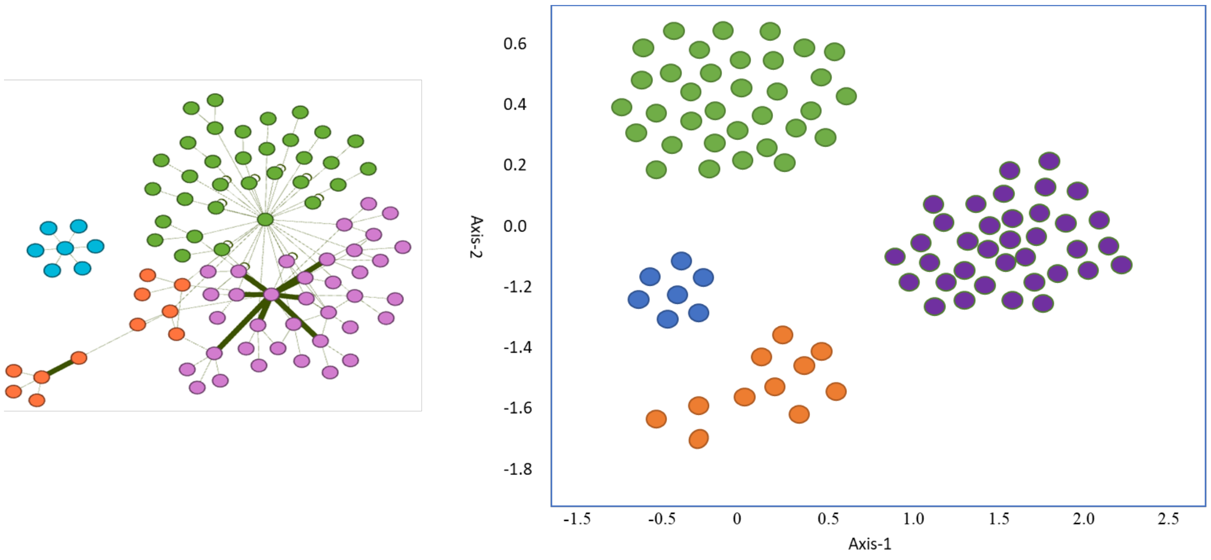


Figure 4.6: Node Embedding using Node2vec

4.3 *Spectral Clustering*

It's a way of data exploration for decomposing large multidimensional datasets into small clusters of data with similar data in fewer dimensions. The idea is to categorize all unorganized data sets into various classify based on how similar they are. The popular type of multivariate statistical analysis is called spectral clustering. It takes the connectivity approach to cluster, which identifies groups of nodes in a graph that are related or immediately next to one another. After that, the nodes are then grouped into a low-dimensional space which is combined skillfully [56]. In Spectral Clustering, the eigenvalues of the Laplacian Matrix are generated from the graph or data set. Suppose we have a graph, which obtained by the K nearest neighbor. To find the bipartition, we take the second eigenvector of Laplacian v_2, and the reason we take the second eigenvalue of Laplacian is that as we can see at the Laplacian matrix the sum of every column is equal to zero so the first eigenvalues is always zero, corresponding to λ_2, the algebraic connectivity of G.

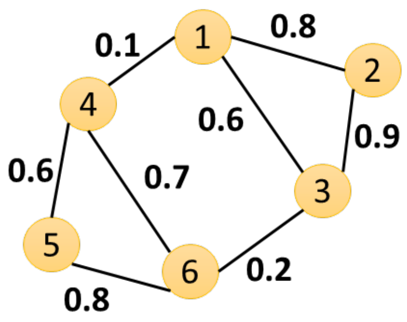
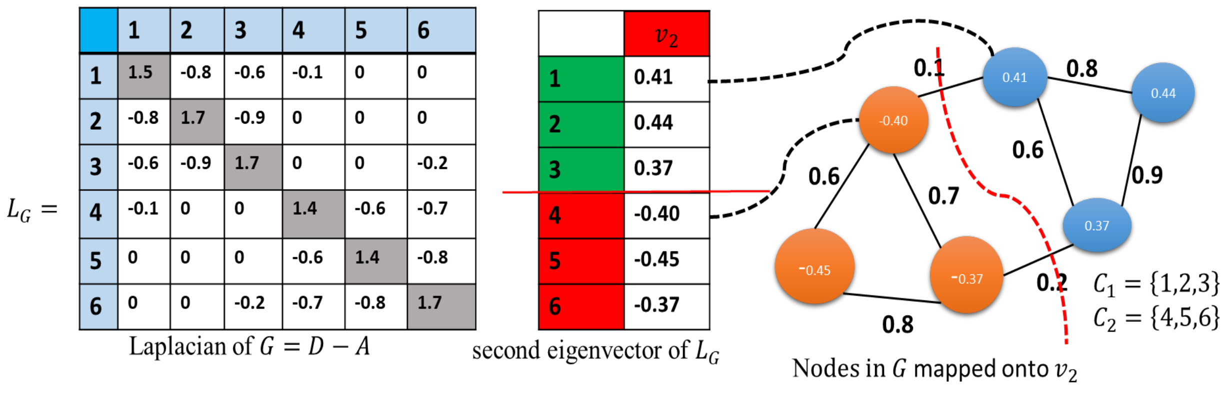


Figure 4.7: Weighted Graph for Spectral Clustering

4.3.1 *Bi partitioning via Spectral Clustering:*

To Find the bipartition, we choose a second Laplacian eigenvector v_2, which corresponds to λ_2, algebraic connectedness, G.

- The shorter λ_2 the greater the partitioning quality.
- Calculate the value $v_2(i)$ e.g.$v_2(1)$=0.41 in G to every node i in G,
- To find the clusters C_1 and C_2 calculate nodes along with:
  $v_2(i) > 0$ To C_1 as well as $v_2(i) < 0$ to $C_2$$C_1 = \{iIv_2(i) > 0\}$
  $C_2 = \{iIv_2(i) < 0\}$



$L_G =$

	1	2	3	4	5	6
1	1.5	-0.8	-0.6	-0.1	0	0
2	-0.8	1.7	-0.9	0	0	0
3	-0.6	-0.9	1.7	0	0	-0.2
4	-0.1	0	0	1.4	-0.6	-0.7
5	0	0	0	-0.6	1.4	-0.8
6	0	0	-0.2	-0.7	-0.8	1.7

Laplacian of $G = D - A$

	v_2
1	0.41
2	0.44
3	0.37
4	-0.40
5	-0.45
6	-0.37

second eigenvector of L_G

Figure 4.8: Spectral Clustering

Idea is the similarity of embedding's in the Network, the distance among nodes reveals their similarity. If two nodes are near to each other, for example, in Network perhaps should embedded close together in embedding space. By increasing the dimension of the vector, we can accurately predict the link, node and sub-graph in the network. Although the features of a node link or graph can be predicted from its feature vector but high dimensionally of this feature vector includes a lot of many computations which consumes high power and computing capacity. By doing the graph embedding or node embedding we are mapping the high dimensional feature vector into low dimensional embedding space which is very useful to predict feature of the new node.

We have discussed only two techniques over here:

1. Factor clustering which can use unsupervised learning.
2. Node2vec which can use semi supervised learning.

If we look at the Laplacian matrix the sum of every column is equal to zero so first eigenvalue is always zero. Only second eigenvalue will decide the partitioning. If second eigenvalue is very small the partitioning will be of very high quality. By using the recursive fashion, we can obtain multiple clusters.

Dataset Source: https://data.world/esimone/iot-platforms

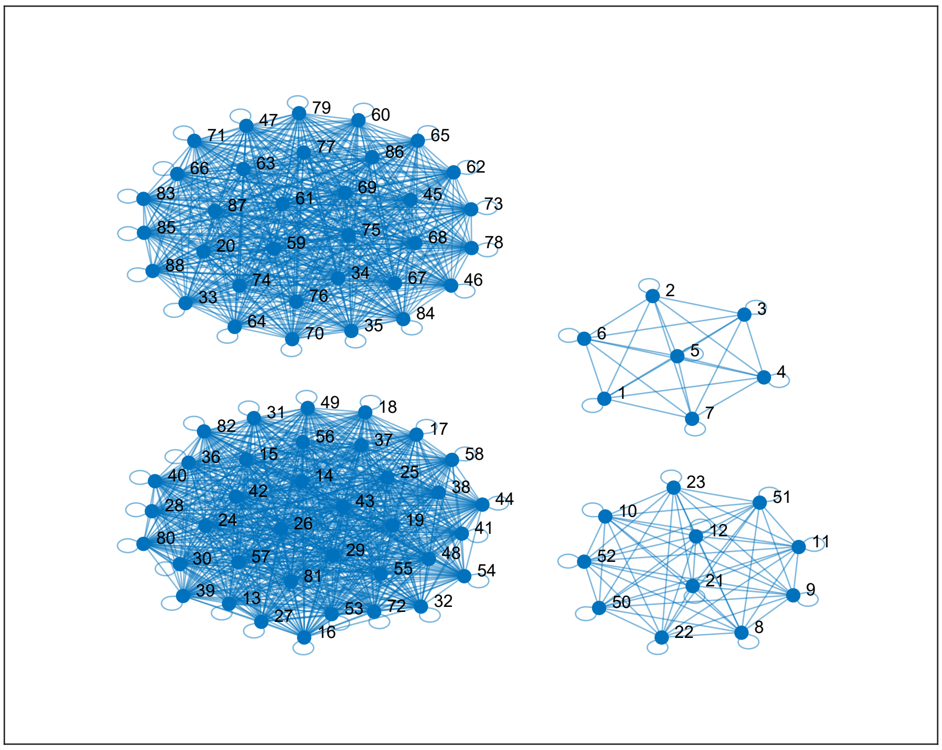


Figure 4.9: Spectral Clustering Partition Data Using Spectral Clustering

4.4 *Results and Discussions*

The feature engineering is the manual computational task to train the graph. The efficiency of graph training for downstream prediction can be increased by node embedding or graph embedding where high dimensional feature vector is mapped into the low dimensional feature space using biased ransom walk. Further the mapped nodes can be divided in to the clusters using spectral clustering.

CHAPTER 5

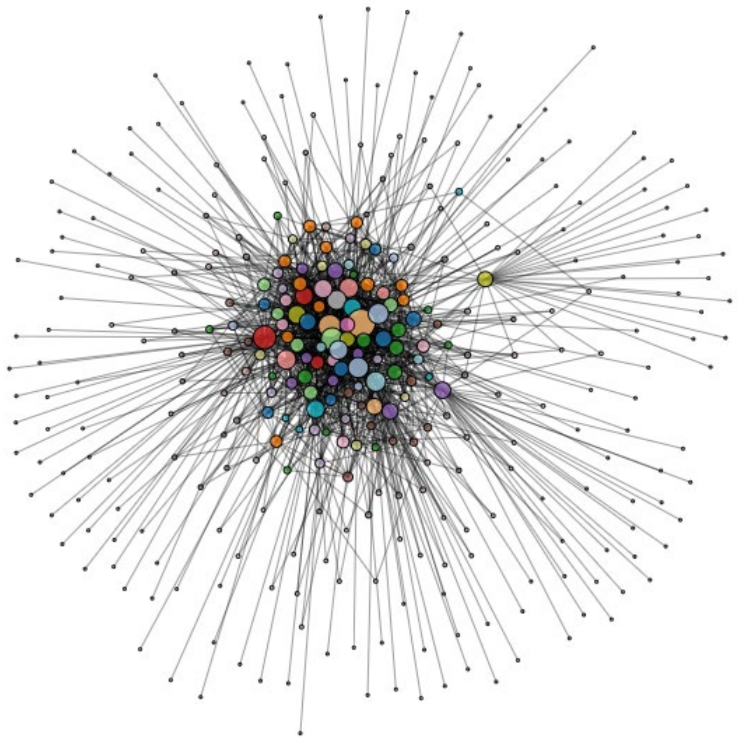
SOCIAL IOT NODE/PAGE RANK

5.1 *Introduction and Motivation*

The focus of this chapter is on learning from a matrix as well as a graph analysis standpoint. By considering a graph as a matrix, you can:

1. Use a random walk for determining the relevance of nodes (PageRank).
2. Matrix factorization (MF) is for obtaining the node embeddings.
3. Description of another node embedding's as MF (Node2Vec).

See another node embedding's as MF (such as Node2Vec). Node embedding's, MF, Random walks, and all have a lot in common. All the smart objects in SIoT are the entities of a connected graph, where IoT objects are the nodes and communication links in between then are edges of the graph [61]. It is basically the main innovation that lead Google Search become, Google Search. It was developed by the two students of Stanford University of Computer Science, Larry Page and Sergey Brin. For example, there is the context of Google and the context of the Web Search and Web as a Graph. So, basically the first basic question is how to represent the Web? If it is represented as a graph, where nodes would be Web pages and link between the nodes would be hyperlinks so that anyone can navigate from one page to another. Today, the Web has evolved a lot and there is the issue, of what is a node? There are a lot of dynamic pages created on the fly, and there is a lot a Dark Web, so a lot of places, that are passwords protected, that are inaccessible, by basically just browsing the Web. Basically, Web in older days had a set of static pages. And then the static pages have hyperlinks, so because of which the links that anyone can click and move to the next page. So, in early days, this is how web structured with a lots of static Web pages and a lot of navigational links. Numerous links presently are commercial, indicating they are being used to complete activities such as posting something, commenting on something, liking something, buying something, and then moving to another website. There are many other places or other types of information that can represent in the same way. For example, there are many citations where nodes would be, page and page are citing to another paper, which means that is a directed link from the source paper to the paper it cites and it is called a



Sensors/Actuators	7.1K
Communication Links	107K

Figure 5.1: SIoT Prototype

citation networks. Even references in Encyclopedia like Wikipedia, and represent them as a graph of how one concept is related or links to another concept. SIoT graph prototype is shown in Figure 5.1.

5.2 *Ranking Nodes on the Graph*

Web search to understand what nodes on the web are important than the others? Some unknown domain name might be a priori less important, less trustworthy than, well-established domain name, well-established website that a lot of other, web pages link. For example, thisperson.com versus jpuniversity.edu, they both might have very different a priori, importance. Because the web-graph connection structure is so diverse, it might utilize to rank the pages in a sense, which one is more important, more popular, more trustworthy, and which one is less important, less trustworthy, and less popular, in a sense when anyone rank search results, this can use as a signal into what to put on the top and what to put on the bottom? This is the idea how it was developed.

5.3 *SIoT Link Analysis Algorithm*

Link Analysis Algorithm and approaches to Link Analysis, in which the step is to find out how significant nodes in a graph are. It is categorized into three parts:

1. Page/node Rank
2. Personalized Page Rank (PPR)
3. The relevance of the source IoT node is proportional to the significance of every link's connection.

5.3.1 *Social IoT Node/ Page Rank Flow model*

So, the goal is to, compute the importance of a Web pages on the Web. The concept is to consider connections as votes. As a result, if a page has more incoming or outgoing links, it is considered more important. Incoming links are harder to fake because other people on the Web have to link to the one and the outgoing link are easier to kind of fake because it can generate them on the website. So, let's just use links-to-links as voters [62]. For example, jaypeeuniversity.edu has 10,000 in-links and thisperson.com has one in-link. So, jaypeeuniversity.edu is perhaps more important. All in-links are not equal like an in-links from an important page should count more. It is now recursive, because the confidence, the amount of vote it get from anyone depends on their importance.

1. If page i has di out-links and priority r_i each link gets $\frac{r_i}{d_i}$ connections.
2. The relevance of page/node $j's$, the sum of votes on its in links is r_i.
3. If another major page's link to a page, it is important.
4. The following is the importance ranking:

$$r_j = \sum_{i \to j} \frac{r_i}{d_i} \qquad d_i \ldots \text{out-degree of node i} \tag{5.1}$$

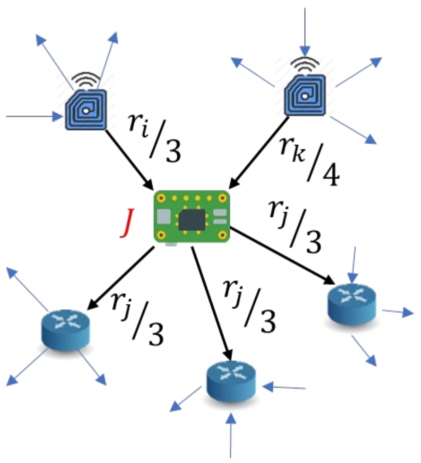


Figure 5.2: IoT Node Rank Flow Model

$$r_j = {}^{r_i}/_3 + {}^{r_k}/_4 \tag{5.2}$$

$$\begin{array}{c|c|c|c|}
 & r_y & r_a & r_m \\ \hline
r_y & ½ & ½ & 0 \\ \hline
r_a & ½ & 0 & 1 \\ \hline
r_m & 0 & ½ & 0 \\ \hline
\end{array}$$

$$\mathbf{M}$$

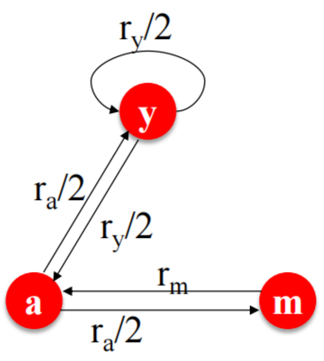


Figure 5.3: IoT Node Rank Flow Model Equations

With the help of example of SIoT Web graph, back in old days the web use to be very small, there were only three web pages on the web, and the hyperlinks structure of the web graph. The equations for this network flow dynamics are:

$$r_y = {}^{r_y}/_2 + {}^{r_a}/_2 \tag{5.3}$$

$$r_a = {}^{r_y}/_2 + r_m \tag{5.4}$$

$$r_m = {}^{r_a}/_2 \tag{5.5}$$

Know there are three unknown and three equations, whether it should be solved with the help of Gaussian elimination. For doing this it needs a fourth constraint which is the sum of equal to the one, or could use the large- scale equation solver. It is not possible because it's not scalable. So, there exists so much more elegant way to solve this system of equations. For this stop looking at the graph as a set of nodes and edges, but represent it as a matrix. So, this notion is defining as a Stochastic Adjacency Matrix M, where if a page j has out-degree d_j, and the sum of all the columns individually is equal to one. So, if $j \rightarrow i$, then $M_{ij} = {}^{1}/_{d_j}$[63].

5.3.1.1 Rank Vector **r**

It will have only one entry per page, and entry i of the vector r will be the importance score of page i.

$$\sum_i r_i = 1 \tag{5.6}$$

So, the probability distribution of over the nodes, in the network.

The flow equations can be written:

$$r = M.r \tag{5.7}$$

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

$$\underbrace{\boxed{\begin{array}{c} r_y \\ r_a \\ r_m \end{array}}}_{r} = \underbrace{\boxed{\begin{array}{ccc} ½ & ½ & 0 \\ ½ & 0 & 1 \\ 0 & ½ & 0 \end{array}}}_{M} \underbrace{\boxed{\begin{array}{c} r_y \\ r_a \\ r_m \end{array}}}_{r}$$

Consider the following scenario for a random web surfer:

1. The surfer is on another page i, at any given time t.
2. At given time $t + 1$, the surfer will randomly follow an out-link from i.
3. It stops up on a page j that is connected from i.
4. The procedure is rehashed endlessly.
   - i. $p(t)$ ….. be a vector with the i^{th} coordinate representing the chance that the surfer is on page i at time t.
   - ii. As a result, $p(t)$ denotes the probability over pages.

The flow equations in terms of Random Walk:

$$p(t + 1) = M.p(t) = p(t) \tag{5.8}$$

$r = M.r$ satisfied by the original rank vector r. So, for a random walk, r is the stationary distribution.

5.3.1.2 Power Iteration

$$r = M.r$$

$$\begin{array}{c|c|c|c|} & r_y & r_a & r_m \\ \hline r_y & ½ & ½ & 0 \\ \hline r_a & ½ & 0 & 1 \\ \hline r_m & 0 & ½ & 0 \\ \hline \end{array}$$

0th Iteration $r = \begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$ **Repeat till converge to final solution**

1st Iteration

$$r = \begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix} \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix} = \begin{bmatrix} 1/3 \\ 3/6 \\ 1/6 \end{bmatrix}$$

final solution

$$\begin{pmatrix} r_y \\ r_a \\ r_m \end{pmatrix} = \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix} \begin{pmatrix} 1/3 \\ 3/6 \\ 1/6 \end{pmatrix} \begin{pmatrix} 5/12 \\ 1/3 \\ 3/12 \end{pmatrix} \begin{pmatrix} 9/24 \\ 11/24 \\ 1/6 \end{pmatrix} \dots \begin{pmatrix} 6/15 \\ 6/15 \\ 3/15 \end{pmatrix}$$

Iteration 0, 1, 2, …

5.3.1.3 *Brin-Page Rank Equation*

Person who randomly navigates the web pages of the web. Somebody who surfer at some time t is one of the pages of the web gets bored, they decide to randomly surfer the web and jumps to another page.

Random surfer has two alternatives at each step:

- Pick a link at random β and follow it.
- Chance to go to a random page or node $1-\beta$.

$$r_j = \sum_{i\to j} \beta \frac{r_i}{d_i} + (1-\beta)\frac{1}{N} \tag{5.9}$$

New column stochastic matrix is:

$$G = \beta M + (1-\beta)\left[\frac{1}{N}\right]_{NXN} \tag{5.10}$$

The solution is recursive

$$r = G.r \tag{5.11}$$

In practice β =*0.8,0.9* (make *5* steps on avg., jump)

The equation 5.7 is rewritten as

$$G = \beta M + (1-\beta)\left[\frac{1}{N}\right]_{NXN} = \underbrace{0.8}_{\beta}\underbrace{\begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix}}_{M} + \underbrace{0.2}_{(1-\beta)}\underbrace{\begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}}_{\left[\frac{1}{N}\right]_{NXN}} \qquad (5.9)$$

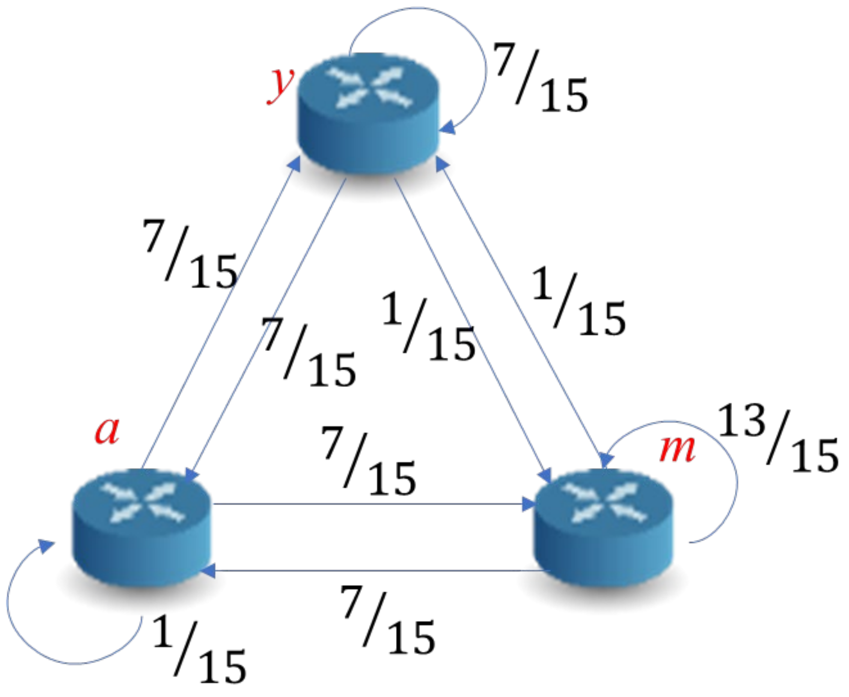


Figure 5.4: Brin-Page Rank Flow Model

$$G = \begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{bmatrix}$$

The transpose of G:

$$G' = \begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 7/15 \\ 1/15 & 1/15 & 13/15 \end{bmatrix}$$

It has seen that the sum of each row of $G'$is equal to one. It implies that the first eigenvalue of this matrix is equal to one. It reveals the nature of page rank is a Regular Markov Chain as shown in Figure 5.5.

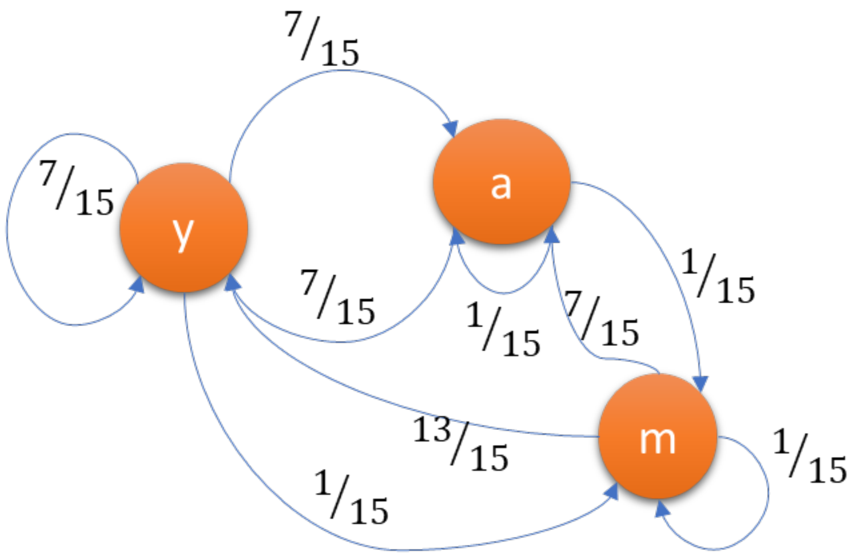


Figure 5.5: Brin-Page Rank Regular Markov Chain

The recursive solution using equation 5.11 is:

$$\begin{array}{l} y \\ a \\ m \end{array} = \begin{array}{c} 1/3 \\ 1/3 \\ 1/3 \end{array} \quad \begin{array}{c} 0.33 \\ 0.20 \\ 0.46 \end{array} \quad \begin{array}{c} 0.24 \\ 0.20 \\ 0.52 \end{array} \quad \begin{array}{c} 0.26 \\ 0.18 \\ 0.56 \end{array} \quad \cdots \quad \begin{array}{c} 7/33 \\ 5/33 \\ 21/33 \end{array}$$

The visualization and page rank distribution are shown in Figure 5.6 and 5.7.

Dataset Source: https://data.world/esimone/iot-platforms

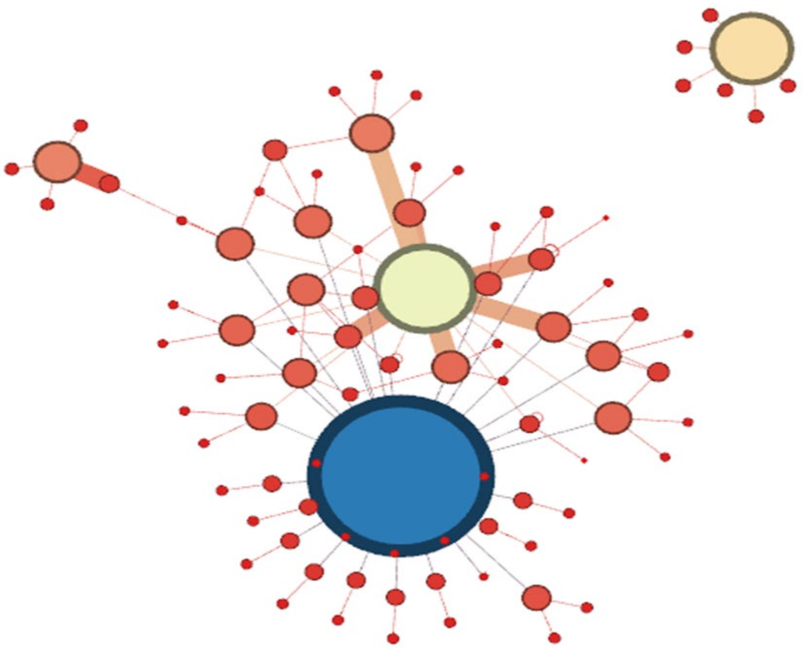


Figure 5.6: Visualization of Page Rank Distribution

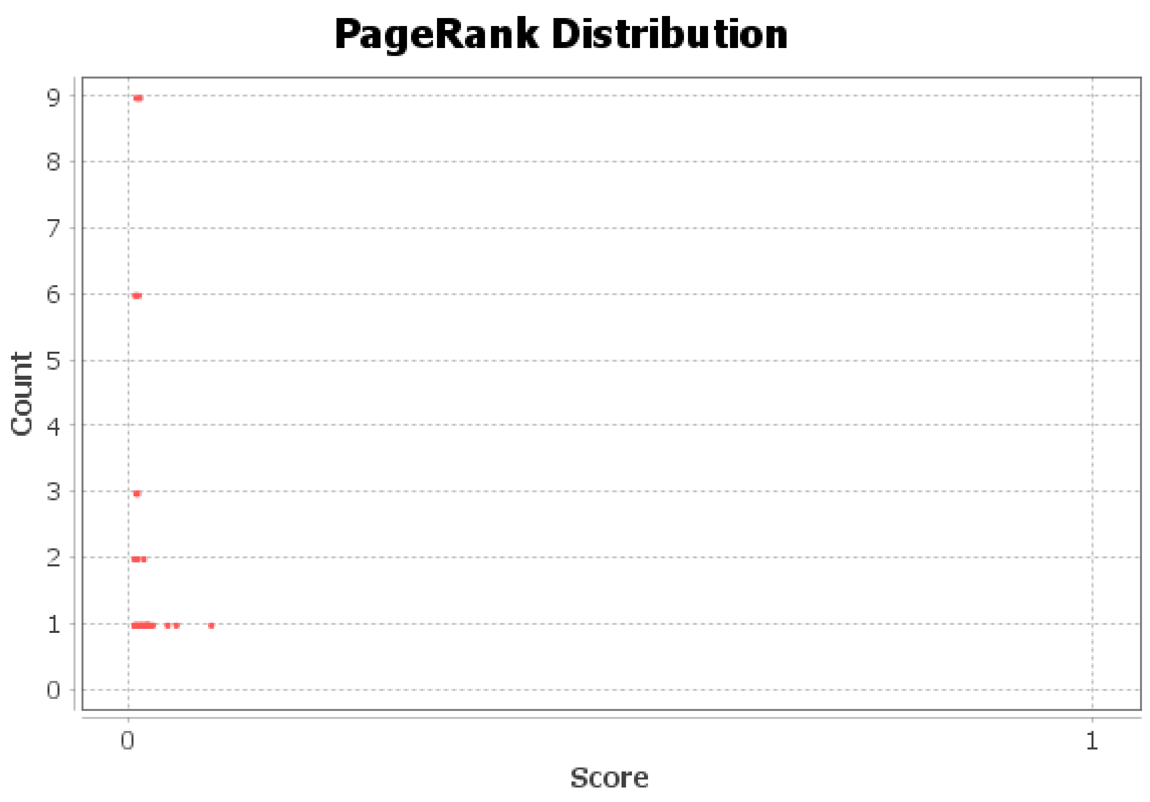


Figure 5.7: Page Rank Distribution

5.4 *Result and Discussion:*

Page/ Node Rank in a SIoT gives the importance in a node in SIoT. The most significant use of node rank to determine the node of importance for an automated, broadcasting task in diversified IoT applications. The possibility of multiple node navigation is addressed by Brin-Page Rank Equation where the generated regular Markov Chain gives the recursive steps during converging to a final solution.

CHAPTER 6 COMMUNITY DETECTION FOR DISASTER MANAGEMENT

6.1 *Introduction and Motivation*

Social media has surpassed email as the most widely used method of interaction. Individuals can voice their opinions, shared data, advertise their enterprises, and advertise future projects, Locate prominent people, and engage in political activism. As a result, the amount of knowledge is available on media platforms is rapidly growing. This increase of data has created new barrier for viewing, evaluating data published on social media platforms [73]. Social Network are common examples of networks with community structure as a key characteristic. A set of community is a group of people who is strong, personal, powerful, regular, and good ties with one another. The problem of identifying communities in systems is crucial. As a result, a numerous strategy for community detection have been proposed. Understanding the difficult behavior of complicated networks is aided by community structure. It is the set of web pages that are all about the similar thing: communities. Communities aid in the comprehension, and analysis, and discovery of hidden patterns, as well as recommendation systems, location and abased information models, data dissemination, and branding. A network is a subgraph of the Network, which are groups of nodes that allow people to interact and build relationships with one another [75].

Over the last years, complex network has risen in popularity. With the introduction of Social Platforms like Facebook, Twitter etc., and Social Networks of representing the fundamental building block become accessible, paving the way for the computational social sciences to emerge. However, networks are also ubiquitous in fields like neurology and biology. Several of these networks have a number of qualities in common. They have asymmetrical degree patterns, a high degree of clustering, and a short average path length. Nodes frequently cluster together in dense clusters, which are referred to a community. Metabolites with comparable functions and persons with similar backgrounds are common among nodes in a community. The network can thus be better understood by revealing the community structure.

Despite its shortcomings, modularity continues to be the most effective community detection strategies. Numerous algorithms for improving modularity have been proposed. The initial algorithm generated a full dendrogram and decided on a cutting point based on modularity. It slow, with a $O(n^2m)$ time complexity, where n is the number of nodes and m is the number of links. External optimization, simulated annealing, spectral approaches, greedy methods, and a variety of other methods were swiftly adopted to improve modularity. The Louvain algorithm, which is the thought to execute in $O(m)$ is one of the quickest and most effective algorithms. In competitive real-world tests, it has proven to be quite effective. Because the algorithm is generally independent of the target function to optimize, it has been used to a variety of techniques.

6.2 *Modularity Clustering*

Modularity which measures the quality of a given split then we will present the Louvain method an approximation algorithm that tries to find a graph partition with a high modularity score. One of the enduring problems of community detection is there is not an official definition of what makes a community. From sociology, this emphasizes that actor in a community have many direct and frequent ties with others in their community.

Therefore, when we divide a network into two communities, we have to take into account two thing:

1. The structure within a community we want a community to be internally coherent.
2. The structure between communities we do not want lots of coherence between two separate communities if that were the case we would rather merge those communities into.

We want to think about communities as maximally coherent sub-networks for example we were add a few edges to this network we may choose to merge two communities into one but it also might depend upon which edges we add how many and where they connect. We want many edges within a community and few edges between communities and note that this definition is both global and local. We start with a network and partition into communities. We then look at the fraction of edges that run between the communities ideally. We keep all the vertex degrees the same but recreate the edges by randomly pairing up vertices and this known as the configuration model. If our partition does much better on the actual network compared to the average random network then we have confidence that our choice of community structure is a good one finally it is worth noting that during the process we focus on maximizing edges within communities as a consequence this tends to minimize edges between communities [79-80]..

Here is the mathematics behind validating our partition into communities against a randomly rewired network the quantity Q, which is called the modularity of the community partition this formula for undirected network appears. The formula for directed networks is similar.

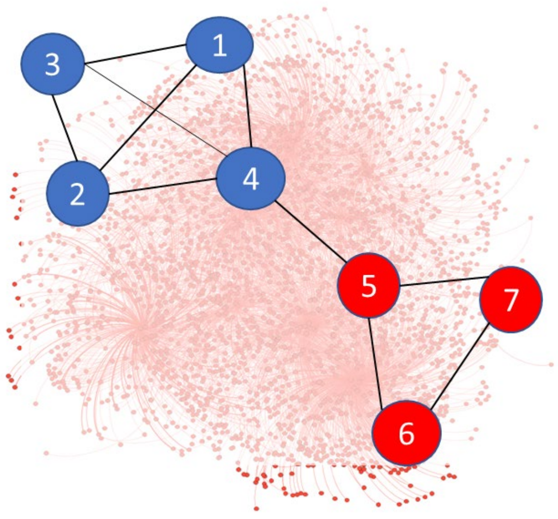


Figure 6.1: Modularity clustering

$$Q = \frac{1}{2m} \sum_c \sum_{i,j \epsilon c} \left(A_{ij} - \frac{d_i d_j}{2m} \right) \qquad 6.1$$

Where

A is the adjacency matrix, $i \; and \; j$ are vertices in the network

In network, we allow for multiple edges, which we will represent as weighted edges for the adjacency matrix if there are 3 edges between i, j then A_{ij} will be 3 and if there are 7 edges between them then A_{ij}would be equal to 7. m is the total number of edges so $2m$ is just the sum of the degrees in the network meanwhile d_i and d_j are just the degrees of the verticesi, j. Therefore, c is the name of the community or cluster containing vertex i, j.

Here in modularity formulae we using triple sum to calculate modularity pick a community c then pick a vertex i in that community then pick another vertex j and now calculate

$$A_{ij} - \frac{d_i d_j}{2m}. \qquad (6.2)$$

A_{ij} is the number of edges connecting i and j and comapare the actual number of i, j edges to the expected number of i, j edges in the randomly rewired network. To use modularity in practices naïve arguments, show that the modularity score is somewhere between negative one and one a positive score means that our community partition has done better than average a higher score corresponds to more pronounce community structure. Scores above 0.3 significant and score zero mean that the partition has not picked up any community structure. That is how we use modularity to validate whether a given partition does capture some community structure in the network.

Degrees of the 7 nodes are: $d = [3,3,3,4,3,3,2]$

Total Degree: 2m=20.

The modularity matrix below has (i, j) entry: $A_{ij} - \frac{d_i d_j}{2m}$

Node 1, 2, 3, 4 tend to form one community and node 5, 6, 7 for another. The Modularity Q based on this division is the sum of all green cells in modularity matrix divided by 2m: 0.355

Table 6.1: Modularity Matrix

j i	1	2	3	4	5	6	7
1	-0.45	0.55	0.55	0.4	-0.45	-0.3	-0.3
2	0.55	-0.45	0.55	0.4	-0.45	-0.3	-0.3
3	0.55	0.55	-0.45	0.4	-0.45	-0.3	-0.3
4	0.4	0.4	0.4	-0.8	0.4	-0.4	-0.4
5	-0.45	-0.45	-0.45	0.4	-0.45	0.7	0.7
6	-0.3	-0.3	-0.3	-0.4	0.7	-0.2	0.8
7	-0.3	-0.3	-0.3	-0.4	0.7	0.8	-0.2

6.3 *Louvain Method for Community Detection*

It is a very popular community detection algorithm. This approximation algorithm tries to maximize the modularity score and does well in practice. One nice feature of the vein method is that it decides how many communities to return as part of its process. Basic flow of the algorithm we start with the network and perform a modularity optimization process this splits the network into four communities. Next, we aggregate the communities into single nodes, internal edges become loops of weight. Meanwhile external edges merge into a single edge of the proper weight. Once created, the aggregation network we repeat the modularity optimization on the new network.

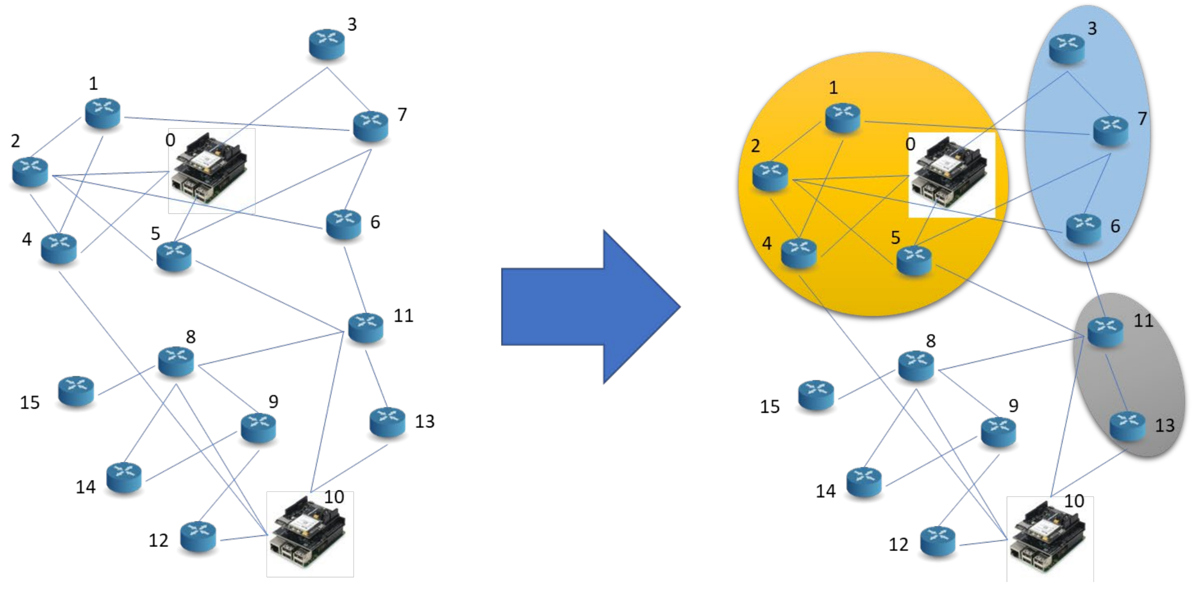


Figure 6.2: Detected Communities on the IoT network

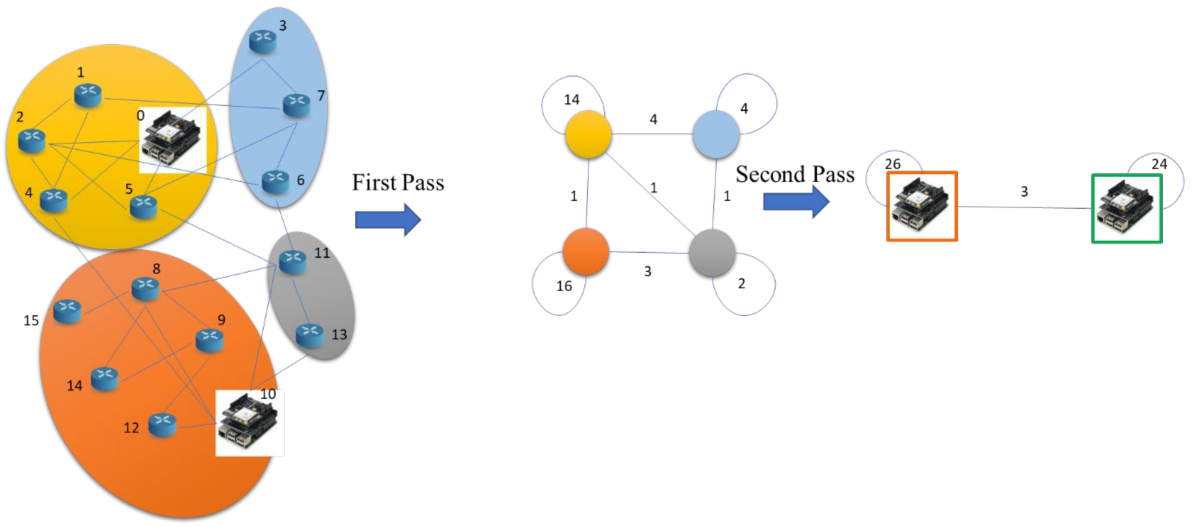


Figure 6.3: Detected Communities on the IoT network

When the process stops merging our Meta community nodes, we stop the process and then return the corresponding split in the original graph. We can chase back through the process to find the split into these two communities. The Modularity Distribution with different resolution is shown in Figure 6.4 -6.6.

Modularity: 0.535
Modularity with resolution: 1
Number of Communities: 8

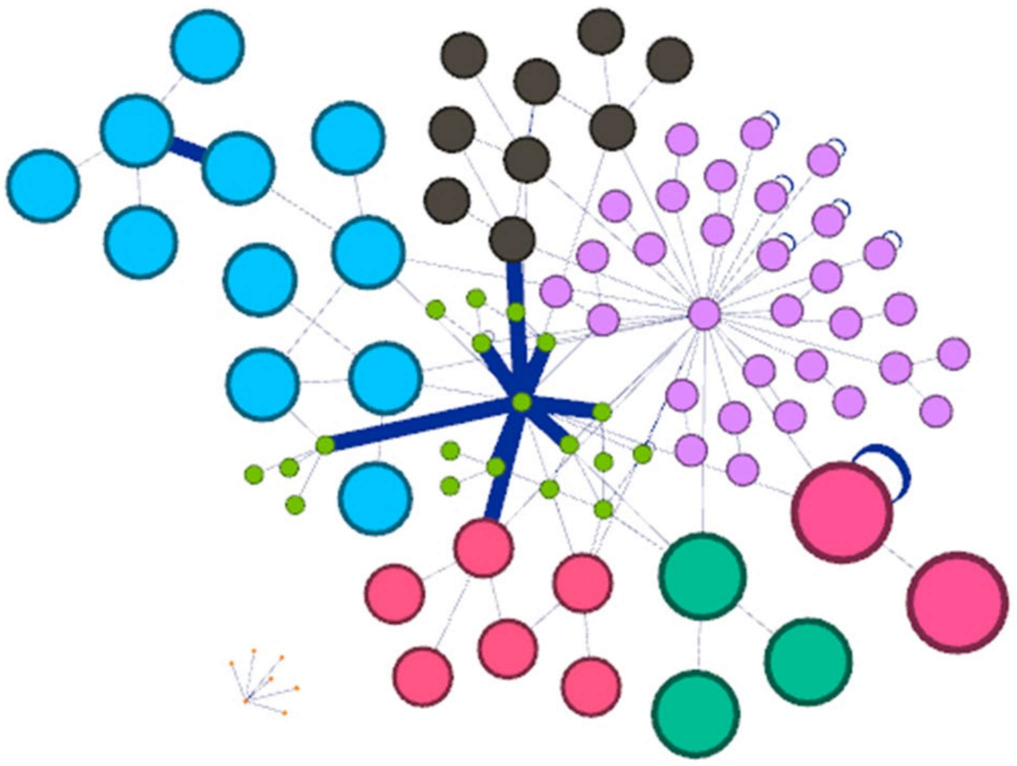


Figure 6.4: Visualization of Modularity Distribution with resolution: 1

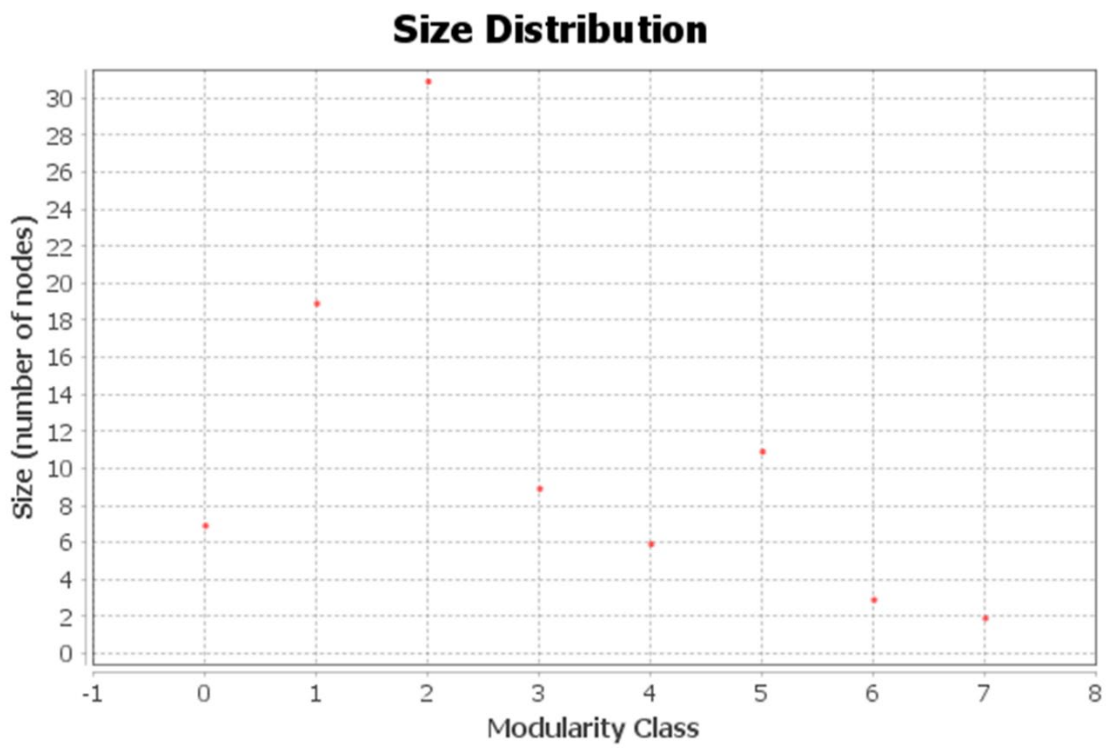


Figure 6.5: Modularity Distribution with resolution: 1

Modularity: 0.467

Modularity with resolution: 2

Number of Communities: 4

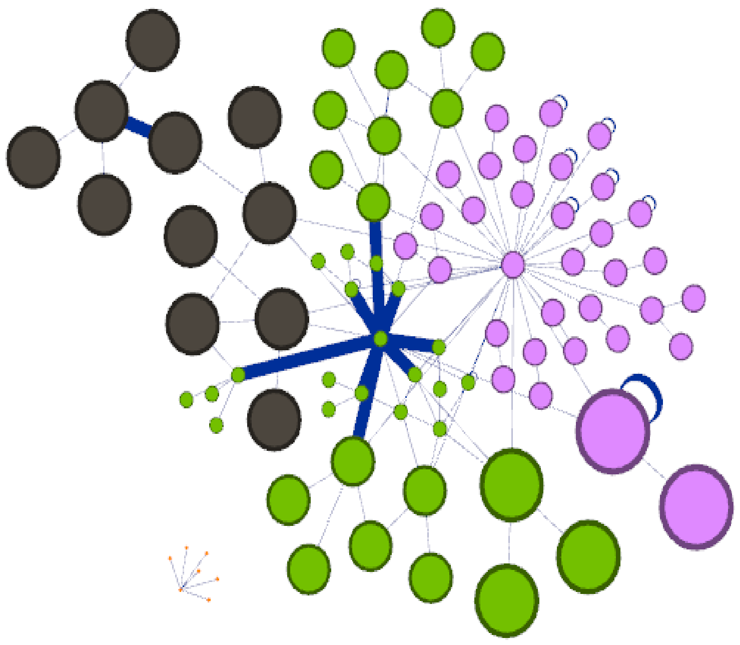


Figure 6.6: Visualization of Modularity Distribution with resolution: 2

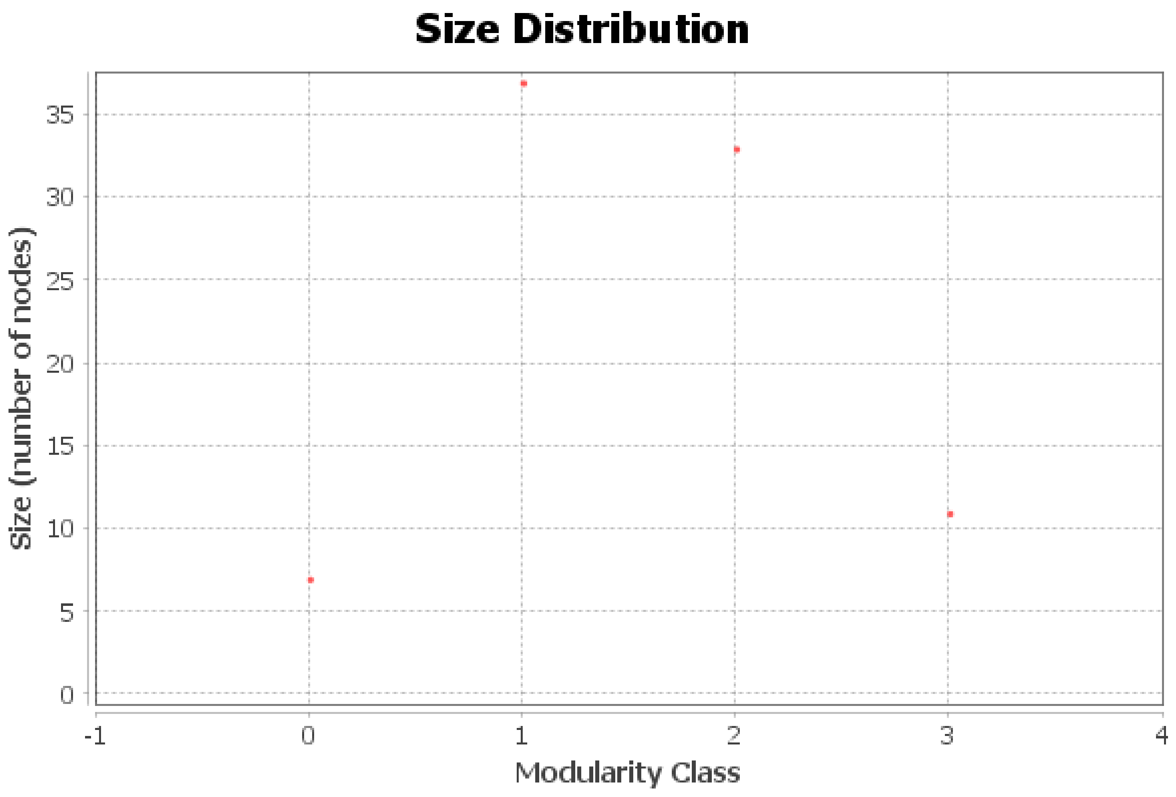


Figure 6.7: Modularity Distribution with resolution: 2

6.4 *Disasters Management*

Natural disasters are not predefined. They may appear in the natural processes of the earth in the form of landslides, earthquakes, floods, firestorms, and other geological processes. A disaster can be natural or man accelerated where the infrastructure is developed by the human may cause disaster due to poor quality of construction and aged in fracture. The IoT sensors may be deployed statically or dynamically in disaster-prone areas. These sensors are highly sensitive and accurate, deliver the information very precisely to the next stage to take the decision. The disaster management algorithm is burnt in the IoT devices is according to the input data from the sensors. The decision is taken and broadcasted to the IoT gateway. The situation sensed before the natural disaster occurrence is broadcasted, the early warning to SIoT as shown in Figure 6.8. The contribution of this chapter to identify the node with high centrality. Here the early warning system is to be interfaced. The eigenvalue of dynamic Social IoT is continuously monitored to identify the node which high centrality as shown in figure 6.9.

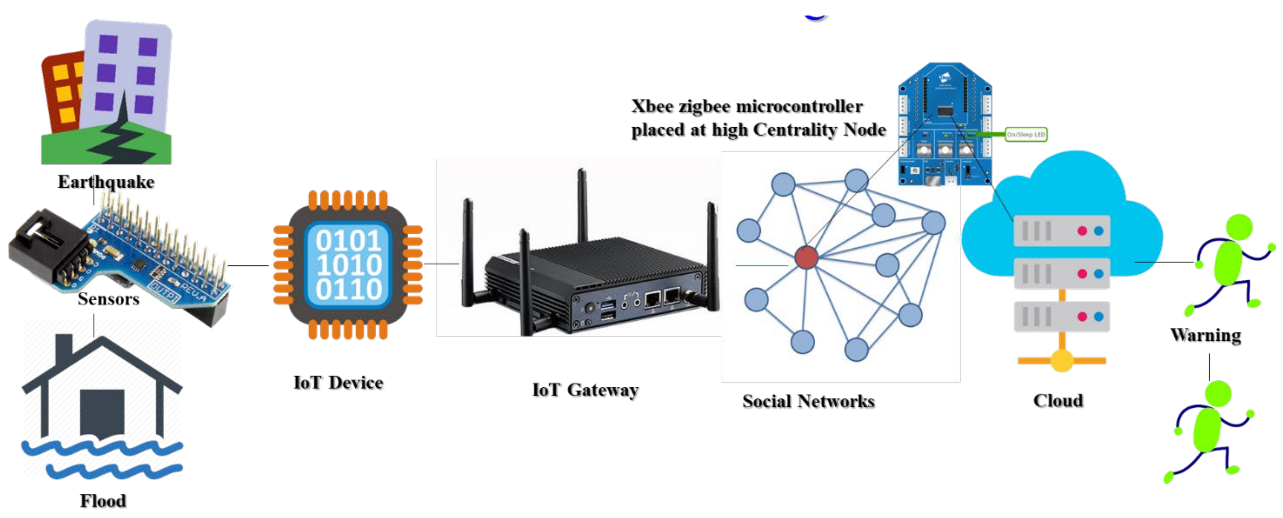
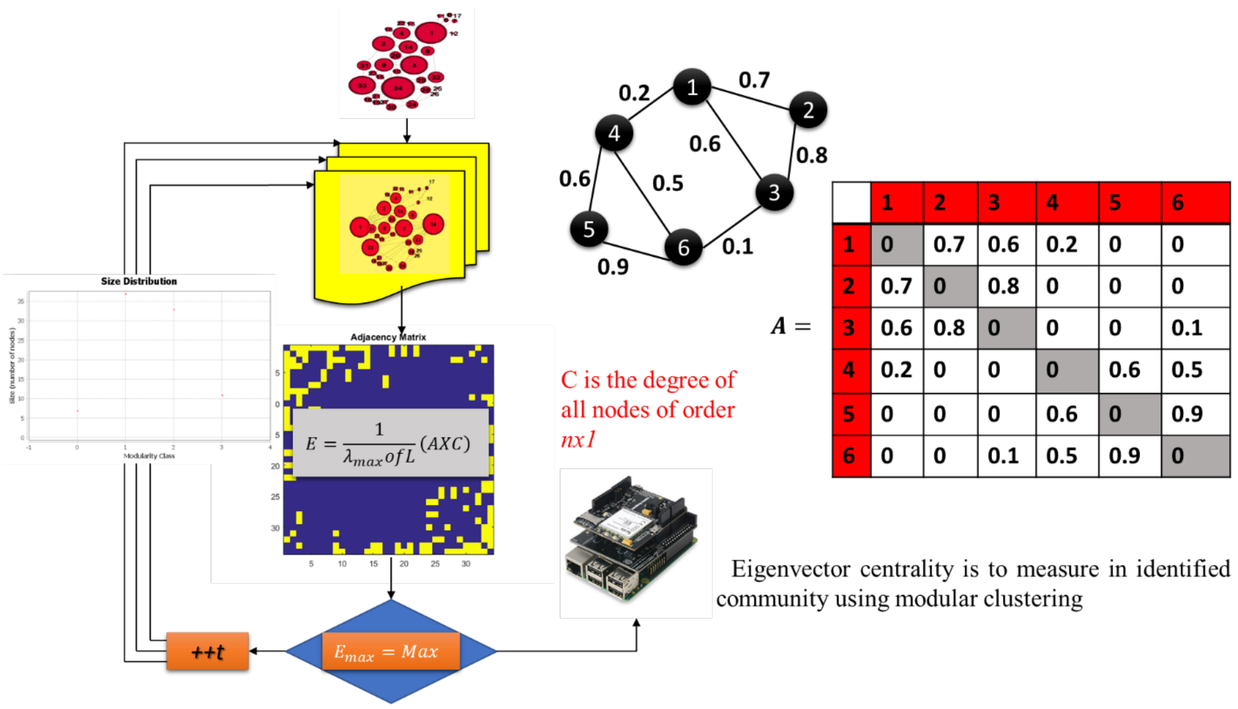


Figure 6.8: Natural Disasters Management



$$E = \frac{1}{\lambda_{max} of L}(AXC)$$

C is the degree of all nodes of order *nx1*

$$A =$$

	1	2	3	4	5	6
1	0	0.7	0.6	0.2	0	0
2	0.7	0	0.8	0	0	0
3	0.6	0.8	0	0	0	0.1
4	0.2	0	0	0	0.6	0.5
5	0	0	0	0.6	0	0.9
6	0	0	0.1	0.5	0.9	0

Eigenvector centrality is to measure in identified community using modular clustering

Figure 6.9: Early Warning system placement in Natural Disasters Management

6.5 *Results and Discussions*

The proposed algorithm for early warning system placement in natural disaster management is very accurate and fast when used with dynamic SIoT. It identifies the community of interest using modularity clustering and Louvain method for community detection. The modularity clustering identifies the different communities there one can find out the requirement of early warning system interface on a node with high centrality.

CHAPTER7

CONCLUSIONS AND FUTURE SCOPE

The social relationship between humans is well promoted by internet platforms like Facebook, Instagram, Twitter, and personnel blogs. In today's scenario, IoT technology promotes digital objects to communicate with each other. The expected number of these digital objects is in the range of trillions. The established communications between the smart IoT objects form a network that is the replica of social networks. By embedding these smart objects into social network format, navigability, scalability, and trustworthiness can be used to authenticate the social object relationship.

The research work in this dissertation has been carried out to address the trustworthiness of smart objects in IoT as a function of the node, link, and subgraph. The structural features for node-level prediction are node degree, node centrality, clustering coefficient, and graphlet. Similarly, the link level structural feature are: local and global neighborhood overlap, are discussed in Chapter 3. The compatibility of two subgraphs using Weisfeiler-Lehman Kernel with stable matching is the significant contribution of this chapter.

Node-level prediction, link-level prediction, and graph-level prediction can be used for traditional learning when a graph is used as an input. This traditional learning has high precision: if the order feature vector is high. The future vector of high dimensionality will consume more computational power and time during using a graph with structured features as input. In Chapter 4, Instead of feature engineering, the graph representation of graph embedding is used for downstream predictions. Node embedding or graph embedding are used to map the graph or node structural features in low dimensional mapping space for further downstream predictions. The spectral clustering algorithm for entity resolution and data mining is also used in this Chapter.

The use of smart objects in social engineering is the concept of SIoT. Each smart object is a SIoT node in the graph. the importance of each node can be found by calculating its rank on the graph. The contribution of Chapter 5 is to calculate the Node/PageRank by using a link analysis algorithm- a step to find out how significant nodes in a graph are. The Social IoT Node/PageRank model can be obtained from its network flow dynamic which defines a stochastic Adjacency matrix used for flow equations in terms of random walk. The power iteration is used to converge the node traversal towards a final solution. Sometimes the node importance is to be calculated with a possibility of the secondary node navigation. This is addressed by Brin-Page Rank Equation. Here the random decision to jump from one node to another is taken. Brin-Page Rank Markov Chain is used for a recursive solution to converge the node traversal towards a final solution.

IoT smart objects are the nodes in SIoT that play different roles in the network formed by these smart devices. In Chapter 6, the modularity density measures are used to identify the different IoT nodes of the same nature as the network community. In a natural disaster management system, it is very important to identify, the node of high centrality. The deployment of an early warning system to a community of interest is the better solution compared to the whole network for an effective fast disaster management system.

REFERENCES

1. Arjunasamy and T. Ramasamy, "A proficient heuristic for selecting friends in social Internet of Things," 2016 10th International Conference on Intelligent Systems and Control (ISCO), 2016, pp. 1-5, doi: 10.1109/ISCO.2016.7727014.
2. Dipraj, S. Vishwakarma and J. Singh, "Social Internet of Things: The collaboration of Social Network and Internet of Things and its Future," 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), 2020, pp. 535-539, doi: 10.1109/ICACCCN51052.2020.9362847.
3. A. M. Esfahani, A. M. Rahmani and A. Khademzadeh, "MSIoT: Mobile Social Internet of Things, A New Paradigm," 2020 10th International Symposium on Telecommunications (IST), 2020, pp. 187-193, doi: 10.1109/IST50524.2020.9345837.
4. P. Dong, J. Ge, X. Wang and S. Guo, "Collaborative Edge Computing for Social Internet of Things: Applications, Solutions, and Challenges," in IEEE Transactions on Computational Social Systems, doi: 10.1109/TCSS.2021.3072693.
5. Vahdat-Nejad, Hamed, Zahra Mazhar-Farimani, and Arezoo Tavakolifar. "Social Internet of Things and New Generation Computing—A Survey." In Toward Social Internet of Things (SIoT): Enabling Technologies, Architectures and Applications, pp. 139-149. Springer, Cham, 2020.
6. Rad, Mozhgan Malekshahi, Amir Masoud Rahmani, Amir Sahafi, and Nooruldeen Nasih Qader. "Social Internet of Things: vision, challenges, and trends." Human-centric Computing and Information Sciences 10, no. 1 (2020): 1-40.
7. Fu, Cai, Chenchen Peng, Xiao-Yang Liu, Laurence T. Yang, Jia Yang, and Lansheng Han. "Search engine: The social relationship driving power of Internet of Things." Future Generation Computer Systems 92 (2019): 972-986.
8. Ahmad, Awais, Murad Khan, Anand Paul, Sadia Din, M. Mazhar Rathore, Gwanggil Jeon, and Gyu Sang Choi. "Toward modeling and optimization of features selection in Big Data based social Internet of Things." Future Generation Computer Systems 82 (2018): 715-726.

9. L. Atzori, A. Iera and G. Morabito, "SIoT: Giving a social structure to the Internet of things", IEEE Commun. Lett. vol. 15, no. 11, pp. 1193-1195, Nov. 2011.
10. Atzori, Luigi, Antonio Iera, Giacomo Morabito, and Michele Nitti. "The social internet of things (siot)–when social networks meet the internet of things: Concept, architecture and network characterization." Computer networks 56, no. 16 (2012): 3594-3608.
11. M. Nitti, R. Girau and L. Atzori, "Trustworthiness Management in the Social Internet of Things," in IEEE Transactions on Knowledge and Data Engineering, vol. 26, no. 5, pp. 1253-1266, May 2014, doi: 10.1109/TKDE.2013.105.
12. L. Ding, P. Shi and B. Liu, "The clustering of Internet of things and social network", Proc. 3rd Int. Symp. KAM, 2010.
13. J. Yu, K. Wang, P. Li, R. Xia, S. Guo and M. Guo, "Efficient Trustworthiness Management for Malicious User Detection in Big Data Collection," in IEEE Transactions on Big Data, doi: 10.1109/TBDATA.2017.2761386.
14. M. A. Azad, S. Bag, F. Hao and A. Shalaginov, "Decentralized Self-Enforcing Trust Management System for Social Internet of Things," in IEEE Internet of Things Journal, vol. 7, no. 4, pp. 2690-2703, April 2020, doi: 10.1109/JIOT.2019.2962282.
15. C. Boudagdigue, A. Benslimane, A. Kobbane and J. Liu, "Trust Management in Industrial Internet of Things," in IEEE Transactions on Information Forensics and Security, vol. 15, pp. 3667-3682, 2020, doi: 10.1109/TIFS.2020.2997179.
16. W. Z. Khan, Q. -u. -A. Arshad, S. Hakak, M. K. Khan and Saeed-Ur-Rehman, "Trust Management in Social Internet of Things: Architectures, Recent Advancements, and Future Challenges," in IEEE Internet of Things Journal, vol. 8, no. 10, pp. 7768-7788, 15 May15, 2021, doi: 10.1109/JIOT.2020.3039296.
17. Sherchan, Wanita, Surya Nepal, and Cecile Paris. "A survey of trust in social networks." ACM Computing Surveys (CSUR) 45, no. 4 (2013): 1-33.
18. Kourtellis, Nicolas, Tharaka Alahakoon, Ramanuja Simha, Adriana Iamnitchi, and Rahul Tripathi. "Identifying high betweenness centrality nodes in large social networks." Social Network Analysis and Mining 3, no. 4 (2013): 899-914.
19. Brandes, Ulrik. "A faster algorithm for betweenness centrality." Journal of mathematical sociology 25, no. 2 (2001): 163-177.
20. Abbasi, Alireza, Liaquat Hossain, and Loet Leydesdorff. "Betweenness centrality as a driver of preferential attachment in the evolution of research collaboration networks." Journal of Informetrics 6, no. 3 (2012): 403-412.

21. Geisberger, Robert, Peter Sanders, and Dominik Schultes. "Better approximation of betweenness centrality." In 2008 Proceedings of the Tenth Workshop on Algorithm Engineering and Experiments (ALENEX), pp. 90-100. Society for Industrial and Applied Mathematics, 2008.
22. Newman, Mark EJ. "A measure of betweenness centrality based on random walks." Social networks 27, no. 1 (2005): 39-54.
23. Alamsyah, Andry, and Budi Rahardjo. "Social network analysis taxonomy based on graph representation." arXiv preprint arXiv: 2102.08888 (2021).
24. S. Aalibagi, H. Mahyar, A. Movaghar and H. E. Stanley, "A Matrix Factorization Model for Hellinger-based Trust Management in Social Internet of Things," in IEEE Transactions on Dependable and Secure Computing, doi: 10.1109/TDSC.2021.3052953.
25. A. Zhao, M. Li and Q. Wang, "Cyber Security Risk Identification in Power Internet of Things System," 2020 IEEE 4th Conference on Energy Internet and Energy System Integration (EI2), 2020, pp. 777-781, doi: 10.1109/EI250167.2020.9346579.
26. N. Malik, S. K. Awasthi and N. Sood, "Centrality as a Friendship Selection Heuristic in Social Internet of Things," 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2020, pp. 1-6, doi: 10.1109/ICCCNT49239.2020.9225540.
27. Mehmood, Arif, Gyu Sang Choi, Otto F. von Feigenblatt, and Han Woo Park. "Proving ground for social network analysis in the emerging research area “Internet of Things” (IoT)." Scientometrics 109, no. 1 (2016): 185-201.
28. Riquelme, Fabián, Pablo Gonzalez-Cantergiani, Xavier Molinero, and Maria Serna. "Centrality measure in social networks based on linear threshold model." Knowledge-Based Systems 140 (2018): 92-102.
29. Borgatti, Stephen P. "Centrality and network flow." Social networks 27, no. 1 (2005): 55-71.
30. Davoudi, Anahita, and Mainak Chatterjee. "Social trust model for rating prediction in recommender systems: Effects of similarity, centrality, and social ties." Online Social Networks and Media 7 (2018): 1-11.
31. Amin, Farhan, and Gyu Sang Choi. "Hotspots Analysis Using Cyber-Physical-Social System for a Smart City." IEEE access 8 (2020): 122197-122209.

32. Daly, Elizabeth M., and Mads Haahr. "Social network analysis for routing in disconnected delay-tolerant manets." In Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing, pp. 32-40. 2007.
33. Niwattanakul, Suphakit, Jatsada Singthongchai, Ekkachai Naenudorn, and Supachanun Wanapu. "Using of Jaccard coefficient for keywords similarity." In Proceedings of the international multiconference of engineers and computer scientists, vol. 1, no. 6, pp. 380-384. 2013.
34. Aljubairy, Abdulwahab, Wei Emma Zhang, Quan Z. Sheng, and Ahoud Alhazmi. "SIoTPredict: A Framework for Predicting Relationships in the Social Internet of Things." In International Conference on Advanced Information Systems Engineering, pp. 101-116. Springer, Cham, 2020.
35. Kumari, Anisha, Ranjan Kumar Behera, Kshira Sagar Sahoo, Anand Nayyar, Ashish Kumar Luhach, and Satya Prakash Sahoo. "Supervised link prediction using structured-based feature extraction in social network." Concurrency and Computation: Practice and Experience (2020): e5839.
36. T. Qiu, M. Zhang, X. Liu, J. Liu, C. Chen and W. Zhao, "A Directed Edge Weight Prediction Model Using Decision Tree Ensembles in Industrial Internet of Things," in IEEE Transactions on Industrial Informatics, vol. 17, no. 3, pp. 2160-2168, March 2021, doi: 10.1109/TII.2020.2995766.
37. Kumar, Ajay, Shivansh Mishra, Shashank Sheshar Singh, Kuldeep Singh, and Bhaskar Biswas. "Link prediction in complex networks based on Significance of Higher-Order Path Index (SHOPI)." Physica A: Statistical Mechanics and its Applications 545 (2020): 123790.
38. Peng, Hao, Jianxin Li, Qiran Gong, Yuanxin Ning, Senzhang Wang, and Lifang He. "Motif-Matching Based Subgraph-Level Attentional Convolutional Network for Graph Classification." In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 04, pp. 5387-5394. 2020.
39. Ghanem, Hashem, Nicolas Keriven, and Nicolas Tremblay. "Fast Graph Kernel with Optical Random Features." arXiv preprint arXiv: 2010.08270 (2020).
40. P. Wongsriphisant, C. Lursinsap, A. Suratanee and K. Plaimas, "A classification of biochemical compounds based on their primitive structures and graph kernels," 2020 17th International Joint Conference on Computer Science and Software Engineering (JCSSE), 2020, pp. 104-109, doi: 10.1109/JCSSE49651.2020.9268317.

41. Goldberg, Yoav, and Omer Levy. "word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method." arXiv preprint arXiv:1402.3722 (2014).
42. Perozzi, Bryan, Rami Al-Rfou, and Steven Skiena. "Deepwalk: Online learning of social representations." In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 701-710. 2014.
43. Grover, Aditya, and Jure Leskovec. "node2vec: Scalable feature learning for networks." In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 855-864. 2016.
44. Dong, Yuxiao, Nitesh V. Chawla, and Ananthram Swami. "metapath2vec: Scalable representation learning for heterogeneous networks." In Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, pp. 135-144. 2017.
45. Abu-El-Haija, Sami, Bryan Perozzi, Rami Al-Rfou, and Alex Alemi. "Watch your step: Learning node embeddings via graph attention." arXiv preprint arXiv: 1710.09599 (2017).
46. Tang, Jian, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. "Line: Large-scale information network embedding." In Proceedings of the 24th international conference on World Wide Web, pp. 1067-1077. 2015.
47. Chen, Haochen, Bryan Perozzi, Yifan Hu, and Steven Skiena. "Harp: Hierarchical representation learning for networks." In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, no. 1. 2018.
48. Goyal, Palash, and Emilio Ferrara. "Graph embedding techniques, applications, and performance: A survey." Knowledge-Based Systems 151 (2018): 78-94.
49. Duvenaud, David, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. "Convolutional networks on graphs for learning molecular fingerprints." arXiv preprint arXiv: 1509.09292 (2015).
50. Li, Yujia, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. "Gated graph sequence neural networks." arXiv preprint arXiv: 1511.05493 (2015).
51. Ivanov, Sergey, and Evgeny Burnaev. "Anonymous walk embedding's." In International conference on machine learning, pp. 2186-2195. PMLR, 2018.
52. Cai, Hongyun, Vincent W. Zheng, and Kevin Chen-Chuan Chang. "A comprehensive survey of graph embedding: Problems, techniques, and applications." IEEE Transactions on Knowledge and Data Engineering 30, no. 9 (2018): 1616-1637.

53. Yue, Xiang, Zhen Wang, Jingong Huang, Srinivasan Parthasarathy, Soheil Moosavinasab, Yungui Huang, Simon M. Lin, Wen Zhang, Ping Zhang, and Huan Sun. "Graph embedding on biomedical networks: methods, applications and evaluations." Bioinformatics 36, no. 4 (2020): 1241-1251.
54. Savoy, Jacques. "Information retrieval on the web: a new paradigm." Upgrade 3, no. 3 (2002): 9-11.
55. Qiu, Jiezhong, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. "Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec." In Proceedings of the eleventh ACM international conference on web search and data mining, pp. 459-467. 2018.
56. Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." Computer 42, no. 8 (2009): 30-37.
57. Ghanbari, Yasser, Alex R. Smith, Robert T. Schultz, and Ragini Verma. "Identifying group discriminative and age regressive sub-networks from DTI-based connectivity via a unified framework of non-negative matrix factorization and graph embedding." Medical image analysis 18, no. 8 (2014): 1337-1348.
58. Pan, Shirui, Ruiqi Hu, Guodong Long, Jing Jiang, Lina Yao, and Chengqi Zhang. "Adversarially regularized graph autoencoder for graph embedding." arXiv preprint arXiv: 1802.04407 (2018).
59. Ou, Mingdong, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. "Asymmetric transitivity preserving graph embedding." In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 1105-1114. 2016.
60. Zhou, Chang, Yuqiong Liu, Xiaofei Liu, Zhongyi Liu, and Jun Gao. "Scalable graph embedding for asymmetric proximity." In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31, no. 1. 2017.
61. Zhang, Yuan, Dong Wang, and Yan Zhang. "Neural IR meets graph embedding: A ranking model for product search." In The World Wide Web Conference, pp. 2390-2400. 2019.
62. Chung, Fan. "The heat kernel as the PageRank of a graph." Proceedings of the National Academy of Sciences 104, no. 50 (2007): 19735-19740.
63. Hamilton, William L., Rex Ying, and Jure Leskovec. "Representation learning on graphs: Methods and applications." arXiv preprint arXiv: 1709.05584 (2017).

64. You, Jiaxuan, Zhitao Ying, and Jure Leskovec. "Design space for graph neural networks." Advances in Neural Information Processing Systems 33 (2020).
65. Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." In International conference on machine learning, pp. 448-456. PMLR, 2015.
66. Kipf, Thomas N., and Max Welling. "Semi-supervised classification with graph convolutional networks." arXiv preprint arXiv: 1609.02907 (2016).
67. Zhang, Si, Hanghang Tong, Jiejun Xu, and Ross Maciejewski. "Graph convolutional networks: a comprehensive review." Computational Social Networks 6, no. 1 (2019): 1-23.
68. Atwood, James, and Don Towsley. "Diffusion-convolutional neural networks." In Advances in neural information processing systems, pp. 1993-2001. 2016.
69. Defferrard, Michaël, Xavier Bresson, and Pierre Vandergheynst. "Convolutional neural networks on graphs with fast localized spectral filtering." arXiv preprint arXiv: 1606.09375 (2016).
70. Gori, Marco, Gabriele Monfardini, and Franco Scarselli. "A new model for learning in graph domains." In Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005. vol. 2, pp. 729-734. IEEE, 2005.
71. Niepert, Mathias, Mohamed Ahmed, and Konstantin Kutzkov. "Learning convolutional neural networks for graphs." In International conference on machine learning, pp. 2014-2023. PMLR, 2016.
72. Wu, Zonghan, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S. Yu Philip. "A comprehensive survey on graph neural networks." IEEE transactions on neural networks and learning systems (2020).
73. Barber, Michael J. "Modularity and community detection in bipartite networks." Physical Review E 76, no. 6 (2007): 066102.
74. Yang, Liang, Xiaochun Cao, Dongxiao He, Chuan Wang, Xiao Wang, and Weixiong Zhang. "Modularity Based Community Detection with Deep Learning." In IJCAI, vol. 16, pp. 2252-2258. 2016.
75. Shang, Ronghua, Jing Bai, Licheng Jiao, and Chao Jin. "Community detection based on modularity and an improved genetic algorithm." Physica A: Statistical Mechanics and its Applications 392, no. 5 (2013): 1215-1231.

76. Li, Shudong, Laiyuan Jiang, Xiaobo Wu, Weihong Han, Dawei Zhao, and Zhen Wang. "A weighted network community detection algorithm based on deep learning." Applied Mathematics and Computation 401 (2021): 126012.
77. Jin, Di, Binbin Zhang, Yue Song, Dongxiao He, Zhiyong Feng, Shizhan Chen, Weihao Li, and Katarzyna Musial. "ModMRF: A modularity-based Markov Random Field method for community detection." Neurocomputing 405 (2020): 218-228.
78. Pan, Yu, Jiupeng Yang, Shuaihui Wang, Junhua Zou, Guyu Hu, and Zhisong Pan. "Learning Deep Embedding for Community Detection." In 2020 International Conference on Big Data & Artificial Intelligence & Software Engineering (ICBASE), pp. 288-291. IEEE, 2020.
79. Chaudhary, Laxmi, and Buddha Singh. "Community detection using maximizing modularity and similarity measures in social networks." In Smart Systems and IoT: Innovations in Computing, pp. 197-206. Springer, Singapore, 2020.
80. Blondel, Vincent D., Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. "Fast unfolding of communities in large networks." Journal of statistical mechanics: theory and experiment 2008, no. 10 (2008): P10008.

LIST OF PUBLICATIONS

Journal Publications:

Published:

1. Rangra, Abhilasha, and Vivek Kumar Sehgal. "On Performance of Big Data Storage on Cloud Mechanics in Mobile Digital Healthcare," *International Journal of E-Health and Medical Communications (IJEHMC)* 12(5), (2021), doi:10.4018/IJEHMC.20210901.oa3

[ESCI/SCOPUS INDEXED]

2. Rangra, Abhilasha, Vivek Kumar Sehgal, and Shailendra Shukla. "A Novel Approach of Cloud Based Scheduling Using Deep-Learning Approach in E-Commerce Domain," International Journal of Information System Modeling and Design (IJISMD) 10(3), (2019), doi:10.4018/IJISMD.2019070104

[ESCI/SCOPUS INDEXED]

3. Rangra, Abhilasha, and Vivek Kumar Sehgal. " Natural Disasters Management Using Social Internet of Things" Multimedia Tools and Applications (Accepted) DOI : 10.1007/s11042-021-11486-8

[SCIE/SCOPUS INDEXED] [IF 2.7]

Under Review/Communicated

4. Rangra, Abhilasha, and Vivek Kumar Sehgal. " Markovian Embedding of high dimensional feature vector of Social Internet of Things" IEEE Transactions on Computational Social Systems (Under Communication)

[SCIE/SCOPUS INDEXED]

Book Chapters:

1. Rangra A., Kumar Sehgal V., Shukla S. (2019) Sentiment Analysis of Tweets by Convolution Neural Network with L1 and L2 Regularization. In: Luhach A., Singh D., Hsiung PA., Hawari K., Lingras P., Singh P. (eds) Advanced Informatics for Computing Research. ICAICR 2018. Communications in Computer and Information Science, vol 955. Springer, Singapore. https://doi.org/10.1007/978-981-13-3140-4_32

2. Rangra A., Sehgal V.K., Shukla S. (2019) Analysis and Impact of Trust and Recommendation in Social Network Based Algorithm for Delay Tolerant Network. In: Singh M., Gupta P., Tyagi V., Flusser J., Ören T., Kashyap R. (eds) Advances in Computing and Data Sciences. ICACDS 2019. Communications in Computer and Information Science, vol 1046. Springer, Singapore. https://doi.org/10.1007/978-981-13-9942-8_12

Paper presented in Conferences:

1. Rangra A., Kumar Sehgal V., Shukla S., “Sentiment Analysis of Tweets by Convolution Neural Network with L1 and L2 Regularization”. International Conference on Advanced Informatics for Computing Research. ICAICR 2018.
2. Rangra A., Sehgal V.K., Shukla S. “Analysis and Impact of Trust and Recommendation in Social Network Based Algorithm for Delay Tolerant Network” International Conference on Advances in Computing and Data Sciences. ICACDS 2019.