ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

NOVEL CENTRALITY, TOPOLOGY AND HIERARCHICAL-AWARE LINK PREDICTION IN DYNAMIC NETWORKS

Ph.D. THESIS

Abubakhari SSERWADDA

Department of Computer Engineering

Computer Engineering Programme

SEPTEMBER 2023

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

NOVEL CENTRALITY, TOPOLOGY AND HIERARCHICAL-AWARE LINK PREDICTION IN DYNAMIC NETWORKS

Ph.D. THESIS

Abubakhari SSERWADDA
(504182516)

Department of Computer Engineering

Computer Engineering Programme

Thesis Advisor: Assoc. Prof. Dr. Yusuf YASLAN
Thesis Co-Advisor: Asst. Prof. Dr. Alper ÖZCAN

SEPTEMBER 2023

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

DİNAMİK AĞLARDA MERKEZİLİK, TOPOLOJİ VE HİYERARŞİK TABANLI BAĞLANTİ TAHMİNİ

DOKTORA TEZİ

Abubakhari SSERWADDA
(504182516)

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Programı

Tez Danışmanı: Doç. Dr. Yusuf YASLAN
Eş Danışman: Dr. Öğr. Üyesi Alper ÖZCAN

EYLÜL 2023

Abubakhari SSERWADDA, a Ph.D. student of ITU Graduate School student ID 504182516, successfully defended the dissertation entitled "NOVEL CENTRALITY, TOPOLOGY AND HIERARCHICAL-AWARE LINK PREDICTION IN DYNAMIC NETWORKS", which he/she prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Assoc. Prof. Dr. Yusuf YASLAN** Istanbul Technical University	.....
Co-advisor :	**Asst. Prof. Dr. Alper ÖZCAN** Akdeniz University	.....
Jury Members :	**Asst. Prof. Dr. Berna KİRAZ** Fatih Sultan Mehmet University	.....
	Assoc. Prof. Dr. Ahmet Cüneyd TANTUĞ Istanbul Technical University	.....
	Prof. Dr. Şule GÜNDÜZ ÖĞÜDÜCÜ Istanbul Technical University	.....
	Prof. Dr. Hatice KÖSE Istanbul Technical University	.....
	Asst. Prof. Dr. Erkan USLU Yildiz Technical University	.....

Date of Submission : 04 July 2023
Date of Defense : 05 September 2023

To my spouse and children,

FOREWORD

I have been supported and encouraged by quite a number of organizations and individuals during my Ph.D. career worth mentioning. To begin with, I would like to thank "Turkiye Burslari", the Turkish Government Scholarship Board, for granting me the Ph.D. scholarship to facilitate my studies. Next, I extend warm gratitude to my supervisor Assoc. Prof. Yusuf YASLAN and the co-advisor Asst. Prof.Dr. Alper ÖZCAN, for guidance, advice, encouragement, constant supervision, and monitoring to ensure success in my studies. Your expertise and innovative ideas have been so vital in improving the quality of the work in my thesis. I appreciate your sacrifice in dedicating your precious time to several meetings to discuss my research and monitor my progress. Likewise, I eternally appreciate the jury members of my thesis committee, including Asst. Prof. Dr. Berna KIRAZ, who, on numerous occasions, has spared time to listen and advised me towards producing this vital work. I am extremely grateful to my family, especially my mother, Aisha NAMUKYAYA, for having laid a strong education foundation that has been a pillar for me to reach this remarkable success. In addition, I wholeheartedly appreciate Soroti University administration for having granted me study leave and support for the smooth running of my Ph.D. studies. Likewise, I am so grateful to the whole staff and administration of the faculty of Computer Engineering for the knowledge and expertise availed to me, in addition to the favorable learning atmosphere, including the laboratories and the proper Computer tools for practicing. Last but not least, I thank my lab group colleagues, especially Nurullah ATEŞ, for the wonderful cooperation and teamwork that have been key in realizing our goals.

September 2023

Abubakhari SSERWADDA
(researcher)

TABLE OF CONTENTS

Page

ABBREVIATIONS

AE	**:** Autoencoder
AUC	**:** Area Under the ROC Curve
PD	**:** Drug-Drug
GAT	**:** Graph Attentional Network
GAN	**:** Generative Adverserial Network
GCN	**:** Graph Convolutional network
GNN	**:** Graph Neural Network
GPU	**:** Graphics Processing Unit
HC-POSE	**:** Hierarchical and Centrality Aware POSE
LSTM	**:** Long Short-Term Memory
MSE	**:** Mean squared error
NLP	**:** Natural language processing
POSE	**:** Polypharmacy Side Effect
PD	**:** Protein-Drug
PP	**:** Protein-Protein
RGCN	**:** Relation Graph Convolutional network
STP-GCN	**:** Structural and Topology-Aware GCN
VAE	**:** Variational autoencoder

SYMBOLS

$\mathbf{A}_{ij}$	**:** Graph G's adjacency matrix $\in \mathbb{R}^{n \times n}$
$\widehat{\mathbf{A}}_{ij}$	**:** Graph G's adjacency matrix with self-loop $\in \mathbb{R}^{n \times n}$
λ	**:** Constant value
γ	**:** Constant value
$\mathbf{D}$	**:** Degree matrix with diagonal elements
$\mathbf{v}_i$	**:** Input node vector for the i^{th} node $\in \mathbb{R}^n$
$\widehat{\mathbf{v}}_i$	**:** Predicted vector for i^{th} input node $\mathbf{v}_i$, $\in \mathbb{R}^n$
$\mathbf{N}$	**:** Overall number of subjects in the population
$\mathbf{n}$	**:** Overall number of nodes in a graph
$\widehat{\mathbf{A}}_{ij}$	**:** Predicted adjacency matrix for node pair i and j $\in \mathbb{R}^{n \times n}$
tf	**:** Topological-similarity function
sf	**:** Strength centrality function
s	**:** Node strength value
$\mathbf{S}$	**:** Strength matrix
$\widetilde{\mathbf{v}}_i$	**:** Vector embedding for i^{th} input node $\mathbf{v}_i$, $\in \mathbb{R}^d$, where $d < n$

LIST OF TABLES

Page

LIST OF FIGURES

Page

NOVEL CENTRALITY, TOPOLOGY AND HIERARCHICAL-AWARE LINK PREDICTION IN DYNAMIC NETWORKS

SUMMARY

The increasing availability of social network data has given rise to research devoted to solving problems associated with social network-related applications. However, the hugeness and complexity of relationships among social network elements render the prediction of links between the entities a challenging task. The previous research often focuses primarily on investigating local node connectivity data while ignoring other important network-characterizing properties. The key network-characterizing properties that are often underrated include network topology, node structural centrality roles, and network hierarchical information. Furthermore, whereas many real-world graphs change over time, several works assume static networks. In order to overcome these challenges, first, we compute several topological similarity-based convolution feature matrices by using various topological similarity metrics such as Common Neighbour, Jaccard Index, Adamic Adder, Salton Index, Resource Allocation, and Sørensen Index. We then utilize the resulting topological feature matrices to capture the prevailing topological information in the input graphs efficiently. Second, we leverage the strength centrality, a stronger variant of node degree, to conserve the node's centrality and the structural connectivity information in the network. In addition, we systematically aggregate such diverse features to yield quality higher-level feature representations. Lastly, we leverage an LSTM layer to capture the prevailing temporal information in the graph sequences. To learn the low dimensional node representations, first, we deployed a fully connected variational autoencoder that efficiently explores variations in the input graphs to learn high-quality node embeddings. Furthermore, we imposed centrality and topological constraints on the learning model to further enforce the preservation of the centrality and topological ınformation of input graphs in the learned embeddings. However, variational autoencoders have large computational time and memory requirements due large number of parameters characterizing the fully connected encoders and decoders, especially when they are applied on large networks.

In order to extend our implementations to large datasets while minimizing the computational time and memory requirements, we adopted a Graph Convolution Network (GCN)-based implementation. The proposed Structural and Topological based geometric deep learning approach was evaluated on five real-world temporal social networks. Based on experimental results, on average, they yield a 4% link prediction AUC improvement in link prediction accuracy, a small increment in training for each epoch (0.2s (10%)), and a 56% MSE reduction in centrality prediction when compared to the best benchmarks. The proposed end-to-end centrality and

topological guided link prediction framework for dynamic networks preserve not only the centrality node roles and the topological information in the learned embeddings but also captures the prevailing temporal information in the dynamic networks. The models utilize node centrality and topological features to capture and conserve the network topology and the structural roles of nodes during embedding learning. Thus, obtaining pretty-quality embeddings that enhance the link prediction and centrality prediction accuracies.

For all our proposed methods, we assess the impact of the various modules of the proposed models by comparing them with their variants that lack such modules, and we present and explain the results accordingly.

In other related work, we introduce a Hierarchical and Centrality aware Polypharmacy Side Effect Prediction (HC-POSE) Model. We model side effect prediction as a link prediction task problem and leverage core decomposition to explore the prevailing hierarchical information in the heterogeneous protein-protein, protein-drug, and drug-drug interaction datasets. Following k-core decomposition, for each k-core subgraph produced, a node strength matrix is computed to store the centrality information of each subgraph. Then we systematically aggregate the obtained centrality with the k-core adjacency matrix to have higher-level diverse feature representations. We deployed a GCN-based auto-encoder to learn low-dimensional representations for the homogeneous sub-graphs and an RCGN-based auto-encoder for the heterogeneous subgraphs. Based on the experimental results, HC-POSE exhibited a 3% accuracy improvement in POSE prediction as compared to the best baseline. This is due to the ability of the model to efficiently capture the prevailing hierarchical and centrality information in such heterogeneous networks. We study datasets and propose better network property metrics to suit the structure of the datasets. For instance, datasets with high hierarchies, we propose hierarchical-based approaches to efficiently benefit from the structure of the data when embedding learning. To preserve the node's centrality and structural roles, we choose strength centrality, a powerful variant of the node degree centrality, since nodes with high degree centrality tend to have high scores in other centrality metrics (betweeness centrality, closness centrality and eigen centrality).

We demonstrated our proposed models' practical applicability and assessed their efficiency by implementing the models on real-world clinical polypharmacy side effect datasets, communication datasets, and social network datasets (Facebook, email messages, and stack overflow datasets).

The network property-based embedding models we propose can be extended to solve many other real-world problems. For instance, the proposed network topology and node centrality aware embedding learning models can be flexibly integrated with data augmentation models, including graph-generating networks and generative adversarial networks to synthesize data with desired network properties. This could solve data scarcity problems, especially for the scarce clinical datasets, by augmenting quality network property-aware brain connectomes, drug-drug interactions, and drug-disease interaction datasets. Moreover, we introduce parameters for regulating the importance attached to the conservation of each network property during embedding learning. This significantly helps in customizing the structure of the generated data by attaching more weight to the most desired network properties during embedding learning.

We highlight open opportunities and future works that can be adapted to add value to the proposed models in this dissertation. We discussed other network properties that can be explored so as to improve the quality of learned embeddings. Thus, improving the accuracy of the end network applications, such as link prediction, node classification, and graph clustering, whose accuracy depends on the quality of learned embeddings.

By following our approaches, we open directions and opportunities for exploring other centrality measures, more topological similarity metrics and other key network-characterizing properties including the modularity and diffusion coefficient, so as to enhance the quality of embeddings generated by geometric deep learning models.

DİNAMİK AĞLARDA MERKEZİLİK, TOPOLOJİ VE HİYERARŞİK TABANLI BAĞLANTİ TAHMİNİ

ÖZET

Sosyal ağ verilerinin artan kullanılabilirliği, sosyal ağ ile ilgili uygulamalara çözüm bulmayı amaçlayan araştırmaların ortaya çıkmasına neden olmuştur. Bununla birlikte, sosyal ağ elemanları arasındaki etkileşimlerin genişliği ve karmaşıklığı, elemanlar arasındaki ilişkilerin tahminini zorlu bir görev haline getirir. Önceki çalışmalar genellikle ağı tanımlayıcı diğer önemli özellikleri ihmal ederek yalnızca yerel düğüm bağlantı bilgilerini keşfetmeye odaklanmaktadır. Genellikle hafife alınan temel ağ karakterize edici özellikler arasında ağ topolojisi, düğüm yapısal merkezilik rolleri ve ağ hiyerarşik bilgileri bulunur. Ek olarak, çoğu çalışma; ağı statik olarak kabul eder, ancak bir çok gerçek dünya verisi zaman içinde değişmektedir. Bu çalışmada bu sınırlamaların üstesinden gelmek için, öncelikle Common Neighbour, Jaccard Index, Adamic Adder, Salton Index, Resource Allocation, and Sørensen Index gibi başarılı topolojik benzerlik ölçümlerini kullanarak çeşitli topolojik benzerlik tabanlı evrişim özellik matrisi hesaplanmaktadır. Ardından, girdi grafiklerinden temel topolojik bilgileri verimli bir şekilde yakalamak için ortaya çıkan topolojik özellik matrislerini kullanırız. İkinci olarak, ağ boyunca düğüm merkezilik rollerini ve düğümün yapısal bağlantı bilgisini korumak için düğüm derecesinin daha güçlü bir varyantı olan düğüm gücü merkeziyet matrisinden yararlanırız. Ek olarak, nitelikli üst düzey özellik gömmeleri elde etmek için bu tür çeşitli özellikleri sistematik olarak bir araya getiriyoruz. Son olarak, örtük ağ geçici bilgilerini keşfetmek için bir LSTM katmanı kullanıyoruz. Düşük boyutlu düğüm gömmelerini öğrenmek için, önce, yüksek kaliteli düğüm gömmelerine giriş grafiklerindeki varyasyonları etkili bir şekilde tespit eden tam bağlantılı bir varyasyonel otomatik kodlayıcı kullandık. Ayrıca, öğrenilen gömmelerde giriş grafiklerinin topolojik yapısının ve merkeziliğinin korunmasını daha da güçlendirmek için model öğrenimine topolojik benzerlik ve merkezilik kısıtlamalarını uyguladık. Ek olarak, nitelikli üst düzey özellik gömmeleri elde etmek için bu tür çeşitli özellikleri sistematik olarak bir araya getiriyoruz. Son olarak, örtük ağ geçici bilgilerini keşfetmek için bir LSTM katmanı kullanıyoruz. Düşük boyutlu düğüm gömmelerini öğrenmek için, önce, yüksek kaliteli düğüm gömmelerine giriş grafiklerindeki varyasyonları etkili bir şekilde tespit eden tam bağlantılı bir varyasyonel otomatik kodlayıcı kullandık. Ayrıca, öğrenilen gömmelerde giriş grafiklerinin topolojik yapısının ve merkeziliğinin korunmasını daha da güçlendirmek için model öğrenimine topolojik benzerlik ve merkezilik kısıtlamalarını uyguladık. Bununla birlikte, varyasyonel otomatik kodlayıcılar, özellikle büyük ağlarda doğrusal kodlayıcı ve kod çözücüleri tanımlayan çok sayıda parametre nedeniyle büyük hesaplama süresi ve bellek gereksinimlerine ihtiyaç duyar.

Hesaplama süresini ve bellek gereksinimlerini en aza indirirken uygulamalarımızı büyük veri kümelerine genişletmek için Grafik Evrişim Ağı (GCN) tabanlı bir uygulamayı benimsedik. Önerilen Yapısal ve Topolojik tabanlı geometrik derin öğrenme yaklaşımı, gerçek dünyadaki en fazla beş geçici sosyal ağ üzerinde test edilmiştir. Deneysel sonuçlara dayalı olarak, ortalama olarak, en iyi kıyaslamalarla karşılaştırıldığında, bağlantı tahmini doğruluğunda %4'lük bir bağlantı tahmini AUC iyileştirmesi, her epoch için ihmal edilebilir eğitim süresi artışı ve yapısal merkezilik tahmininde %56'lık bir MSE azalma sağlarlar. Zamansal ağ modelleri için önerilen uçtan uca topolojik benzerlik ve merkezilik farkında bağlantı tahmini, yalnızca öğrenilen yerleştirmelerdeki topolojik benzerlik ve merkezilik bilgisini korumakla kalmaz, aynı zamanda dinamik ağlarda örtük zamansal bilgiyi de keşfeder. Modeller, ağ topolojisinin korunmasını ve gömmeleri öğrenme sırasında düğümlerin yapısal merkezilik rolünü yakalamak ve uygulamak için topolojik benzerlik özelliklerini, düğüm merkezilik özelliklerini ve ilgili kısıtlamaları kullanır. Böylece, bağlantı tahmini ve merkezilik tahmini doğruluklarını artıran yüksek kaliteli yerleştirmeler elde edilir.

Diğer ilgili çalışmada, Hiyerarşik ve Merkeziyet odaklı bir Polifarmasi Yan Etki Tahmin Modeli (HC-POSE) tanıtıyoruz. Yan etki tahminini bir bağlantı tahmin görevi problemi olarak modelliyoruz ve heterojen protein-protein, protein-ilaç ve ilaç-ilaç etkileşimi veri kümelerindeki temel hiyerarşik bilgileri keşfetmek için çekirdek ayrışımından yararlanıyoruz. K-çekirdek ayrışımının ardından, ortaya çıkan her k-çekirdek için, her bir alt grafiğin merkezilik bilgisini depolamak için bir düğüm gücü matrisi hesaplanır; bu matris, daha yüksek seviye özellik gömmelerine sahip olmak için k-çekirdek bitişik matris ile sistematik olarak toplanır. Homojen alt grafikler için düşük boyutlu gösterimleri öğrenmek üzere GCN tabanlı bir otomatik kodlayıcı ve heterojen alt grafikler için RCGN tabanlı bir otomatik kodlayıcı kullandık. Deneysel sonuçlara dayalı olarak, HC-POSE, POSE tahmin doğruluğunda en iyi çalışmaya göre %3'lük bir gelişme sergiledi. Bunun nedeni, modelin bu tür heterojen ağlarda hakim olan hiyerarşik ve merkezi bilgileri verimli bir şekilde yakalama yeteneğidir.

Veri kümelerini inceliyoruz ve veri kümelerinin yapısına uyması için daha iyi ağ özelliği ölçümleri öneriyoruz. Örneğin, yüksek hiyerarşiye sahip veri kümelerinde, gömme vektörlerini öğrenme sırasında verilerin yapısından verimli bir şekilde yararlanmak için hiyerarşik tabanlı yaklaşımlar öneriyoruz. Düğümün merkeziliğini ve yapısal rollerini korumak için, düğüm derecesi merkeziliğinin güçlü bir varyantı olan kuvvet merkeziliğini seçiyoruz çünkü yüksek derecede merkeziliğe sahip düğümler, diğer merkezilik ölçütlerinde (arada olma merkeziliği, yakınlık merkeziliği ve öz merkeziliği) yüksek puanlara sahip olma eğilimindedir.

Önerilen modellerimizin pratik uygulanabilirliğini gösterdik ve modelleri gerçek dünyadaki klinik polifarmasi yan etki veri kümeleri, iletişim veri kümeleri ve sosyal ağ veri kümeleri (Facebook, e-posta mesajları ve yığın taşma veri kümeleri) üzerinde uygulayarak verimliliklerini değerlendirdik.

Önerdiğimiz ağ özelliğine dayalı gömme modelleri, diğer birçok gerçek dünya problemini çözmek için genişletilebilir. Örneğin, tanıttığımız ağ topolojisi ve düğüm merkeziliği yerleştirme öğrenme modelleri, verileri istenen ağ özellikleriyle sentezlemek için grafik üreten ağlar ve çekişmeli üretken ağlar dahil olmak üzere veri artırma modelleriyle esnek bir şekilde entegre edilebilir. Bu, kaliteli ağ özelliğine duyarlı beyin bağlantılarını, ilaç-ilaç etkileşimlerini ve ilaç-hastalık etkileşimi veri

kümelerini artırarak, özellikle kıt klinik veri kümeleri için veri kıtlığı sorunlarını çözebilir. Ayrıca, gömme öğrenme sırasında her bir ağ özelliğinin korunmasına verilen önemi düzenlemek için parametreler sunuyoruz. Bu, gömülü öğrenme sırasında en çok istenen ağ özelliklerine daha fazla ağırlık vererek oluşturulan verilerin yapısının özelleştirilmesine önemli ölçüde yardımcı olur.

Bu tezde önerilen modellere değer katmak için uyarlanabilecek açık fırsatları ve gelecekteki çalışmaları vurguluyoruz. Öğrenilen gömmelerin kalitesini artırmak için keşfedilebilecek diğer ağ özelliklerini tartıştık. Böylece, doğruluğu öğrenilen gömmelerin kalitesine bağlı olan bağlantı tahmini, düğüm sınıflandırması ve grafik kümeleme gibi uç ağ uygulamalarının doğruluğunu iyileştirmek

Yaklaşımımız sayesinde, geometrik derin öğrenme modelleri tarafından üretilen yerleştirmelerin kalitesini artırmak amacıyla modülerlik ve yayılma katsayısı dahil olmak üzere diğer merkezilik ölçülerini, topolojik benzerlik ölçümlerini ve diğer önemli ağ karakterize edici özellikleri keşfetmek için yönler ve fırsatlar yaratıyoruz.

1. INTRODUCTION

Recently, there has been a remarkably growing availability and size of social network data due to the rise in social network users and social network platforms, including WhatsApp, Facebook, Instagram, and TikTok. Hence, a motivation for research dedicated to solving social network challenges, including node classification [2–4], graph clustering [5, 6], and link prediction [7–10]. Link prediction, as a network application in particular, aims at discovering missing or future relationships (links) in the network based on the information gained from the network structure. Complex real-world interaction phenomena, including social networks and biological and chemical network datasets, are often modeled as graphs to ease the learning of their network structures. By doing so, it enables the application of the known graph theory techniques on such sophisticated interaction datasets, thus easing information extraction and decision-making on those datasets. Learning the role that the individual node plays in building the network structure is a crucial step in network analysis. Among the common metrics for examining the node roles include node centrality [11–15]. For instance, [16] proposes to assess a node's role in a social network using a normalized sum of its node degree, eigenvector centrality and betweenness centrality. Yet [17] deployed the node strength notion, a strong variant of node degree centrality, to conserve the centrality roles of brain regions of interest(RIO) while synthesizing the target brain connectome. In [18], The propagation of false information on WhatsApp was examined using the WhatsApp user's centrality score, and it was shown that users with a higher centrality frequently contribute the greatest extent to the coverage of societies in terms of content variety. Centrality was used for investigating the brain neural connections that are so important for information flow in the brain [19], used in the discovery of the crucial nodes in road transport networks to increase the safety of traffic systems [20], predicting the spread of diseases such as AIDS [21], track changes in temporal networks and predict unusual and suspicious behavior in networks including fraud [22] and many other applications [23–32].

On the other hand, numerous studies have emphasized the role of topological-based features in network link prediction problems. For instance, [33] proposes to use node similarity-based topological features as graph convolutional matrices for link prediction in complex biomedical networks. They comprehensively tested different topological measures and realized that topological-based GCN feature matrices yield better results as compared to the common Laplacian-based convolution feature matrices. On the other hand, SFCN [34] proposes the similarity-based future common neighbors model that efficiently learns to discover future common neighbors of nodes in a network based on topological similarity features. In addition, in [35], uses topological features to conduct link prediction in social networks, where, initially, five distinct similarity factors are used to determine how similar users are, and then a hybrid similarity criteria is deployed in the second step. Other related works include [36–39]

However, all the above-mentioned methods rely on the traditional graph theory-based engineered features, thus rendering them inefficient in analyzing large and complex networks. Recently, with inspiration from the overwhelming efficiency of deep learning in image processing and computer vision at large, many researchers have adapted deep learning-based approaches to graphs, a technique referred to as geometric deep learning [40–44]. However, different from computer vision and image processing, where image pixels reside on the Euclidean domain (i.e., the shortest distance between any two image pixels is obviously a straight line), graphs data reside on a non-Euclidean domain. Moreover, graph nodes normally have significantly varying sizes of neighbors. Some nodes might possess self-loops and have links with far-distant nodes. Such unique graph properties render deployments of the vital conventional deep learning operations to graphs non-obvious. To overcome these graph data challenges, as a pioneer study, [45] proposes a graph convolution network (GCN) layer that determines the node's new features by taking a weighted average of itself and its local neighbors. Following this landmark, many other variants of GCN have recently been introduced [46–52].

However, the GCN can only extract the immediate node connectivity information at the expense of the global and other key network property information. In addition, the original GCN was built to handle static networks, yet many real-world networks do

evolve with the emergence of new connections and the disappearance of some links over time.

In this thesis, in order to overcome the above-discussed challenges, firstly, we propose a novel Topological Similarity and Centrality aware Hybrid Deep Learning framework for Temporal Link Prediction, where we extract centrality and topological similarity-based features from the input graph adjacency matrices that are fed to the deep fully connected variational autoencoder architecture that learns the low dimensional representation. The framework is integrated with LSTM layers to capture the prevailing temporal information in the datasets. We impose the centrality and topological similarity-based constraints to enforce further preservation of the network topology and node centrality roles in the learned embeddings. The model exhibits an improvement in link prediction accuracy as well as in centrality prediction. However, the fully connected layers in variational autoencoders are characterized by many learnable parameters leading to overwhelming model growth. This renders the model inefficient for application on large-scale graphs. So as to handle large datasets while saving both the computer memory and the model training time, we adopted a GCN-based model termed Structural and Topological Similarity aware GCN (STP-GCN) for Link Prediction in dynamic networks, with a new loss function that utilizes the generalized GCN framework to leverage node strength centrality and topological similarity-based features so as to capture and preserve the node centrality roles and the network topological information.

1.1 Purpose of the Thesis.

In this thesis, we learn node embeddings by capturing and preserving the network property information, including centrality, topology, and hierarchical information, so as to improve the learned network embeddings' quality and, consequently, the link prediction accuracy. Owing to the fact that the efficiency of various end-network applications, including node classification, link prediction, and graph clustering, depends on the quality of learned node embeddings. The thesis unites the graph theory and deep learning approaches by modeling real-world interactions and social and biological network datasets as graphs and applying graph theory and deep-learning tools and algorithms to the data for better decision-making. Notably, this research provides approaches for utilizing deep learning frameworks to efficiently capture and preserve vital graph properties, including topological, centrality, and hierarchical network information. The research introduces efficient mechanisms for handling the subtasks leading to the learning of quality graph embeddings that preserve the desired network properties. First, logical means for extracting the desired network property-related features from the raw network interaction datasets are introduced. In this regard, methodologies for extracting topological, centrality, and hierarchical-based features from the raw network interaction datasets are introduced. Second, the research introduces a systematic procedure for aggregating the graph property-based features to meet the desired quality of embeddings. In addition, appropriate geometric-deep learning models for learning low-level representation for the network dataset are formulated. To further enforce the preservation of the desired graph properties in the learned network embeddings, the study establishes network property-constrained loss functions for optimizing the embedding learning frameworks, most importantly in an end-to-end manner to avoid error accumulation that characterizes subdivided tasks. More to this, this research explains the practical applicability of proposed network property-based graph embedding learning approaches in solving real-world problems, including quality data augmentation to solve the challenges of scarcity of data for training learning models, especially in clinical medical fields. Finally, the research handles complex heterogeneous polypharmacy side effect prediction tasks

by providing methodologies for systematically aggregating features from individual subgraphs of the heterogeneous network.

1.2 Contribution of the Thesis

This dissertation focuses on capturing and preserving vital graph properties while learning low-dimensional node representations so as to improve the quality of network embeddings and, thus, the efficiency of end network applications, including link prediction. The research unveils the vital underlooked graph property information while learning node embeddings. The properties explored in this research include network centrality, topological and hierarchical structures. Below we detail the contributions of this thesis.

1.2.1 Systematic mining of the topological and centrality-based features that suit deep learning frameworks from the raw interaction graph networks.

Methodologies for extracting topological features from the raw input network interaction datasets are introduced. The features are extracted in such a way that they can be conveniently fed into deep learning frameworks that are efficient at learning the low-dimensional representation of the dataset. In this regard, several feature matrices are computed based on conventional node topological similarity metrics, including common neighbor [53], Adamic-Adar [54], Jaccard Index [55], and Salton Index [56], Resource Allocation Index [57] and Sørensen Index [58]. In addition, this dissertation highlights the importance of capturing and preserving the node centrality roles in the learned network embeddings. To meet this goal, node strength centrality features are methodically computed from the input network interaction datasets, as further described in the next chapters of the thesis.

1.2.2 Flexibly modeling the conventional deep learning to attach more importance to preserving the desired network properties during graph node embedding learning.

To meet the targets of preserving the desired graph properties, this research establishes strategic methodologies for modeling the convectional deep learning algorithms, including the fully connected autoencoders and graph convolutional networks (GCNs), to emphasize preserving of the desired network properties in the learned embeddings.

1.2.3 Network property-based embedding learning constraints.

The dissertation formulates several graph property-based learning constraints, including the topological similarity and node strength-centrality, that are imposed on the learning model to further guide the model towards preserving the target network property during embedding learning. in addition, this research establishes mechanisms for regulating the importance/weight attached to the target network property while learning model. In addition, model cost functions are built and optimized intuitively with the goal of preserving the desired property in the learned embeddings.

1.2.4 Clearing directions on generalizing geometric deep learning models to flexibly adapt to preserving other vital graph properties when learning network low representations.

This research gives guidance to readers on how other graph-based properties, including modularity and diffusion coefficient, could be swiftly integrated with powerful geometric deep-learning tools to improve the quality of learned embeddings. This, in turn, improves the quality of network end applications, including link prediction, node classification, node recommendation, and graph clustering.

1.2.5 Elaborating the practicality of proposed models in solving other crucial real-world challenges, including augmentation of the scarce clinical medical data.

The dissertation establishes how the proposed solutions can be adapted to solve other crucial real-world challenges, such as data scarcity in the clinical medical field. These topology, centrality, and hierarchical-based embedding models can easily be integrated with data-generating models like Generative Adversarial Networks (GANs) and Graph Translation Networks to synthesize high-quality network-property-aware datasets that meet the needs of many fields where data is scarce. The generated data could also boost learning models' efficiency in making sound decisions on numerous real-world phenomena. Moreover, the thesis establishes means for regulating the weight attached to the conservation of the desired network work property during embedding learning. This is an efficient way of guiding data augmenting models to customize the generated data so that it meets the set target for which it is synthesized.

1.3 Thesis Outline.

The remaining part of the thesis is structured as follows. In Chapter 2, graph properties, including centrality, topology, and hierarchical information, are discussed. The various measures of node centrality are presented, together with their roles in defining the network structure. We also describe the topological similarity measures and the methodologies for the computation of related feature matrices from these similarity measures. The chapter also reveals how to capture the underlying hierarchical information in the network using with k-core decomposition tool. Chapter 3 introduces the recent deep-learning approaches for link prediction, including fully connected neural networks and the Graph Convolutional Network (GCN). Chapter 4 discusses the integration of the topology and centrality-based network features with geometric deep learning architectures, including GCN and GNNs. In this chapter, we propose several novel graph-property-based embedding learning frameworks. First, a Structural and Topological aware GCN for Link Prediction in Temporal networks (STP-GCN) is proposed to incorporate preserving explicitly the network topology and structural roles of nodes in a network. With the goal of improving link prediction accuracy, a Topological Similarity and Centrality aware Hybrid Deep Learning for a Temporal Link Prediction model is proposed that relies on the power of the variational auto-encoder to efficiently capture the variations in the input graph patterns so as to improve the quality of learned embeddings. In addition, to further guarantee that the network topology and node centrality role of the input graphs is preserved in the learned embeddings, we impose topological similarity and node strength centrality constraints on the model learning. Further, we assign special learning hyperparameters for regulating the weight attached to the topological and centrality properties while learning node embeddings. Chapter 5 extends the network-property-based embedding learning to complex heterogeneous network datasets characterized by different types of networks and/or nodes. The section introduces unique feature aggregation schemes and deep learning architectures for modeling embedding learning in such unique types of networks. In this regard, centrality and hierarchical network-based properties were chosen to best benefit from the structure of the heterogeneous polypharmacy side effect (POSE) dataset set that was used for implementing the models. The polyphamarcy

side effect prediction is modeled as link prediction problem. Then, a Hierarchical and Centrality-guided Polypharmacy Side Effect Prediction Model (HC-POSE) is proposed that leverages k-core decomposition to systematically capture the prevailing hierarchical information in the complex, heterogeneous polypharmacy datasets. The model intuitively aggregates hierarchical and centrality-based rich features that are then fed to the geometric deep-learning framework to learn hierarchically and structurally sound embeddings for drug-drug interaction prediction. HC-POSE is compared with related models in the literature, and the results are presented. Chapter 6 presents concluding remarks and discusses the feasible directions for future related work. Finally, additional information on several mathematical models adopted, experiments performed, and the result analysis is provided in the Appendices section.

2. OVERVIEW OF GRAPH PROPERTIES; CENTRALITY, TOPOLOGICAL-SIMILARITY, and HIERARCHICAL INFORMATION.

2.1 Node Centrality.

The key step in network analysis is to monitor how connections vary between any pairs of nodes. This can be done by computing node degree, a measure that quantifies the number of links that connect each node with the rest of the network nodes. For instance, people with large followers (large degrees) are often the most influential on the social network. Yet, in communication networks, such highly connected nodes are the hubs for spreading information throughout the network.

2.1.1 Node degree.

The node degree for an undirected network is computed as the total number of links incident to the node. The degree k for a node i a network with other nodes $j = 1...N-1$ is computed as:

$$k_i = \sum_j A_{ij}, \tag{2.1}$$

where A denotes the adjacency matrix.

2.1.2 Node strength.

The node strength is analogous to the node degree for a weighted network. In an undirected network, the node strength measures the connectivity weights of the links connected to each node. The node strength s_i of a node i is computed as in the formula below:

$$s_i = \sum_j \mathbf{W}_{ij}. \tag{2.2}$$

where, W_{ij} represents the weight for the edge between nodes i and j. By adding up all edge weights associated with the reference node [59], the node strength quantifies the node's global connection. Below we give a simple example to clearly differentiate the

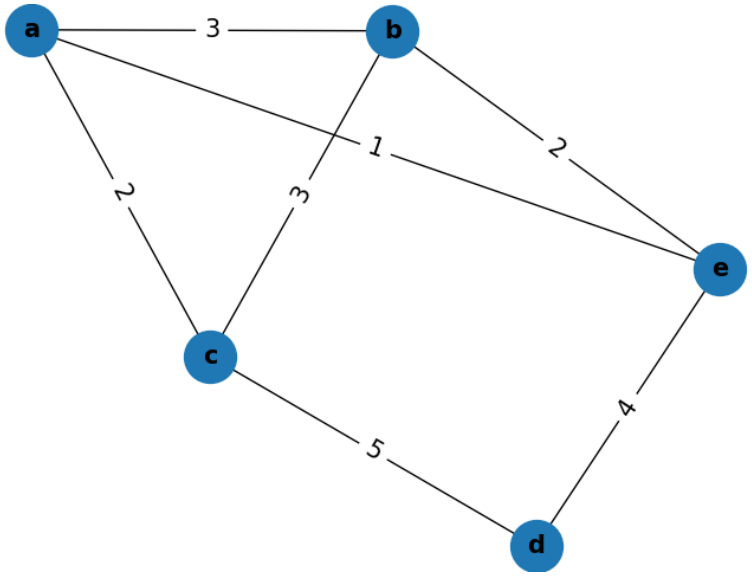

Figure 2.1 : A sample undirected weighted graph.

node degree and node strength, given a weighted adjacency matrix A for a graph G, with nodes a,b,c,d, and e.

$$A=\begin{array}{c} \begin{matrix} a & b & c & d & e \end{matrix} \\ \begin{pmatrix} 0 & 3 & 2 & 0 & 1 \\ 3 & 0 & 3 & 0 & 2 \\ 2 & 3 & 0 & 5 & 0 \\ 0 & 0 & 5 & 0 & 4 \\ 1 & 2 & 0 & 4 & 0 \end{pmatrix} \end{array}$$

From Fig. 2.1, graph nodes *a*,*b*,*c*, and *e* all have the same node degree, which is 3. However, they have varying node strength scores, i.e., 6, 8, 10, and 7, respectively.

2.2 Topological Similarity.

Topological similarities metrics are quantitative measurements used to evaluate the similarity of network structures between nodes. These metrics quantify the degree of similarity between connectivity patterns, relationships, and other network properties. In the research, we utilized topological similarity metrics to extract topological features from the input graph shots. Below we describe some of the topological measures and how we extracted input features based on these measures. Considering A = $[a_{uv}]_{N\times N}$, as an adjacent matrix for graph G, $\Gamma(u)$, a set of neighbors for vertex(u), q as the similarity score for vertices u and v. k_u denotes the degree for node u. The topological similarity metrics are categorized into local, quasi-local and global ones. Between global and local methods are quasi-local ones.

2.2.1 Local metrics

Local metrics focus on the immediate neighborhood properties of the reference pair node pairs. Below we detail the commonly used local topological similarity methods.

2.2.1.1 Common neighbor

Common neighbor [53] considers any two vertices that share links with the same group of other vertices to have a high likelihood of being connected. It quantifies the number of shared neighbors between two nodes and measures the degree of overlap between their connectivity patterns. Below are the procedures for common neighbor similarity computation proceedures between nodes u and v.

1. Determine the set of neighbors for node u, $\Gamma(u)$
2. Determine the set of neighbors for node v, $\Gamma(v)$
3. Compute the number of similar neighbors between nodes u and v.
4. Normalize the result by dividing the number of shared neighbors by the number of total neighbors for node A or node B. This step is optional. The normalization serves to account for variations in node degree and renders the measure comparable to different network structures.

Below in Eq.2.3 is the formula for computing common neighbor.

$$q_{uv} = |\Gamma(u) \cap \Gamma(v)|. \tag{2.3}$$

In a matrix form, Common Neighbor can be formulated as

$$q_{uv} = (A^2)_{uv}. \tag{2.4}$$

2.2.1.2 Jaccard index

The Jaccard Index [55] calculates the proportion of common neighbors in relation to all neighbors. Hence, it measures the degree of overlap between two nodes' neighbors. It attains the maximum score when both vertices share similar neighbors (i.e., $\Gamma(u) = \Gamma(v)$). Listed below are the step by step procedures for computing the Jaccard index for two nodes, u and v, for in a graph.

1. Determine the set of neighbors for node u, $\Gamma(u)$
2. Determine the set of neighbors for node v, $\Gamma(v)$
3. Compute the ıntersection of the two sets $\Gamma(u)$ and $\Gamma(v)$. (i.e., the number of similar neighbors between nodes u and v).
4. Obtain the Jaccard Index by dividing the intersection by the union.

Eq.2.5 shows the formula of the computation of the Jaccard Index.

$$q_{uv} = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u) \cup \Gamma(v)|}. \tag{2.5}$$

In a matrix form, it can be expressed as

$$q_{uv} = A^2_{uv} \oslash (A_{uv}M + MA_{uv} - A^2_{uv}). \tag{2.6}$$

M indexes an all-ones matrix with identical to A,

2.2.1.3 Salton index (cosine similarity)

The Salton Index [56] computes the cosine angle between vectors of the adjacency matrix that correspond to the reference vertices. (i.e., it computes the size of overlap based on the degrees of the vertices). Below are the set by step procedures for computing the Cosine Similarity of two nodes u and u according to their feature vectors.

- Represent the features of nodes u and v as vectors, with each dimension representing a distinct feature.
- Determine the dot product of the two feature vectors, whose value is the sum of the element-by-element multiplication of respective dimensions.
- Determine the magnitude (Euclidean norm) of each feature vector, that is the square root of the sum of its dimensions' squared values.
- The two feature vectors' magnitudes are multiplied.
- The Cosine Similarity is obtained by dividing the dot product by the product of the magnitudes.

The Formula in Eq.2.7 illustrates the computation of the Cosine Similarity.

$$q_{uv} = \frac{|\Gamma(u) \cap \Gamma(v)|}{\sqrt{k_u \times k_v}}. \tag{2.7}$$

where k_u is the degree of node u.

In a matrix form, it can be expressed as:

$$q_{uv} = A^2_{uv} \oslash D.WhereDisthedegreematrix. \tag{2.8}$$

2.2.1.4 Sørensen index

The Sorensen index [58] (Dice similarity coefficient), computes the size of an intersection of the neighbors of the reference vertices, much like the Jaccard Index. It quantifies the proportion of neighbors shared by both nodes. It provides a measure of similarity determined by the size of the intersection in relation to the size of the union. Below we detail steps for computing the Sørensen index.

1. Determine the set of neighbors for node u, $\Gamma(u)$

2. Determine the set of neighbors for node v, $\Gamma(v)$

3. Compute the intersection size of the two sets $\Gamma(u)$ and $\Gamma(v)$.

4. Divide the intersection size by the union size.

Below 2.9 is the formulae for computing the Sørensen index.

$$q_{uv} = \frac{2|\Gamma(u) \cap \Gamma(v)|}{k_u + k_v}. \tag{2.9}$$

The resultant Sørensen index ranges from 0 to 1, where 0 represents no similarity (no shared neighbors) and 1 represents maximal similarity (all neighbors are shared). It has been extensively used in numerous applications, such as clustering, document similarity and image analysis, to determine the degree of similarity or dissimilarity between entities. In a matrix form, it can be expressed as:

$$q_{uv} = 2A^2 \oslash (AM + MA) \tag{2.10}$$

2.2.1.5 Adamic–adar index.

The Adamic-Adar [54] computes common neighbors by allocating weights to vertices inversely proportionate to their degrees. Thus, a common neighbor that is exclusive to a few vertices is considered to be more important as compared to a hub. It determines the extent of "rarity" of common neighbors by allocating greater weights to those with lower connections in the network. Below we present steps for computing the Adamic-Adar index topological similarity index:.

1. Determine the set of neighbors for node u, $\Gamma(u)$

2. Determine the set of neighbors for node v, $\Gamma(v)$

3. Obtain the Adamic-Adar index by summing the reciprocals of the degrees of node u and node v's common neighbors.

4. Assign greater weights to neighbors who have lower degrees, since they are more relevant for determining similarity.

It is expressed as:

$$q_{uv} = \sum_{z \in \Gamma(i) \cap \Gamma(j)} \frac{1}{\log k_z}. \tag{2.11}$$

Below is its matrix representation.

$$q_{uv} = Alog(D^{-1})A \tag{2.12}$$

2.2.1.6 Hub-promoted index.

The Hub-promoted index (HPI) [60] allocates large scores to edges that are connected to hubs (nodes with high degrees). Intuition; Nodes that shares links with hubs are like to have high similarity. Note that the denominator selects the minimum degree of the reference vertices. Below we summarize the procedures for computing the Hub Promoted Index over nodes u and v.

1. Determine the network's hubs (nodes with large degree scores)
2. Obtain the neighboring node sets $\Gamma(u)$ and $\Gamma(v)$ for node u, and v respectively.
3. Compute the intersection of the hub-connected neighbors of u and the hub-connected neighbors of v.
4. The intersection is normalized by dividing it with the minimal number of node u and v's neighbors

The result is of the above procedures is the Hub-Promoted Index for nodes u and v. Higher values reflect a greater level of similarity between nodes based on their common neighbors linked to hubs. Below are the equations for computing the Hub-Promoted Index .

$$q_{uv} = \frac{|\Gamma(u) \cap \Gamma(v)|}{\min\{k_x, k_y\}}. \tag{2.13}$$

In the matrix form it can be represented as:

$$q_{uv} = A^2_{uv} \oslash min\{A_{uv}N, NA_{uv}\}. WhereN is a matrix of all ones. \tag{2.14}$$

2.2.1.7 Hub-depressed index.

In contrast to the Hub Promoted Index, Hub-Depressed Index (HDI) [60] allocates smaller scores to edges connected to hubs. Rich neighborhoods are penalized. It is computed as shown below:

$$q_{uv} = \frac{|\Gamma(u) \cap \Gamma(v)|}{\max\{k_x, k_y\}}. \tag{2.15}$$

In the matrix form it is computed as:

$$q_{uv} = A^2_{uv} \oslash max\{A_{uv}N, NA_{uv}\}. \tag{2.16}$$

2.2.1.8 Resource allocation index.

Inspired by the resource allocation processs, the Resource Allocation metric [6] quantifies the amount of the resource conveyed between nodes u and v. It is computed as.

$$q_{uv} = \sum_{z \in \Gamma(u) \cap \Gamma(v)} \frac{1}{k_z}. \tag{2.17}$$

2.2.2 Global metrics

Global metrics utilize the all underlying information throughout the entire graph.

2.2.2.1 Katz index

The Katz Index [61], assess similarity between any pair of nodes in the network by examining the total number of paths between the node pairs and the corresponding lengths of these paths. It allocates higher scores to shorter paths or higher similarity ratings to node pairs that are connected directly. On the otherside, longer paths are penalized exponentially. Below we present the steps for computing the Katz index for any network nodes u and v.

1. Allocate a positive weight,α, to the direct links between nodes. α, varies across 0 and 1.
2. Calculate the Katz index by adding the weighed contributions of all paths across nodes u and v, while taking each path's length into consideration.
3. Each path's contribution is calculated by multiplying the weights of each link along the path.

Below we present the formula for computing the Katz index.

$$q_{uv} = \sum_{l=1}^{\infty} \beta^l |paths_{uv}^{<l>}|, \tag{2.18}$$

Where l denotes the path length, β is an arbitrary parameter. The sum converges if β is less than the reciprocal of adjacency matrix's greatest eigenvalue. The weights drop exponentially with increasing path length. The matrix form of Katz Index could be represented as:

$$q_{uv} = (I - \beta A)^{-1} - I. \tag{2.19}$$

2.2.2.2 Average commute time

The Average commute time (ACT) measures the average travel time between two nodes within a network.

$$s_{uv} = \frac{1}{n(u,v)} = \frac{1}{m(u,v) + m(u,v)}, \tag{2.20}$$

where, $m(x,y)$ denotes the average number of steps a random walker must take to reach v from u. Any node pairs are more similar when they are in close proximity and have a short commute time. The average commute time could be calculated by solving a set of linear equations. Average Commute Time can be calculated using the pseudoinverse of the Laplacian matrix, L as follows:

$$n_{uv} = 2M(l^{+}_{uu} + l^{+}_{vv} - 2l^{+}_{uv}), \tag{2.21}$$

where $l^{+}_{uv} = [L^{+}]_{uv}$.

M denotes the total number of edges.

The pseudoinverse L^{+} of the Laplacian matrix computed as in Eq. 2.22 [62].

$$L^{+} = \left(L - \frac{ee^{T}}{n}\right)^{-1} + \frac{ee^{T}}{n}, \tag{2.22}$$

e is a column vector consisting only of ones (1's).

2.2.2.3 Normalized average commute time

This is a version of the Average Commute Time that considers node degrees, $m(u,v)$ is normally very small for hubs (high–degree nodes) regardless of u. Below we present its formulation.

$$q_{uv} = \frac{1}{(m(u,v)\pi_{v} + m(u,v)\pi_{u})}, \tag{2.23}$$

where π is a stationary Markov chain distribution characterizing a random walker on a connected graph.

$$\pi(x) = \frac{k_x}{\sum_y k_y}. \tag{2.24}$$

2.2.2.4 Random walk with restart (RWR)

The Random walk with restart (RWR) utilizes the concept of random walks to calculate the probability of moving from one network node to another, considering both direct and indirect relationships. It adopts the page rank algorithm [63]. Random Walk with Restart (RWR) takes network connectivity into consideration by assigning greater probabilities to network nodes that are more accessible or reachable from the source node. It provides a method for estimating the probability of reaching a node via random paths, thereby capturing the structural connection among the network nodes. Below are the steps for computing topological similarity of two nodes with random walk technique.

1. Begin at a specified source node in a network.
2. Allocate a starting probability distribution to all network nodes. The source node is usually given a probability of 1, while all other nodes are allocated a probability of 0.
3. Execute random walks strating from the source node by propagating probabilities across edges to neighboring nodes in an iterative manner.
4. The likelihood of transitioning to a neighbor at each step is computed from a combination of the network structure and the current probability distribution. This can be achieved by computing transition probabilities determined by the adjacency matrix or through deploying other measures like the personalized PageRank.
5. (Optional), provide a restart probability allowing the random walk to restart from the source node with a particular probability at each step.
6. Repeat the random walk procedure until convergence or until an exact number of steps has been reached.
7. The resultant probabilities can be used as a measure for topological similarity between any two nodes. Nodes that have higher probabilities indicate a higher topological similarity level to the source node.

Below we provide expressions for computing the Random Walk with Restart Index. Assume a random walker that begins at node u and returns periodically, with probability α, to u. Let z_u be a stationary Markov chain distribution characterizing this walker.

$$z_u = (1-\alpha)P^T z_u + \alpha e_u, \tag{2.25}$$

where, e_u represents a unit vector with the value 1 at the location which corresponds to node u.

P denotes a transitional matrix characterizing the common random walker.

$P_{uv} = 1/k_u$ if $A_{uv} = 1$, else 0. Below is the joint solution for all the nodes.

$$z = [z_1|z_2|\ldots|z_n] = \alpha(I-(1-\alpha)P^T)^{-1}. \tag{2.26}$$

To attain symmetry, the RWR similarity index is expressed as:

$$q_{uv} = z_{uv} + z_{uv}. \tag{2.27}$$

2.2.2.5 Matrix forest index

The Matrix Forest Index (MFI) [64], measures the structural similarity of two nodes depending on their connectivity characteristics. It is the proportion of the number of spanning rooted forests in which nodes u and v belong to the same tree rooted at node u to the total number of spanning rooted forests in the network. It stems from the theorem of matrix forest, which provides a method for determining the total number of spanning trees in a graph. The procedures for computing the Matrix Forest Index are as described below;

1. Consider a graph with two nodes u and v and adjacency matrix A
2. Create subgraphs by exploring nodes u and v neighbors. A node's neighborhood consists of the node its self and all its neighbors that are directly connected to it. Taking $U^{'}$ and $V^{'}$ to represent subgraghs for nodes u and v respectively. Let $U^{'}A$ and $V^{'}A$ denote the adjacency matrices for these subgraphs.
3. Compute the subgraphs' Laplacian matrices and the exponential matrices of the Laplacian matrices.

Below we present the matrix formulation of the Matrix Forest Index.

$$q = (I + L)^{-1}. \tag{2.28}$$

2.2.2.6 Cosine based on laplacian pseudoinverse (L^{+})

This measure captures the underlying structural similarities between networks, with an emphasis on connectivity patterns. Below are the basic steps for computing the cosine similarity based on the Laplacian matrix.

1. Determine the graph's Laplacian matrix
2. Perform normalization for the Laplacian matrix. Each element of the Laplacian matrix L is divided by a square root of the product of elements in the main diagonal. This step guarantees that each row in the Laplacian matrix is unit norm.
3. Compute the cosine similarity of any two Laplacian matrices.

The computation of this metric is shown in Eq. 2.29 below.

$$q_{uv} = \frac{l^{+}_{uv}}{\sqrt{l^{+}_{uu} l^{+}_{vv}}} \tag{2.29}$$

The cosine similarity will return a scalar value across -1 and 1, where 1 represents maximum similarity, 0 represents no similarity, and -1 represents maximum dissimilarity.

2.3 Graph Hierarchical Information

Graph hierarchical information defines to the stacked arrangement or layout of a graph. It entails the notion of levels or layers of the graph, in which nodes are organized hierarchically according to their relationships or features. Graph theory offers a variety of schemes for studying and analyzing the underlying hierarchical information in networks. Some of these techniques include the k-core decomposition. It reveals concentrated subgraphs as well as connectivity hierarchies within a graph.

Motivated by the efficiency of k-core decomposition in capturing graph hierarchical information [65–72], we leverage K-core decomposition on the adjacency matrix of the input graphs to extract the underlying hierarchical information in the graphs. A k-core of a graph is the largest unique subgraph whose nodes all have a degree size greater or equal to k.

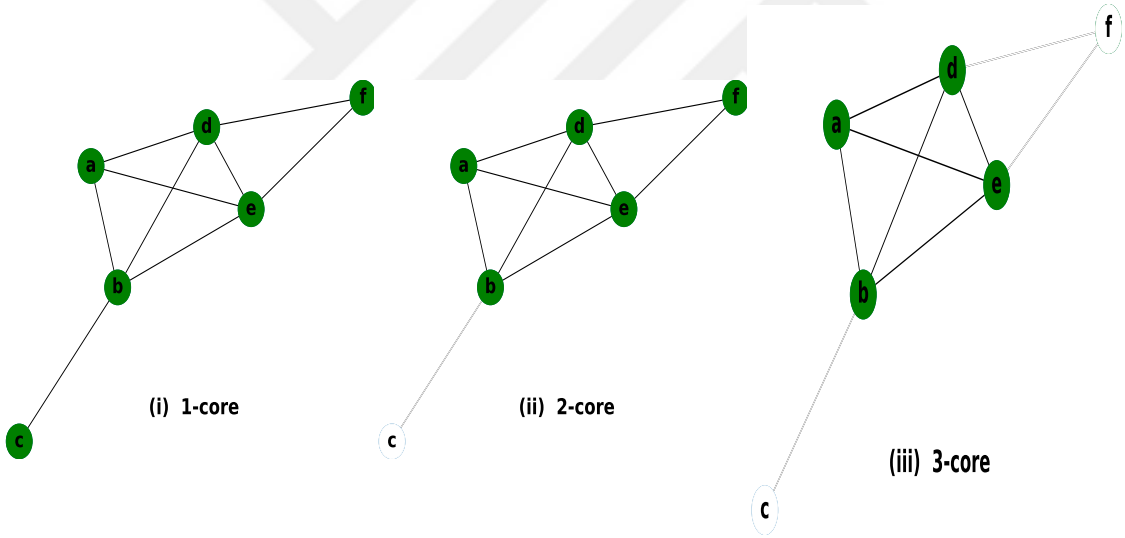

Figure 2.2 : For a sample graph G with 6 nodes, and 9 edges, its 1-core, 2-core, and 3-core are shown on the left, center, and right respectively. The nodes marked in green color and edges marked in dark black constitute the k-core in any case.

Following Fig 1, by repeatedly deleting nodes with degrees less than k, the k-core algorithm progressively reveals the denser core structure of the graph by eliminating away the lesser densely connected regions. Hence, the algorithm reveals the hierarchy of connectivity information throughout the graph, with larger k values indicating more

densely connected subgraphs. The steps below summarize the flow of the k-core decomposition algorithm.

1. Consider the initial raw graph as the the starting point,
2. Identify nodes with less than k degrees.
3. Delete the identified nodes and their linked edges from the graph.
4. Update the node degrees of the graph's remaining nodes by deducting the number of deleted edges incident on those nodes.
5. Repeat procedures 2-4 until there are no remaining nodes with a degree less than k.
6. Upon termination of the algorithm, the resulting graph is the k-core subgraph of the initial graph.

Below 1 is the pseudocode for the k-core decomposition algorithm. 1.

Algorithm 1: Pseudocode for k-core decomposition.

Input: $\mathbf{G} = (V, E)$

```
1  Calculate the node degrees Eq.2.1 ;
2  Arrange the graph node-set in ascending order of their node degrees ;
3  for each v ∈ V do
4      core[v] = degree[v] ;
5      for each u ∈Neighbours(v) do
6          if degrees[u] > degree[v] then
7              degrees[u] = degree[u]-1 ;
8              update the ordering of the nodes V ;
9          end
10     end
11 end
```

Output: A table core storing a core number for every node.

3. OVERVIEW OF GRAPH EMBEDDING LEARNING APPROACHES.

This chapter discusses the graph embedding approaches as per the literature. Generally, these embedding approaches are grouped into; random walk, matrix factorization, and deep learning-based. Below we present these methods in detail.

3.1 Matrix Factorization

Matrix factorization-based methods model a graph as a matrix, and then the matrix is decomposed to yield low-level node representations (embeddings). For instance, in RANMF [73], a regularized asymmetric non-negative matrix factorization learns graph embeddings by capturing pairwise similarities between node feature vectors. Several variants of non-negative matrix factorization are discussed in [74–76]. Yet, [77] learns temporal graph embeddings by negative matrix factorization method that leverages incremental and local block-coordinate gradient descent techniques. In addition, LIST [78] introduces a matrix factorization approach that learns graph node low representations by maximizing the reconstruction and network propagation errors. Furthermore, IsoMap [79] utilizes multidimensional scaling (MD) and shortest distance matrix to capture the network structure of manifolds in the embedding space. Other matrix factorization graph embedding approaches include [46, 80–82]

However, the matrix factorization-based methodologies are associated with large computer memory and time requirements. Furthermore, the target matrix's linear decomposability, as an assumption put forward by matrix factorization techniques, might not always hold true.

3.2 Random Walk.

Inspired by the shortcomings of matrix factorization, researchers leverage random walks to learn node representations in the low dimensional space. Here a graph is traversed with random paths and represented as per the frequency of the random walks

observed. Given a graph G, with the i^{th} path as a triplet $< p_i^o, p_i^d, l_i >$, where p_i^o and p_i^d indexes the origin and destination nodes of the paths respectively and l_i for the path length. Graph G is then represented as a d dimensional vector with the i^{th} element indexing the average frequency of G's i^{th} triplet. For instance, DeepWalk [83] leverages random walk to capture local information for learning low-dimensional node representations. They assumed that node distributions in short random walks are identical to those of words in NLP (Natural language processing). On the other hand, Node2vec [84] is proposed to improve DeepWalk by capturing higher-order graph connectivity information. A sequence-based model, it leverages random walks to learn node representations, with the goal of maximizing the probability for occurring of the subsequent nodes in the random walk. Yet, in [85], the input graphs are compressed prior to the embedding stage, and then random walks are performed to capture higher-order network connectivity information. Furthermore, [38] relies on topological similarity and random walk-based proximity scores as topological features for learning node embeddings. Other random walk-based approaches for learning graph embeddings are presented in [86–93]

However, random walks essentially capture local node connectivity information associated with a path. Moreover, there is no approach to ideal edge or node sampling.

3.3 Deep Learning.

The innovation of Graph Convolution Networks (GCN) [45] was a breakthrough for research devoted to learning embeddings for graphs and manifolds (geometric deep learning). Logically, the GCN layer computes the new node features as the average of itself and its neighbors. A single GCN captures the node's local connection information, just like a regular convolution operation deployed in computer vision. Often, multilayer GCNs are leveraged to try to explore the high-order network connectivity information. However, adding more layers renders the conventional GCN more susceptible to overfitting. As a result, the GCN is restricted to simply collecting local now proximity data and not global network connectivity information. In addition, the GCN was designed to deal with static graphs, but in reality, a lot of networks are constantly changing and developing new node interactions. Numerous GCN variants have recently been introduced to improve GCNs efficiency and adapt

it to various forms of network data and challenges. For instance, EvolveGCN [94], a dynamic variation of GCN that evolves the GCN model along a temporal dimension. It leverages RNN to update the GCN parameters, thus capturing the underlying temporal information in the network sequences. However, EvolveGCN, like GCN, assumes that every node neighbor is equally significant. On the other hand, GAT [95] proposed as a variant GCN that implicitly assigns different weights for the local neighbors of the node according to their significance to the reference node. Other variants of GCN, including FastGCN [96] and GraphSAGE [97], have been presented. They both use sampling techniques to lessen the time and memory demands of GCN. Other variants of GCN are presented in [47, 49, 50, 52, 98–100]

However, all of these GCN variations primarily focus on local node proximity information at the expense of the network topological, structural centrality global connectivity information. This dissertation bridges these gaps by proposing geometric deep-learning-based frameworks that learn network embeddings while paying particular attention to maintaining the topological node similarities, node structural centrality, and the network hierarchical information so as improve link prediction accuracy. These frameworks rely on GCN (Graph convolution network), Fully connected VAE (Variation Autoencoder), and graph translation network (GAT).

3.3.1 Graph convolution network (GCN)

The GCN layer learns how to represent the features of each node by averaging the features of all neighbors of the reference node, including itself. The feature vectors that are produced are then sent into a neural network for training. For a graph $G = (V, E)$, a GCN accepts as input a feature matrix $X \in \mathbb{R}^{N \times C}$, where N is the total number of nodes, C is the total number of input features for each graph node, and a $A \in \mathbb{R}^{N \times N}$ an adjacency matrix which denotes the structure of the graph G. A topical GCN layer is formulated as:

$$H = \sigma(\widetilde{\mathbf{D}}^{-1/2}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-1/2}\mathbf{X}\mathbf{W}). \tag{3.1}$$

where $\widetilde{\mathbf{A}} = \mathbf{A} + I$, A stands for an adjacency matrix containing a self-loop. I denotes an identity matrix to ensure that the reference node's features are incorporated into the embedding learning framework. $\widetilde{\mathbf{D}}_{ij} = \sum_j \widetilde{\mathbf{A}}_{ij}$, is degree matrix D having a self-loop, *sigma* represents a non-linear activation function. The output embeddings are denoted

by **H**. The depth of the network information captured in the GCN-based framework mainly depends on the number of GCN layers deployed. A single-layer GCN can only extract information about the node's immediate neighbors.

3.3.2 Variational autoencoder.

Motivated by the LSTM's success in discovering long-term relationships in sequential data and the effectiveness of the variational autoencoder in realizing variations in the input data [101] developed an end-to-end encoder-LSTM-decoder framework for link prediction in dynamic graphs. The model was effective in predicting both the new links that will be created and the network links that might disappear. Different from the vanilla autoencoder, which is characterized by a discontinuous latent space, the latent space of the Variational Autoencoder (VAE) is continuous, facilitating easy random sampling. It achieves this by outputting two vectors of size n: a vector of means μ, and another vector of standard deviations σ rather than outputting an encoding vector of size n as the case by the vanilla autoencoder. The i^{th} element of μ and σ is the mean and standard deviation of the i^{th} random variable X_i, where we sample to get the encoding, which is then sent to the decoder. The VAE's continuous latent space enables it to efficiently explore variations in the data in a guided manner, thus gaining the upper hand over other generative models. Moreover, VAEs exhibit good performance on diverse types of data, including sequential, non-sequential, continuous, discrete, labeled, and unlabeled, and can easily be trained with LSTM encoder-decoder pairs. Below is a summary of the basic operations of the variational autoencoder. Assuming a distribution with parameters ϕ that generates the latent representation space z from input x: $q_\phi(z \mid x)$. A conditional distribution with parameters θ where, given the latent representation, we can sample x:$p_\theta(x \mid z)$, $p_\theta(z) \sim \mathcal{N}(\mu, \sigma^2)$.

(i) Input data x

(ii) Pass x through the encoder network, which outputs a distribution of the latent states z (the mean and covariance of $q_\phi\ (z \mid x)$).

(iii) Sample latent states from the obtained distribution $q_\phi(z \mid x)$ given the input data x.

(iv) Pass samples of z through the decoder, which will output the distribution of the input data x (the mean and covariance of $p^*_\theta(x \mid z)$)

(v) Finally, sample data $\widehat{x}$ from the obtained distribution $p_\theta(x \mid z)$ given the latent variable z. $\widehat{x}$ should be similar to x.

3.3.3 Long short-term memory

Typical RNNs are characterized by exceptionally large updates to model weights as the information repeatedly goes in the loops, resulting in problems including gradient explosion and vanishing gradient. Gradient explosion is caused by repetitive multiplication of the model gradients through the network layers with values larger than 1; on the hand, the vanishing gradient problem occurs if the values are less than 1. The Long Short-Term Memory (LSTM) model [102] serves to solve these problems by learning the information that should be retained in the long-term memory and the kind of information to get rid of. It achieves this by using cell states and gates. The long-term formation for an extended number of timesteps is stored in a vector of *memory cells*. The gates are sigmoid functions, namely, the input, forget, and output gates. Below are the equations and the descriptions for LSTM gates:

$$\begin{aligned} \mathbf{i}_t &= \sigma\left(\mathbf{W}_i \cdot \mathbf{h}_{t-1} + \mathbf{W}_i \cdot \mathbf{x}_t + \mathbf{b}_i\right), \\ \mathbf{f}_t &= \sigma\left(\mathbf{W}_f \cdot \mathbf{h}_{t-1} + \mathbf{W}_f \cdot \mathbf{x}_t + \mathbf{b}_f\right), \\ \mathbf{o}_t &= \sigma\left(\mathbf{W}_o \cdot \mathbf{h}_{t-1} + \mathbf{W}_o \cdot \mathbf{x}_t + \mathbf{b}_o\right), \end{aligned} \tag{3.2}$$

where: $\mathbf{i}_t$ denotes input gate, $\mathbf{f}_t$ represents the forget gate and $\mathbf{o}_t$ for the output gate, $\mathbf{x}_t$ is the input at any current timestamp, $\mathbf{b}_j$ denotes biases for respective gate (j), $\mathbf{W}_j$ weight for respective gate (j) neurons, σ denotes sigmoid function, $\mathbf{h}_{t-1}$ denotes output of the previous lstm block. The input gate gives the new information to be stored in the memory, while the forget gate tells the information to get rid of, and the output gate gives the activation to the LSTM layer's final output at any given timestamp t.

4. INTEGRATING CENTRALITY, TOPOLOGICAL, AND HIERARCHICAL-BASED FEATURES IN DEEP LEARNING FRAMEWORKS.

This research bridges graph theory and deep learning by proposing to integrate efficient capturing and preserving of the crucial network properties in learning graph embeddings with deep learning architectures. In this regard, it introduces intuitive methodologies for mining such network property-based features, then systematic approaches for aggregating the diverse features and constrained loss functions aimed at preserving the desired network properties in the learned embeddings. The dissertation mainly focused on node topological similarity, node structural role, and network hierarchical information. Below, we detail the methodologies and frameworks leveraged to realize the desired quality of node embeddings.

4.1 Structural and Topological guided Graph Convolutional Network (STP-GCN).

To begin with, the research proposes a structural centrality and network topological similarity-based GCN framework. The framework aims at adopting the conventional GCN to implicitly preserve the node's structural centrality roles and topological information during embedding learning.

4.1.1 Problem definition.

How can we adapt the GCN framework to preserve important network-defining properties like structural centrality and node topological similarity information in order to go beyond capturing only local proximity data? Given input graph $G = (V, E, W)$, where $V = v_1, v_2, ...v_n$ denotes of n nodes, $E = \{e_{i,j}\}_{i,j=1}^{n}$ represents edges, where $e_{i,j} \in E$ connects node $v_i \in V$ to $v_j \in V$. A is the adjacency matrix A, such that $A_{i,j} = 0$ if nodes V_i and V_j are not connected by any edge, otherwise $A_{i,j} > 0$. Given a graph G, network embedding aims at learning a mapping function f:$v_i \longrightarrow \widetilde{v}_i \in \mathbb{R}^d$, $d < \|\mathbf{v}\|$. In dynamic networks, considering $\mathcal{G} = \{G_1, \mathbf{G}_2, ...G_T\}$ as a temporal (evolution) graph of

G, where G_t, is the graph's snapshot at any given time t. We aim to learn the function f_t that effectively maps each node feature vector v in a sequence of low-dimensional vector spaces $\{\widetilde{v}_1, ...\widetilde{v}_t\}$, where $\widetilde{v}_t$ is the vector embedding that represents the node v at time t at any given time t. $\widetilde{\mathbf{v}}_t = f_t(\mathbf{v}_1, ., ., \mathbf{v}_t)$. Our goal is to efficiently learn network embeddings while capturing and preserving the node's structural centrality roles and the network topology in order to increase the accuracy of graph applications, including link prediction, which depends on the quality of the output network embeddings. The architecture of the proposed STP-GCN is illustrated in Fig. 4.1. Given the raw network interaction datasets, we first obtain adjacency matrices from them. Next, we utilize a generalized GCN formulation in Eq. 3.1, which is divided into the feature transformation and aggregation phases as correspondingly illustrated below in Eq. 4.1 and Eq. 4.2:

$$Z = f(X) \tag{4.1}$$

$$H = h(g(A), Z) \tag{4.2}$$

Unique to our strategy, our feature transformation function f in Eq. 4.1 is the node strength centrality Eq. 2.2, a more potent variant of node degree which takes edge connection weights into account. It not only captures the node's structural connectivity information in the entire network but also reveals the node's centrality role. In addition, A topological similarity-based function (g) serves as our feature propagation rule; a feature aggregation function h, and an output matrix H that stores the node embeddings. Z', a consequent intermediate feature propagation matrix with high-level information from both the node features and the graph structure is computed as below;

$$Z' = \sigma(g(A), Z) \tag{4.3}$$

We leveraged an LSTM layer in order to capture the underlying temporal information in the changing graphs at the various time stamps. The graph developing module is defined as below;

$$H^t = LSTM(\mathbf{Z}', H^{t-1}), \tag{4.4}$$

where H^t denotes the graph embedding matrix graph snapshot at time t. H^0 is matrix of zeros. Below we define the model's total loss function.

$$L = \sum_{t-1}^{T} ||(Z^{(t)} - H^{(t-1)})||^2 \tag{4.5}$$

where $Z^{(t)}$ represents the final graph embedding matrix. We test a variety of topological similarity metrics, such as the Adamic Adder, Jaccard Index, Sorensen Index, Common Neighbor, Resource Allocation Index and Salton Index, whose corresponding convolution matrix formulations are detailed in eqns. 2.3- 2.11.

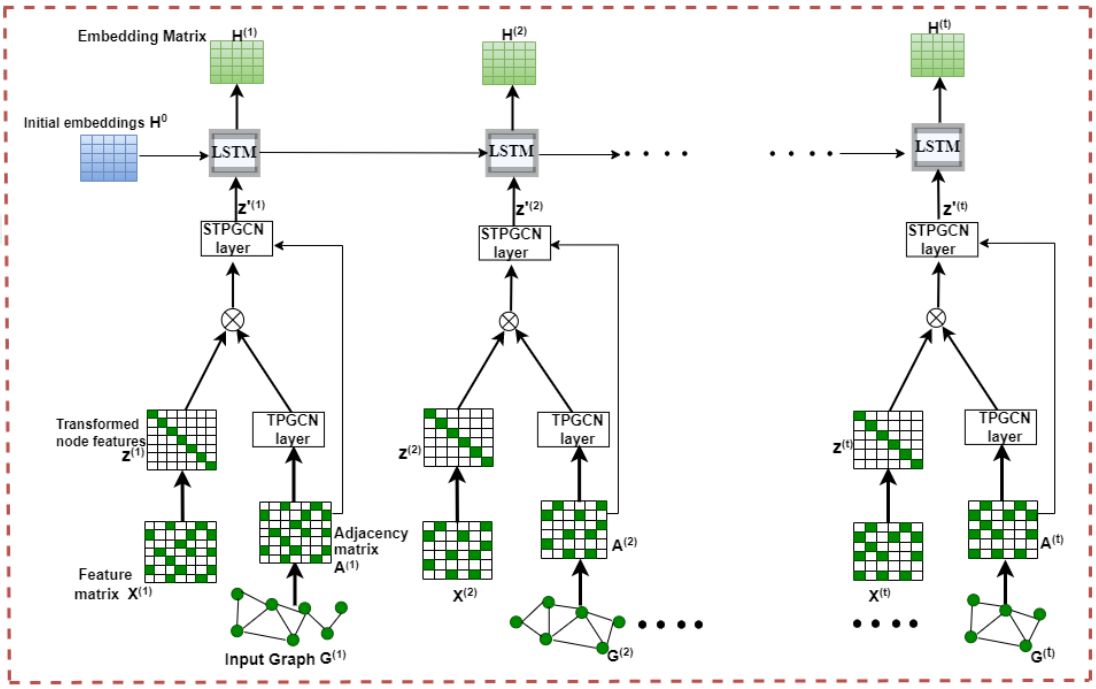

Figure 4.1 : Framework of the STP-GCN model; An adjacency matrix A is taken from the input graph G at each time step and supplied to the TPGCN layer. Following Eqns. 2.3-2.11, we compute the topological similarity convolution matrix inside the TPGCN layer. The TPGCN layer's output is then aggregated with the node centrality feature matrix obtained from the feature matrix. The resultant high-level features and the raw adjacency matrix act as the inputs for the STPGCN layer. Within the STP-GCN layer, Z' is computed following eq. 4.3. The STPGCN layer's output (Z') is fed to the LSTM layer 4.4 before computing the final graph embedding matrix H

. [103].

4.1.2 Experiments

We tested the proposed STP-GCN on 5 (five) real-world social network datasets for the link prediction task.

4.1.2.1 Datasets

The five (5) real-world datasets that were used in this study include: the Internet's network of routers structured into subgraphs called Autonomous Systems (AS), where each AS exchanges traffic flows with a few peers (neighbors)[1]. Math Overflow (Maths), a network for users to ask and answer questions on mathematical issues from the stack exchange[2]. Facebook, a social network user interaction data from Facebook [3], a community message network made up of the University of California, Irvine (UCI) students, where the nodes represent the users of the social network and the edges represent the messages exchanged by the users[4]. Lastly, Enron, a network of email datasets for emails sent and received by Enron Corporation employees[5]. Descriptions of the datasets are provided in Table Table 4.1.

Table 4.1 : Description of the datasets

Dataset	# Nodes	# Edges	# Snapshots
Math	24740	323357	77
AS	6828	1947704	100
Facebook	60730	607487	27
UCI	1899	59835	7
Enron	87036	530284	38

4.1.2.2 Setting of the model parameters

The learning models are optimized using the ADAM optimizer [104], with the number of epochs heuristically fixed to 60. For the embedding size, the range {64,128,256} is searched, and 128 is selected as it produces the best results. Searching from $\{0.001, 0.002, 0.003, 0.004, 0.005\}$, a learning rate of 0.002 was chosen. The model takes a full batch in every training epoch. We set the number of layers to 2, the dropout

[1] http://snap.stanford.edu/data/as-733.html
[2] http://snap.stanford.edu/data/sx-mathoverflow.html
[3] http://networkrepository.com/fb-wosn-friends.php
[4] http://konect.cc/networks/opsahl-ucsocial/
[5] http://networkrepository.com/ia-enron-email-dynamic.php

to 0.5, and the number of head attentions in GAT to 8 in all experiments. A GPU with Ubuntu 20.04.3 LTS, Nvidia GeForce GTX 1080Ti, 32 cores, 64GB of RAM, and clock 33MHz was used to run the experiments.

4.1.2.3 Evaluation metrics

In a dynamic environment, based on the embeddings learned from every prior graph up to time step t, we predict whether a connection between nodes in a network will be present or absent at time step $t+1$. Negative edge samples are obtained by randomly sampling node pairs that are not connected by any edges, and positive edge samples are produced by randomly sampling true edges at time step t. The performance of the model was evaluated by the AUC (Area Under the Receiver Operating Characteristic (ROC) Curve) score. Plotting the true positive rate (TPR) against the false positive rate (FPR) yields the ROC curve. TPR is the ratio of positive edges that were correctly predicted out of all positive edges. It is calculated as in 4.6 below:

$$TPR = \frac{TP}{(TP+FN)} \tag{4.6}$$

Where TP (True Positive) denotes the correctly predicted positive edges and FN (False Negative) represents positive edges that were incorrectly predicted as negatives. On the other hand, FPR is the ratio of edges predicted as positive out of all negative edges. Below is its computation:

$$FPR = \frac{FP}{(TN+FP)} \tag{4.7}$$

Where FP (False Positive) represents the negative edges wrongly predicted as positive, and TN (True Negative) denotes the correctly predicted negative edges. The AUC is the likelihood that a randomly chosen positive edge would appear above a randomly chosen negative edge. It ranges from 0 to 1, with 1 being the ideal. The test accuracy is stated as the mean AUC score.

4.1.2.4 Baselines

- LIST [78], a method for predicting dynamic network links that preserve both spatial and temporal consistency and leverage a dynamic matrix factorization approach for representing the network structure as a function of time.

- GCN [45]: Applies a unique layer-wise propagation mechanism to traditional Convolutional Neural Networks (CNN) to generalize them to graphs and manifolds. It learns the node's new features by averaging the features of all the neighbors of the node, including the node itself.
- GIN [105]: It is a GCN variation that uses MLP in place of linear layers to more clearly differentiate between various network structures.
- Evolve GCN [94]: A GCN's dynamic variant that updates the GCN parameters by capturing the graph-evolving information with an RNN.
- GAT [95]: a GCN variant that uses masked self-attentional layers to learn different weights for various nodes in a node's local vicinity.

4.1.2.5 Experimental results

The link prediction AUC scores for the proposed model and other popular baseline methods are shown in Table 4.2. All methods tested on the five social network datasets that are described in section 4.1. Each method's average training time per epoch is displayed in the table's last column. The best score in each column is displayed in boldface font. The Wilcoxon Signed Rank Test [106] was used to determine the significance level of the results, considering 0.05 as the significance level. On almost all data sets, apart from Facebook, our suggested technique STP-GCN is seen to perform significantly better than the baseline models. This is because, during network embedding learning, it is able to preserve the network topology and the node centrality information in addition to capturing the evolving network information in the temporal networks. The dynamic versions of the GCN, including EvolveGCN, are seen to outperform the typical static GCN; this emphasizes the importance of efficiently utilizing the temporal information in the dynamic networks when learning embeddings.

4.1.3 Results analysis.

From Table 4.2, the proposed STP-GCN is observed to significantly outperform the baseline models in four (4) of the five (5) real-world social networks that were used in this study. This demonstrates the supremacy of the topological and node centrality-based features in upholding the general topology and structure of the

Table 4.2 : Average link prediction AUC scores and average model training time for each epoch (last column)

Method	UCI	AS	Maths	Facebook	Enron	t(s/epoc)
GIN	0.8416	0.8667	0.8849	**0.7298**	0.8552	2.48
GCN	0.7468	0.7891	0.8153	0.5894	0.8426	**1.45**
GAT	0.814	0.6974	0.7353	0.5531	0.8566	15.09
LIST	0.8614	0.8911	0.8695	0.6309	0.8593	23.38
Evolve GCN	0.8864	0.8997	0.8901	0.6429	0.8595	1.96
STP-GCN	**0.8943**	**0.9192**	**0.9168**	0.7266	**0.8846**	2.16

network. However, for the Facebook dataset, it is noteworthy that GIN outperforms STP-GCN along with other baseline GCN-based variations slightly better due to its capacity to figure out sophisticated network patterns that GCN-based models might miss. Dynamic methods, including the proposed STP-GCN, LIST, and Evolve GCN, are consistently observed to perform better than static ones, such as GCN and GAT. This is due to their ability to uncover and utilize the underlying historical information in dynamic networks. The smallest training time length is observed for GCN-based approaches, including the proposed STP-GCN, with the dynamic STP-GCN and Evolve GCN having a little slower training time than the static GCN due to the LSTM layers involved in their models. Noteworthy, GAT is associated with a lengthy training time due to its attention mechanism, and LIST is characterized by the longest time due to the time-consuming matrix factorization procedure.

4.1.3.1 Result analysis due to the different topological similarity-based metrics deployed.

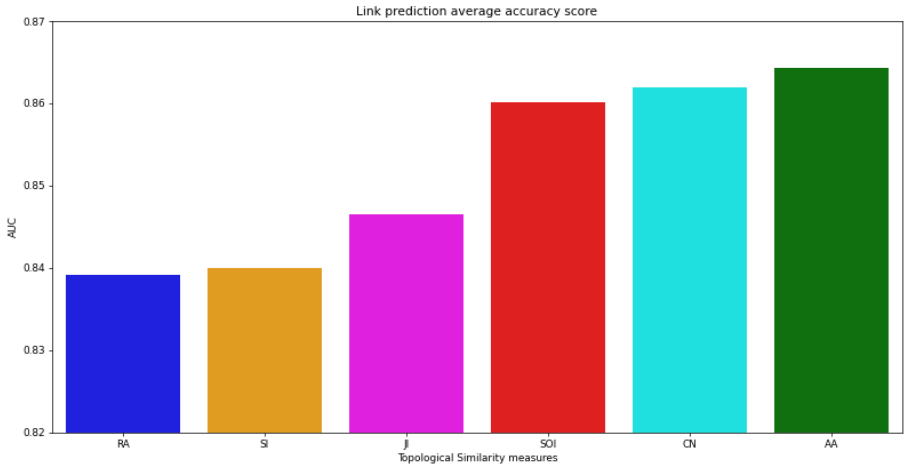

Figure 4.2 : The average link prediction AUC scores across all the five datasets based on the six different topological similarity metric features analyzed.

The average link prediction AUC scores resulting from the 6 (six) distinct topological similarity-based metrics on the 5 (five) datasets deployed are shown in Fig. 4.2. From Fig. 4.2, On average, features based on Adamic Adder, Common Neighbor, and Sørensen Index outperformed other topological similarity metric-based features. Furthermore, we observed the relevance of the topological similarity-based convolution metric features to be dataset-dependent. For instance, on some datasets, but not all, features based on Adamic Adder similarity metrics could perform better than those based on Common Neighbor.

4.1.3.2 Assessing the efficiency of the methods in preserving the structural centrality role of graph nodes in their predicted graphs.

Node centrality was used to deduce the nodes' structural roles played in the network. We assessed the Closeness, Betweenness, and Degree centralities of graphs predicted from embeddings learned by our proposed STP-GCN as well as the baseline approaches. The degree centrality reveals the total number of one-hop links each node has to other nodes in the network, and the node with the large node centrality tends to have high centrality scores for other centrality measures. Yet, the closeness

centrality determines how fast information would be transferred from one node to all other nodes in a network. (i.e., it quantifies the length of the shortest pathways between the reference node and all other nodes in the network). On the other hand, betweenness centrality quantifies the extent to which a reference node is located on the shortest pathways connecting other nodes on the network (i.e., it aids in identifying nodes that take on "bridge-spanning" positions across networks (sub-networks)). The experiment was conducted on the Maths and Enron datasets. Figs. 4.3 and 4.4 illustrate the average between MSE for predicting centralities of the predicted graphs by the proposed STP-GCN and the baseline methods. We compare the centralities of the predicted graphs to those of the grand truth test graphs, and we report the average MSE.

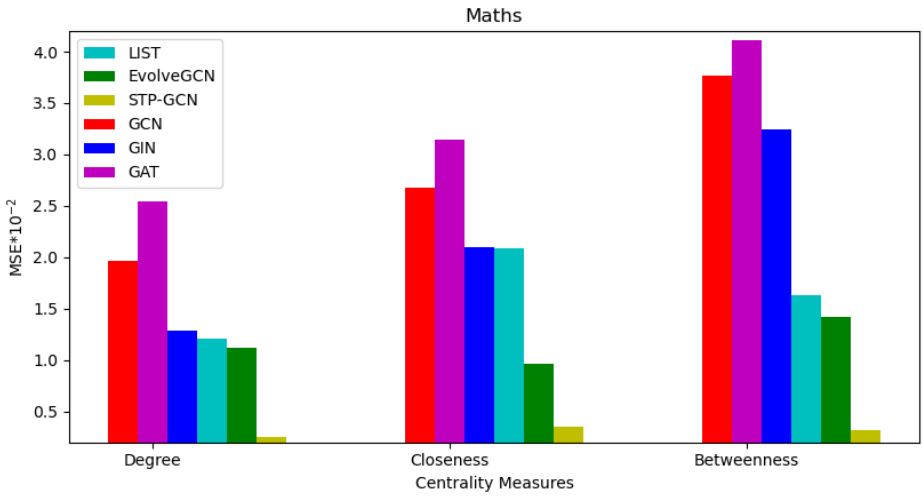

Figure 4.3 : Centrality prediction MSE for the Math dataset

According to Figs. 4.3 and 4.4, Comparing the predicted graphs from the proposed STP-GCN embeddings to the graphs predicted from the embeddings of baseline approaches, STP-GCN-based graphs exhibit consistently lower centrality MSE values. Many thanks to the node centrality and topological-similarity-based features that made it possible for the proposed STP-GCN to effectively capture and preserve the structural role of nodes and the network topology.

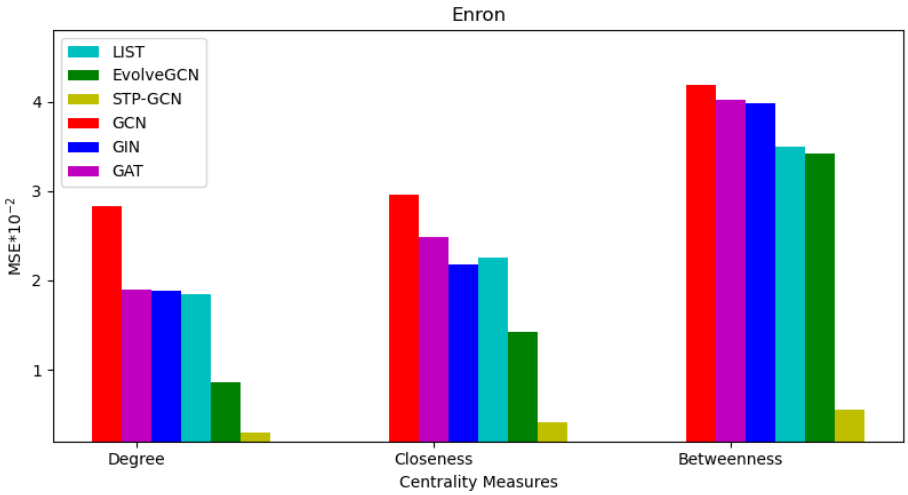

Figure 4.4 : Centrality prediction MSE for the Enron dataset

4.1.4 Summary.

In this study, an end-to-end link prediction framework is proposed that not only captures the underlying temporal information in dynamic networks but also preserves the node structural centrality roles and the general network topological information. The proposed STP-GCN model's AUC score for the link prediction task is significantly greater than the best baseline, while the model is time-efficient and exhibits a considerable decrease in MSE on structural graph centrality prediction. We have shown how the essential topology and centrality-based features may be added flexibly into the GCN framework, and we have examined their impact on the generated embeddings. Following our methodology, future work can flexibly incorporate preserving additional vital network property-based attributes, such as node participation coefficient, eccentricity, and modularity, into the GCN-based embedding learning architecture to enhance the learned embedding quality. It is also worthwhile to investigate features from the global-based topological similarity metrics, such as the Random Walk with Restart, Average Commute Time as the Katz Index's global variant, and Matrix Forest Index, along with the global-based centrality metrics, such as Closeness and Betweenness Centralities, given their efficacy in characterization networks.

4.2 Topological Similarity and Node Centrality-guided Hybrid Deep Learning Model for Link Prediction in Dynamic Networks (TSC-TLP)

The TSCL-TLP model is motivated by; First, the overwhelming success of fully connected graph variational autoencoders in learning variations in the input data. Second, the efficiency of LSTM in exploring historical information sequential data. Lastly, the fact that the quality of learned embeddings is improved by imposing graph structure and property-preserving constraints on low-dimensional learning techniques. The model builds on fully connected encoders, decoders, LSTM layers, and network property-based learning constraints to learn quality embeddings for link prediction. The flowchart in Fig. 4.5 shows the major steps followed in the proposed TSC-TLP. Below we describe the steps shown in the flow chart Fig.4.5 that summarizes the proposed TSC-TLP framework. The model takes as input the aggregation of the topological similarity and node centrality matrices of historical graphs, and its output is predicted the graph at any next time step. The model captures the relationship patterns that exist between network nodes at every time and over the course of the time steps and learns the node embeddings while preserving the node structural roles and the general network topology. To begin with, the node strength matrices and topological similarity-based matrices are computed from the raw interaction social network datasets as guided by Eq. 2.2 and Eqs. 2.3- 2.11. These matrices are then aggregated by a summation operation and sent to a fully connected auto-encoder, which produces a low-dimensionality representation of a graph node u, as seen in the eq. 4.8 below:

$$y_{u_t}^{h} = f_a(W^h y_{u_t}^{h-1} + b^h). \tag{4.8}$$

where the fully connected auto-encoder's output layer is denoted by h. The activation function f_a, a learnable weight matrix W, and a bias b. The learned low-dimensionality graph representation is then fed into the LSTM network, as shown in the equation Eq. 4.10 below:

$$y_{u_t}^{h+1} = O_{u_t}^{h+1} * tanh(C_{u_t}^{h+1}) \tag{4.9}$$

$$O_{u_t}^{h+1} = \sigma_{u_t}^{h+1}(W^{j+1}[y_{u_t-}^{h+1}, y_{u_t}^{h}] + b_o^{h+1}). \tag{4.10}$$

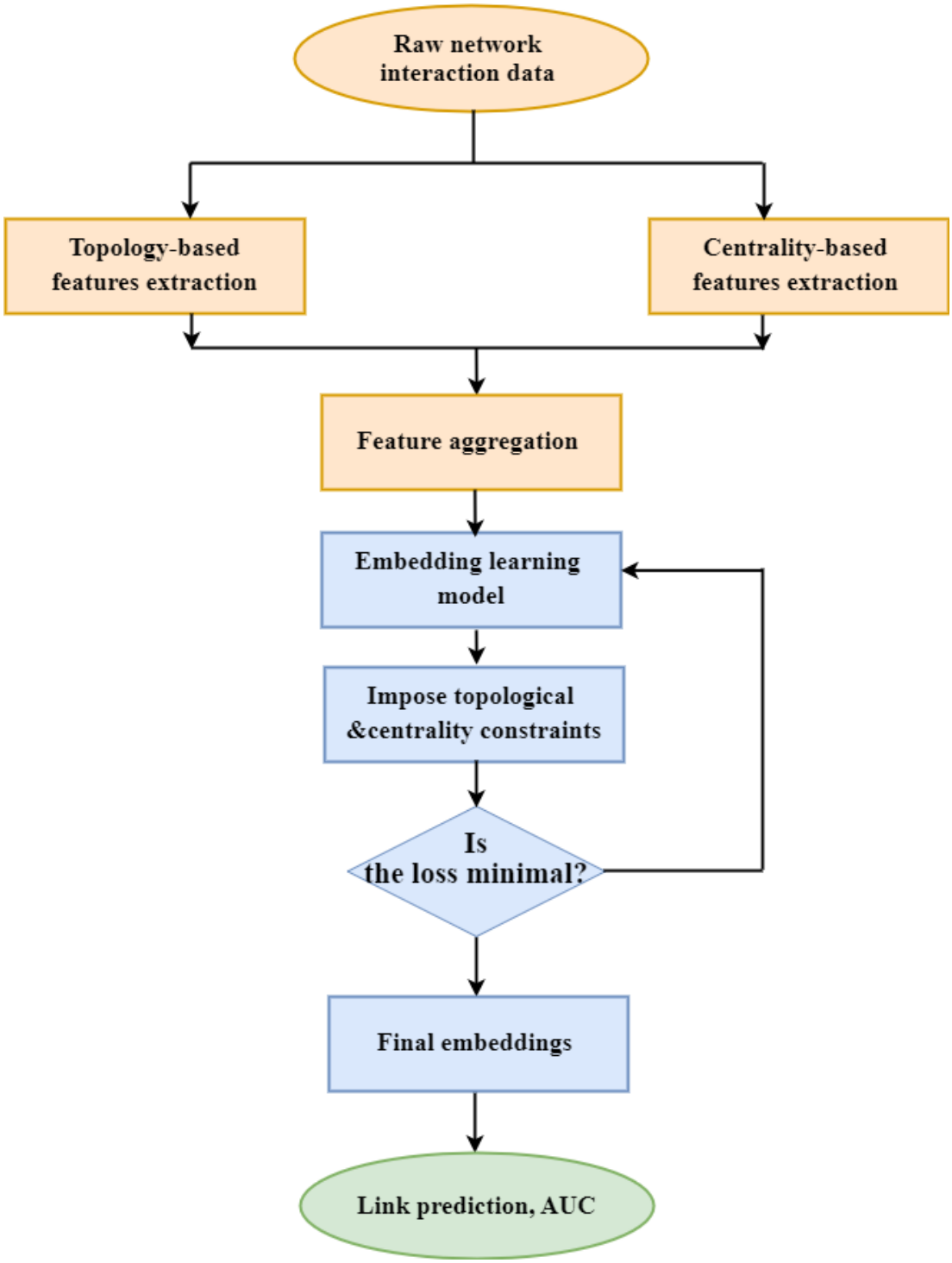

Figure 4.5 : The proposed TSC-TLP's flowchart. Node centrality and topological-similarity features are derived from the social network raw datasets as per Eqn. 2.2 and Eqns. 2.3-2.11. Then the features are systematically aggregated. The subsequent features are supplied to a deep learning model to model to learn the node's low representations. Then, to further enforce the preservation of the node structural centrality roles and the general network topology in the learned embeddings, the centrality and topological-similarity constraints are imposed on the learning model's loss function as per Eqn. 4.12. The learning model is retrained up until the training loss falls to a certain minimum. The network at time $t+1$ is predicted using the final embeddings learned for time t. The link prediction accuracy is calculated using the AUC score as per Eqs. 4.6-4.7

The LSTM layer's output is then passed to the decoder (fully connected neural network). For the model's loss function, we adhere to the strategy proposed in [107], where the model learns the graph's low representations by optimizing loss function presented in eq. 4.11 below:

$$\begin{aligned} \mathscr{L}_{t+n} &= ||(\widehat{A}_{t+n+1} - A_{t+n+1}) \odot \mathscr{B}||^2 \\ \text{where, } \widehat{A}_{t+n+1} &= f(\mathbf{A}_t, ..., A_{t+1}). \end{aligned} \tag{4.11}$$

Where the weighting matrix $\mathscr{B}$ prioritizes observed edges over unobserved ones during edge reconstruction as it is done in [108]. $\odot$ represents the element-wise product. Owing to the recommendation in [109] to incorporate the graph structure preserving constraints in embedding learning models and also the findings in [17, 110] associated topological constraints improved the performance of the learning models. We impose topological similarity and node strength constraints to model learning to ensure that the input graphs' topological and centrality structures are preserved in the learned graph embeddings. Consequently, the model's overal loss function is shown below in eq. 4.12.

$$\begin{aligned} \mathscr{L}_{t+n} =& ||(\widehat{\mathbf{A}}_{t+n+1} - A_{t+n+1}) \odot \mathscr{B}||^2 \\ &+ \gamma ||(tf(\widehat{A}_{t+n+1}) - sf(A_{t+n+1}))||^2 \\ &+ \lambda ||(sf(\widehat{A}_{t+n+1}) - sf(A_{t+n+1}))||^2 \end{aligned} \tag{4.12}$$

Where sf and tf denote the node strength centrality and topological similarity formulating functions. γ and λ, respectively, control the weight allocated to the topological similarity and the node strength centrality constraints. n is a window size parameter that controls the length of the learned temporal patterns. The proposed TSC-TLP flow is summarized in Pseudocode 2. We examined features for four widely used topological similarity metrics, including Jaccard Index [55], Adamic-Adar [54], common neighbor [53], and Salton Index [56]. The proposed TSC-TLP model's framework is shown in Fig. 4.6.

Algorithm 2: The proposed TSC-TLP model's pseudocode

Input: $\mathbf{G} = (V, E)$

1 From the input Adjacency matrix, compute the node strength as guided by Eq. 2.2;
2 Extract topological similarity feature matrices from the adjacency matrix following Eqs. 2.3-2.11 ;
3 Aggregate resulting feature matrices in steps 1 and 2;
4 Supply the aggregated features to the embedding learning model, then train it using Eqs.4.8-4.12 ;
5 Calculate the probability of a link presence or absence between node pairs at a given time $t+1$;

Output: AUC score, following Eqs.4.6-4.7.

4.2.1 Experiments

We performed experiments on 3 (three) publicly accessible real-world social network datasets to assess the quality of embeddings generated by the TSC-TLP model. The datasets include UCI, AS, and Maths as described in Table 4.1.

Baseline methods

- SAGE [97]: An inductive framework for node embedding learning, where a node's low dimensional feature representations are learned by sampling and aggregating features from its immediate neighborhood.
- DynGEM [111]: It makes use of deep autoencoders to gradually learn low-dimensionality graph representations for temporal graphs.
- VAE-LSTM: It is a proposed TSC-TLP variant that leverages LSTM and variational autoencoder layers to learn from previous input graphs and predict the graph at the next time step.
- TSC-TLP-wc: This is a TSC-TLP variant in which the model learning is not subject to any topological similarity or node centrality conserving constraints.

4.2.1.1 Experimental results

Table 4.3 displays the link prediction AUC scores for the proposed TSC-TLP model and the baseline graph embedding studies on the three real-world networks. The bold typeface is used to indicate the best results in each column. In order to access

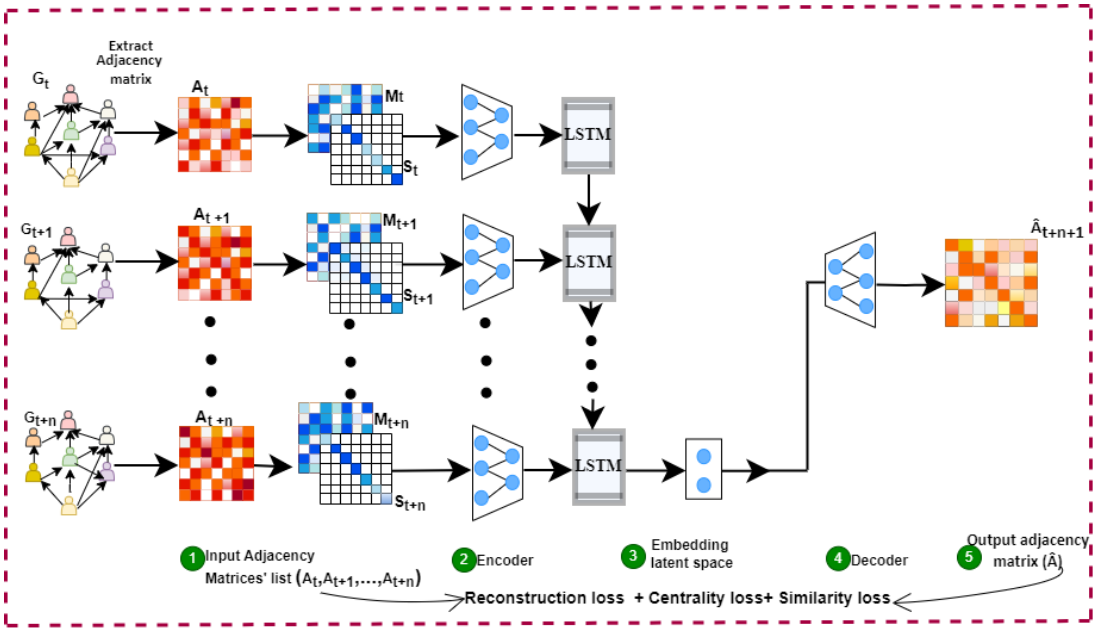

Figure 4.6 : The proposed model's framework. An adjacency matrix is obtained from the network interaction data at each time step, and it is utilized to compute the node centrality and topological-based feature matrices as guided by the formulations in Eq. 2.2 and Eqs.2.3-2.11 respectively. The generated features are aggregated and supplied to the fully connected autoencoder Eq. 4.8, which then learns the node embeddings. The resulting low dimensional node representations are then fed to the LSTM network to capture the prevailing temporal information in the graph sequences directed by Eq.4.10. The output of the LSTM layer is then passed to the decoder network. An adjacency matrix is then sampled from the output of the decoder. The model loss function is made up of the reconstruction loss, topological similarity loss, and node centrality loss, which are all optimized concurrently [1].

the impact of the centrality strength and topological similarity-based features on the proposed TSC-TLP model, we create its variant VAE-LSTM that takes as input the raw adjacency matrices. In addition, we assess the impact of the node strength centrality and topological similarity constraints on the TSC-TLP model by creating a variant TSC-TPL-wc that has no such property-based constraints improved on its loss function. Furthermore, by comparing prominent embedding learning dynamic methods, such as Evolve GCN and its static version GCN, we verified the importance of the temporal modules like the LSTM network in the dynamic-based methods. As observed in Table 4.3, for all datasets, our proposed TSCL-TLP is seen to perform significantly better than the baseline models and its variant. This is a result of its ability

to preserve the node centrality roles, network topology and the underlying temporal information in evolving graphs.

Table 4.3 : Comparison between the baseline approaches and the suggested TSC-TLP model based on the average AUC scores for link prediction task over all datasets and timestamps.

Method	UCI	AS	Maths
GAT	0.8174	0.6974	0.7353
GCN	0.7449	0.7882	0.8027
GIN	0.8416	0.8667	0.8849
Evolve GCN	0.8872	0.8997	0.8901
DynGEM	0.9101	0.9327	0.9248
TSC-TLP	**0.9363**	**0.9471**	**0.9308**
LSTM-VAE	0.9028	0.9246	0.9004

The dynamic EvolveCGN is observed to outperform the typical static GCN, highlighting the significance of capturing of temporal information during embedding learning in dynamic graphs.

4.2.1.2 TSC-TLP variants

We present various variants of the proposed TSC-TLP model in order to analyze the contributions of the various modules it holds.

- C-TLP: Ablation of the TSC-TLP relying solely on centrality-based feature matrices as inputs.
- TS-TLP: A TSC-TLP ablation that only utilizes topological-similarity-based features as inputs.
- TSC-TLP-wc: An ablation with similar inputs as TSC-TLP but without topological similarity and node strength constraints imposed on model learning.

The link prediction AUC scores of these variants are shown in Table 5.1.

Table 4.4 : Comparing the average link prediction AUC scores of the proposed TSC-TLP and its ablated version across all datasets and timestamps.

Method	UCI	AS	Maths
TS-TLP	0.9253	0.9361	0.9282
C-TLP	0.7251	0.7880	0.7104
TSC-TLP	**0.9363**	**0.9471**	0.9308
TSC-TLP-wc	0.9225	0.9299	**0.9316**

4.2.2 Results analysis

The main goal of this study is to develop an end-to-end integrated model that learns low-dimensional graph representations while maintaining the network topology, the node centrality structural roles, and the underlying temporal graph information in graph sequences so as to predict network node connections with high accuracy. In line with the goal, on top of mining and utilizing centrality and topological features from the input data, we further leverage topological similarity and node strength centrality constraints on the model loss function to ensure that the original topological structure and structural role of nodes are preserved in the learned embeddings.

4.2.2.1 Examining how window size affects link prediction accuracy

In this experiment, we investigate the impact of changing n (number of temporal graphs learned in the LSTM layer) on the link prediction AUC. The AS dataset was selected for this study. While recording changes in the AUC, the number of temporal graphs used at a given time is changed as n *in* [5,10,20,25,30]. Fig. 4.7 displays the test results for models on the AS dataset for the various number temporal graphs. For dynamic approaches, such as TSC-TLP, EvolveGCN and DynGEM, the AUC is seen to increase steadily with the growing number of temporal graphs because these models make use of the memory units to keep and learn from the historical temporal patterns. Beyond n=25, however, no significant rise in AUC is noticed. On the other hand, for static methods such as GCN, the AUC initially grows with the window size, but after a certain point (e.g., for window size beyond n=15 for GCN), the AUC is seen to fall

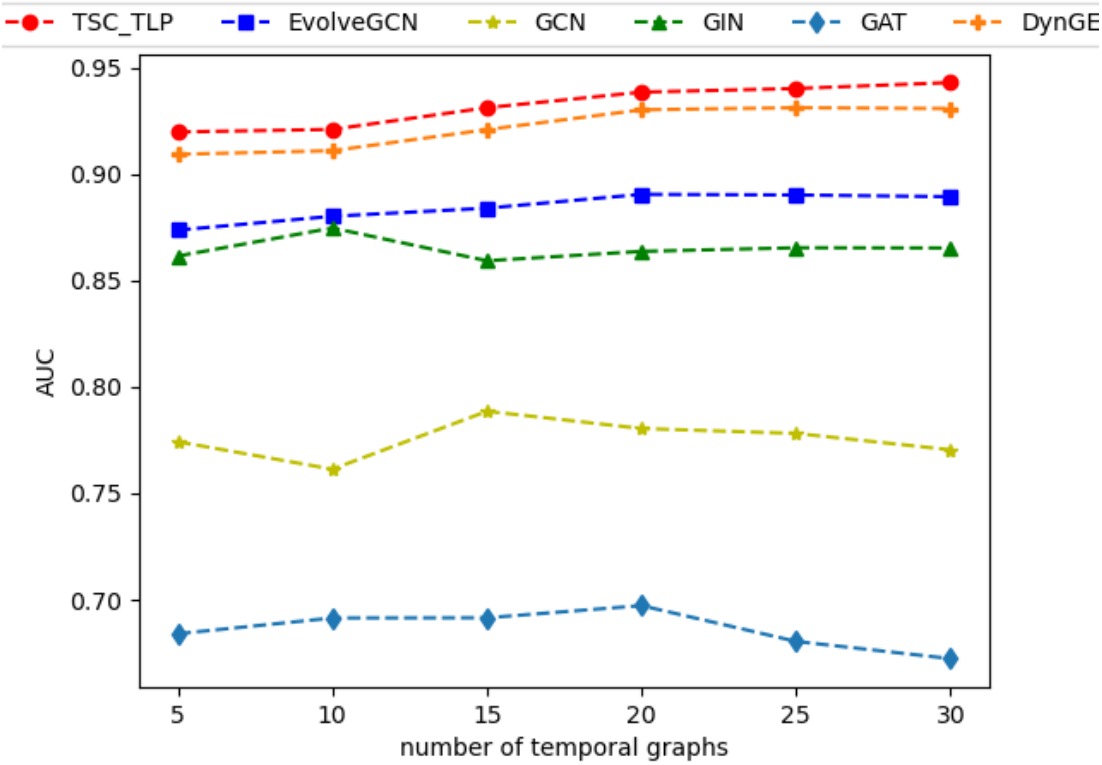

Figure 4.7 : AUC vs. Increasing number of temporal graphs on AS dataset

since such static methods lack memory cells for storing and learning from historical temporal patterns.

4.2.2.2 Examining how the embedding learning model is impacted by the network topological and node centrality preserving constraints.

From Table 5.1, the proposed TSC-TLP is seen to perform better on average than its variant TSC-TLP-wc, which eliminates the topological and centrality-preserving constraints on the loss function. This emphasizes the importance of imposing constraints that preserve network properties when learning network embeddings, as explained in earlier studies [17,109,110]. Thanks to these network property-preserving constraints. The network topological information and centrality roles of graph nodes were efficiently preserved in the learned embeddings thanks to these network property-preserving constraints imposed on the model learning process. As a result, the quality of learned embeddings was enhanced, which in turn, increased the accuracy of link prediction. The TSC-TLP-wc variant, however, appears to produce better results for the Maths dataset than the proposed TSC-TLP; this is probably because the

γ and λ hyper-parameters, which control the weights allocated to the node topological similarity and strength centrality constraints respectively, were not optimized. So, we did not get the best results possible.

4.2.3 Limitations, opportunities and future work

The following are some of our research's major limitations: A limited number of local topological similarity metrics were first utilized. Future research will include a thorough analysis incorporating additional topological similarity metrics, including Resource Allocation Index [57], Hub Depressed Index [60], and Preferential Attachment [112]. Second, crucial hyper-parameters like γ and λ, which control the weights allocated to centrality and topological constraints, were not optimized, thus affecting our results. Better outcomes can be obtained by automating learning and fine-tuning these hyperparameters. Lastly, we focused only on capturing and preserving the node centrality and topological-similarity information. To further enhance the embedding quality, it is worth looking into additional network properties, including community and hierarchical structures.

4.2.4 Summary

This study introduces an end-to-end network topology and node centrality-driven link prediction model for dynamic networks (TSC-TLP). The model captures the underlying temporal information in dynamic networks in addition to maintaining the network topological information and node centrality roles in the learned embeddings. The model is enhanced with special topological similarity and centrality constraints on its loss function to further enforce the preservation of these network properties in learned embeddings. As a result, the model produces quality embeddings that increase link prediction accuracy. It registers an average of 4% in link prediction AUC and a 37% reduction in MSE on predicting node centralities for predicted graphs by its learned embeddings. We have examined the roles of each module of the proposed TSC-TLP, including the property constraints, centrality-based features, topological-based features, and LSTM modules, by assessing various model variants to understand how each module impacts the model's final output.

5. CENTRALITY AND HIERARCHICAL-AWARE EMBEDDING LEARNING IN HETEROGENEOUS NETWORKS

5.1 A Hierarchical and Centrality-aware Model for Polypharmacy Side Effect Prediction (HC-POSE)

In this section we present the HC-POSE model that deploys k-core decomposition to reveal the underlying hierarchical information in the complex biological networks. Besides the model laso utilizes node centrality based feature matrix to better the quality of embedding learning.

5.1.1 Introduction to polypharmacy side effect prediction (POSE)

Polypharmacy is the practice of simultaneously prescribing a number of medications (five or more) to a patient. Drug interactions are one of the main problems associated with polypharmacy. As a number of different medicines are taken together simultaneously, they can interact in a variety of ways, causing unexpected adverse effects or decreasing the effectiveness of the drugs. Drug interactions can occur via various mechanisms, including pharmacokinetic interactions (having an effect the way the drug is absorbed, distribution, metabolism, or discharge of drugs) and pharmacodynamic interactions (altering how drugs function in the body). Common polypharmacy side effects include risk of bleeding, sedation, drowsiness, sexual dysfunction, low blood pressure, muscle pain, dizziness, low blood pressure,weakness and many others. Individual human traits, medical histories and reactions to medications could significantly affect the overall threat of the polypharmacy side effects. Particularly individual factors like one's overall health status, feeding, age, genetic variations, organ function and other medical conditions can significantly affect patient's response to simultaneous multiple medications.

Predicting the adverse effects of polypharmacy can be a difficult task that frequently requires the assistance of medical professionals, such as medical professionals, pharmacists, or researchers. The approaches often deployed to detect possible

polypharmacy adverse effects include performing a thorough evaluation of a patient's therapy, Collecting a patient's comprehensive medical history and analysing drug-drug interactions with special softwares learning models that benefit from the Drug Interaction Databases. However, Each patient's reaction to a therapy is often unique. In addition, personal traits can substantially affect the likelihood and magnitude of adverse effects. Thus, rendering polypharmacy side effect such a challenging task. To enhance effectiveness of research devoted to this task, we aim to learn quality POSE dataset network-property based-embeddings that could consequently improve the POSE prediction accuracy. We modeled the polypharmacy side effect prediction as a link prediction problem. This eased the application of the conventional graph theory and geometric deep learning approaches that are commonly used in link prediction to the POSE (polypharmacy side effect) prediction task. We propose a hierarchical and centrality-aware graph representation learning architecture for heterogeneous graphs and precisely the polypharmacy side effects dataset. These datasets are characterized by different node types, where a node could represent a drug or a protein. The interactions between nodes (edges) represent different types of side effects. We aim at predicting side effects, specifically in the drug-drug (DD) interactions, while benefiting from all the available network information, including Protein-Protein interactions, protein-drug interaction and drug-drug interaction, which diverse information serves to enhance the accuracy in the side effect prediction.

5.1.2 Geometric deep learning for POSE

Inspired by the efficiency of graph convolution (GCN) on learning embeddings for graphs and manifolds, Decagon [113] developed the first geometric deep learning model for predicting polypharmacy side effects (POSE). This GCN-based multi-relation type model predicts a link between a side effect and a co-prescribed drug pair. It learns node (protein/drug) low-dimensional representations by weighted aggregation of the node's local neighborhood information, including the node itself. It maps drugs and proteins in the same latent space and predicts all relationships between all nodes (proteins and drugs). The method is characterized by large time and memory resources, especially for big networks with several nodes and edge types. Moreover, it registers relatively poor prediction accuracy. Many other geometric deep learning models have been recently developed to tackle the POSE problem [114–120]. In

particular, [121] improves the approach in Decagon by mapping drugs and proteins into different latent spaces. It predicts links between any pair of drugs, not all node types like Decagon. Yet, SumGNN [122] introduces a multi-typed drug-drug prediction knowledge graph that utilizes a self-attention subgraph summarization approach and a data integration module to enhance the prediction of multi-type drug-drug associations. However, the above methods accord less importance to sufficiently preserving the essential network information, including the network's hierarchical information, the role of each node in the network, and general network topology. To close this gap, we introduce a hierarchical and centrality-driven approach for POSE prediction that utilizes the k-core decomposition algorithm 1 to uncover the hierarchical-based graph embedding features and node strength centrality features to conserve the centrality roles of network nodes when learning embeddings for POSE prediction [123].

5.1.3 Contributions of the proposed HC-POSE.

Below we summarize the key contributions of the proposed HC-POSE.

- Efficiently capturing the underlying hierarchical information from the POSE datasets to improve their embedding learning quality.
- leveraging node centrality features to enhance the latent node representation quality for the POSE networks.
- Intuitively aggregating features from the various subgraphs of the complex POSE networks so as to suit the heterogeneous complex nature of the datasets during embedding learning.
- Proposing an end-to-end deep hybrid POSE prediction architecture that unites all the submodules in a unified framework to avoid error accumulation due to independent subtasks.

5.1.4 The proposed hierarchical and centrality aware polypharmacy side effect prediction model (HC-POSE).

Inspired by the efficiency of k-core decomposition in revealing hierarchical network information [67, 70], and the role of node strength features in preserving network topological information during embedding learning [17]. We propose to focus on the network hierarchical and node strength-based features primarily. Given the nature of the POSE datasets with dense subgraphs and other sparse ones. We hope to benefit from the unique information of each sub-graph (hierarchy) before spreading it to other subgraphs. In addition, owing to the fact nodes with high strength centrality tend to have high centrality scores for all other centrality metrics as well; the node strength centrality was chosen to preserve the structural roles of the nodes in the learned embeddings. The model's inputs are the Drug and the Protein adjacency matrices. Next, to uncover the underlying hierarchical information, the input graph is subjected to k-core decomposition. In order to conserve the network topological structure and the node centrality roles, node strength feature matrices are subsequently extracted from each adjacency matrix. As a feature propagation stage, the strength matrix and the adjacency matrix of each k-core subgraph undergo a linear transformation. The generated matrices are averaged across all k-cores. Protein embeddings are learned with a GCN encoder and logically concatenated with the drug embeddings to model protein-drug interactions. The generated feature matrix is then supplied to the RGCN auto-encoder to learn drug embeddings for predicting multi-relational side effects on the multi-relationship DD subgraph. Figure 5.1 illustrates the proposed HC-POSE prediction model.

5.1.5 Experiment

We assess the efficiency of our model on real-world biological POSE datasets presented below.

5.1.5.1 Fourth level title: Only first letter capital

5.1.5.2 Datasets

We utilize a similar biomedical heterogenous dataset as in [124]. In this study, they used the human protein-protein interaction platform [125], the side effect resource database (SIDER) [126], and the Search Tool for Interaction of Chemicals database (STITCH) [127], respectively to extract protein-protein, drug-drug, and the drug-protein interaction datasets respectively. The dataset has 645 drug nodes and 19,081 protein nodes, which are connected by 18,596 drug-protein edges, 4,651,131 drug-drug edges, and 715,612 protein-protein edges. The relationship between a protein and a drug identifies the distinct kind of protein that is targeted by a particular drug. The summary of the dataset descriptions is provided in Table 5.1.

Table 5.1 : The BioSNAP-Decagon datasets' description.

Dataset	Node #	Edge #	Unique Label #	Graph Name
ChChSe-Decagon	645(D)	63473	1317	DD
PP-Decagon	19081(P)	715612	1	PP
GhG-TargetDecagon	3648(P), 284(D)	18690	1	PD

5.1.6 Results discussion.

Table 5.2 summarizes the performance of the proposed HC-POSE and the baseline methods. As observed from Table 5.2, the proposed HC-POSE performs significantly

Table 5.2 : Table of results

Model	AUPRC	AUROC
Decagon [113]	0.8457	0.869
R-GCN [128]	0.876	0.917
DistMult [129]	0.852	0.869
HC-POSE	**0.902**	**0.928**

better than the baseline models. Thanks to the centrality and hierarchical mechanisms in the proposed HC-POSE. In addition, HC-POSE is computationally and time efficient as the k-core computation algorithm has a linear relationship with the number of edges. In addition to maintaining the node's centrality roles, the node strength, a powerful form of node degree centrality, also serves to index global node connectivity information and the overall network topology.

5.1.7 Summary

We introduce HC-POSE, a hierarchical and centrality-driven polypharmacy side effect prediction model that captures not only the node centrality roles but also the network hierarchical network information plus the overall network topology during learning embeddings for POSE. Moreover, HC-POSE captures all the diverse information from various data sub-graphs in a unified model to avoid error accumulations due to decentralized sub-tasks. In the future, in order to more accurately predict the job at hand, we shall aim to capture other network properties, including modularity and the community structure, when embedding learning.

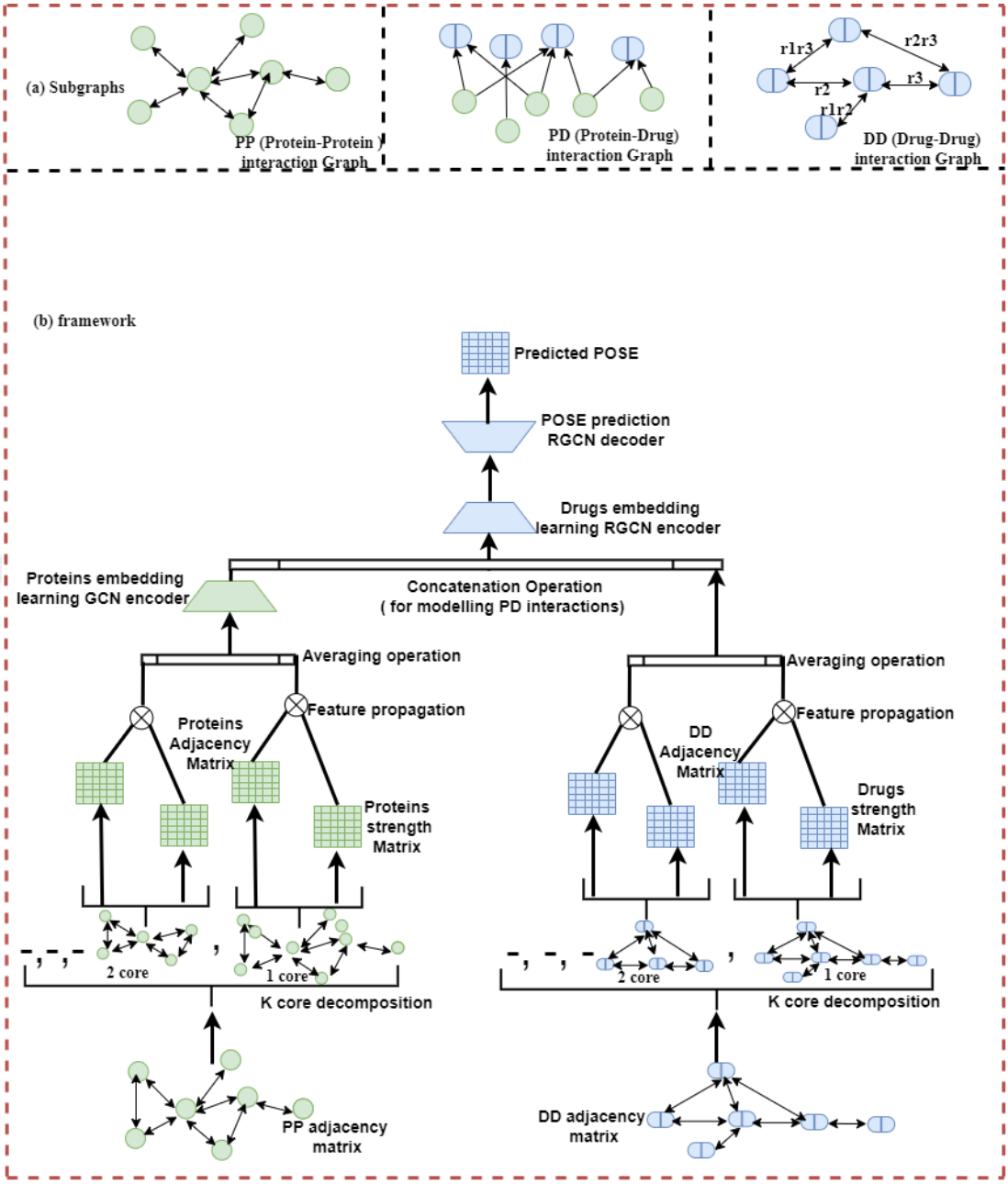

Figure 5.1 : The proposed HC-POSE's framework. The raw PP and DD sub-graphs are subjected to k-core decomposition to uncover the prevailing hierarchical information in them. A node centrality matrix is derived for each generated k-core subgraph and aggregated with the k-core's matching adjacency matrix. Next, the feature matrices across all k-cores are merged by an averaging operation. The GCN encoder is leveraged to learn PP embeddings. Then the resulting feature matrices are systematically concatenated with the DD features matrices and utilized to model PD interactions. The subsequent feature matrix is supplied to the RCGCN auto-encoder to learn embeddings for the multi-type relationship DD subgraph and to predict the polypharmacy side effects

6. CONCLUSIONS

In this thesis, we focus on learning embeddings for graphs while rendering special attention to capturing and preserving the crucial network-characterizing properties. Thus, the thesis intersects well-studied graph theory techniques and geometric deep learning to cover gaps in these arenas. Precisely we cover centrality, topology, and hierarchical-based graph properties. In addition, we explore the prevailing temporal information in evolving real network datasets. To achieve these goals, first, we propose reliable means for capturing such key network-characterizing properties from the raw network interaction datasets; next, we take advantage of the recent geometric deep-learning approaches to learn quality that boosts the accuracy of the network end tasks, including link prediction, node classification, and graph clustering. We explicitly assess the impact of each individual module of the proposed models by introducing model variants lacking such modules so as understand the impact of the module on the mode's final output. We ran experiments on real-world social and biological network datasets to assess the relevancy of the proposed models in solving real-world challenges.

6.1 Final Remarks

Below we summarize the network property-preserving embedding models we have proposed in this dissertation. First, we introduced a topology and centrality hybrid deep learning model that captures topological and centrality-based features from input graphs to preserve the general network topology and nodes' individual structural centrality roles in the learned embeddings. To further enforce the conservation of the node's roles and the general network topology in the learnt embeddings, we imposed a centrality and topological constraint on the model learning loss. We build the model total loss to hold intuitively a reconstruction loss, topological and centrality constraints. The model exhibited a 4% improvement in link prediction AUC. Furthermore, on assessing the efficiency of the model in preserving the node's centrality roles in the

network, notice a 37% average reduction in MSE over the best benchmark. Thanks to the property-based features and the relevant learning constraints that empowered the model. Despite the model's outperforming results for link prediction and node centrality role conservation, the model had large memory and computation time, and it couldn't run on large datasets with our computing resources. This was mainly due to the large parameters associated with the fully connected autoencoders and decoders, in addition to the complex model's loss function encompassing various modules and learning constraints.

Second, in order to cover larger datasets yet with low memory and computation requirements, we proposed a novel structural and topology-driven GCN model (STP-GCN). The model computes a special form of topological similarity convolution matrices together as well as centrality matrices from the raw input dataset. The resulting matrices are aggregated and utilized in a special GCN-based framework. Interestingly the model is very memory and time efficient, even on large datasets yet with promising link prediction AUC and centrality prediction results. On average, each training epoch runs for 2.16s only. Notably, the VAE-based model with fully connected neural networks and a constrained loss function is noted to give slightly better AUC results as compared to the GCN-based model. However, the former could only be implemented on the smallest 3 datasets out of the 5 datasets, yet, the latter can be run conveniently on all 5 datasets, including the largest 2 datasets.

Third, here, we aimed at extending key network-property-based embedding learning in line with geometric deep learning models to complex heterogeneous networks, yet assuring applicability to solving real-life challenges. As a sample case, we considered polypharmacy side effect prediction, where we modeled side effect prediction in drug-drug interactions as a link prediction problem. We then introduced a novel hierarchical and centrality-guided polypharmacy side effect prediction (POSE) prediction model. It basically leverages the k-core decomposition algorithm to capture the prevailing hierarchical information in the POSE datasets. It learns connection in highly connected subgraphs before propagating information to sparsely connected subgraphs. In addition, it utilizes centrality-based features to emphasize the centrality roles of nodes in the learned embeddings. We introduce an intuitive feature aggregation scheme to benefit from the information arising from different subgraphs of the complex

POSE dataset networks. Finally, we leveraged special relation GCN (RGCN) to handle the underlying heterogeneity of the datasets, including the multi-node type (proteins and drug nodes) and the multi-edge types (different side effects). The model exhibited promising results as compared to the benchmark deep learning models.

6.2 Opportunities and Future Works.

Throughout the dissertation, we have explored a few topological similarity measures; we leave a gap for studying features of other powerful topological metrics, including the Local Path Index [57], Average Commute Time [62], the Leicht-Holme-Newman Index [130], Katz Index [61], Preferential Attachment [53], and the Hub Depressed Index [60].

In the same direction, it is worth investigating other centrality measures that are not covered in this study, including the rich Eigen vector Centrality [131] that determines the centrality of a node based on its neighbors' centrality and Page rank centrality [132], that suits directed networks. On the other hand, by following our approach, there are open opportunities for exploring other network properties, including the graph community structure [133, 134], modularity [135] and graph diffusion [136–138].

Last but not least, our network property (centrality, topology, and hierarchical)-aware embedding learning models can be flexibly integrated with data augmenting modules like graph translators, network generators, and graph adversarial networks to synthesize network property-sound graphs. This could help minimize the scarcity challenges of clinical datasets, including brain connectomes, drug-drug interactions, and gene-gene interaction network datasets. Moreover, we have defined means (parameters) for regulating the weight attached to network properties like centrality or topological information. This could help customize data during augmentation to retain the property desired most.

REFERENCES

[1] **Sserwadda, A.**, **Ozcan, A. and Yaslan, Y.** (2023). Topological Similarity and Centrality Driven Hybrid Deep Learning for Temporal Link Prediction, *JUCS - Journal of Universal Computer Science*, *29*(5), 470–490.

[2] **Bhagat, S.**, **Cormode, G. and Muthukrishnan, S.** (2011). *Node Classification in Social Networks*, Springer US, Boston, MA.

[3] **Xu, H.**, **Yang, Y.**, **Wang, L. and Liu, W.** (2013). Node classification in social network via a factor graph model, *Advances in Knowledge Discovery and Data Mining: 17th Pacific-Asia Conference, PAKDD 2013, Gold Coast, Australia, April 14-17, 2013, Proceedings, Part I 17*, Springer, pp.213–224.

[4] **Tang, J.**, **Aggarwal, C. and Liu, H.** (2016). Node classification in signed social networks, *Proceedings of the 2016 SIAM international conference on data mining*, SIAM, pp.54–62.

[5] **Ebeid, I.A.**, **Talburt, J.R. and Siddique, M.A.S.** (2022). Graph-based hierarchical record clustering for unsupervised entity resolution, *ITNG 2022 19th International Conference on Information Technology-New Generations*, Springer, pp.107–118.

[6] **Zhou, Y.**, **Cheng, H. and Yu, J.X.** (2009). Graph clustering based on structural/attribute similarities, *Proceedings of the VLDB Endowment*, *2*(1), 718–729.

[7] **Lü, L. and Zhou, T.** (2011). Link prediction in complex networks: A survey, *Physica A: statistical mechanics and its applications*, *390*(6), 1150–1170.

[8] **Kumar, A.**, **Singh, S.S.**, **Singh, K. and Biswas, B.** (2020). Link prediction techniques, applications, and performance: A survey, *Physica A: Statistical Mechanics and its Applications*, *553*, 124289.

[9] **Al Hasan, M.**, **Chaoji, V.**, **Salem, S. and Zaki, M.** (2006). Link prediction using supervised learning, *SDM06: workshop on link analysis, counter-terrorism and security*, volume 30, pp.798–805.

[10] **Zhang, M. and Chen, Y.** (2018). Link prediction based on graph neural networks, *Advances in neural information processing systems*, *31*.

[11] **Freeman, L.C.**, **Borgatti, S.P. and White, D.R.** (1991). Centrality in valued graphs: A measure of betweenness based on network flow, *Social networks*, *13*(2), 141–154.

[12] **Rodrigues, F.A.** (2019). Network centrality: an introduction, *A mathematical modeling approach from nonlinear dynamics to complex systems*, 177–196.

[13] **Kim, H. and Anderson, R.** (2012). Temporal node centrality in complex networks, *Physical Review E*, *85*(2), 026107.

[14] **Bringmann, L.F.**, **Elmer, T.**, **Epskamp, S.**, **Krause, R.W.**, **Schoch, D.**, **Wichers, M.**, **Wigman, J.T. and Snippe, E.** (2019). What do centrality measures measure in psychological networks?, *Journal of abnormal psychology*, *128*(8), 892.

[15] **Gao, Z.**, **Shi, Y. and Chen, S.** (2015). Measures of node centrality in mobile social networks, *International Journal of Modern Physics C*, *26*(09), 1550107.

[16] **Huang, S.**, **Lv, T.**, **Zhang, X.**, **Yang, Y.**, **Zheng, W. and Wen, C.** (2014). Identifying node role in social network based on multiple indicators, *PloS one*, *9*(8), e103733.

[17] **Sserwadda, A. and Rekik, I.** (2021). Topology-guided cyclic brain connectivity generation using geometric deep learning, *Journal of Neuroscience Methods*, *353*, 108988.

[18] **Nobre, G.P.**, **Ferreira, C.H. and Almeida, J.M.** (2022). A hierarchical network-oriented analysis of user participation in misinformation spread on WhatsApp, *Information Processing & Management*, *59*(1), 102757.

[19] **Kwon, H.**, **Choi, Y.H. and Lee, J.M.** (2019). A physarum centrality measure of the human brain network, *Scientific reports*, *9*(1), 1–13.

[20] **Liu, W.**, **Li, X.**, **Liu, T. and Liu, B.** (2019). Approximating betweenness centrality to identify key nodes in a weighted urban complex transportation network, *Journal of Advanced Transportation*, *2019*.

[21] **Bucur, D. and Holme, P.** (2020). Beyond ranking nodes: Predicting epidemic outbreak sizes by network centralities, *PLOS Computational Biology*, *16*(7), e1008052.

[22] **Aleskerov, F. and Shvydun, S.** (2018). Stability and similarity in networks based on topology and nodes importance, *International Conference on Complex Networks and their Applications*, Springer, pp.94–103.

[23] **Ibarra, H.** (1993). Network centrality, power, and innovation involvement: Determinants of technical and administrative roles, *Academy of Management journal*, *36*(3), 471–501.

[24] **Kuzubaş, T.U.**, **Ömercikoğlu, I. and Saltoğlu, B.** (2014). Network centrality measures and systemic risk: An application to the Turkish financial crisis, *Physica A: Statistical Mechanics and its Applications*, *405*, 203–215.

[25] **Meng, X.**, **Li, W.**, **Peng, X.**, **Li, Y. and Li, M.** (2021). Protein interaction networks: centrality, modularity, dynamics, and applications, *Frontiers of Computer Science*, *15*(6), 1–17.

[26] **Zhuge, H. and Zhang, J.** (2010). Topological centrality and its e-Science applications, *Journal of the American Society for Information Science and Technology*, *61*(9), 1824–1841.

[27] **Jain, A. and Reddy, B.** (2013). Node centrality in wireless sensor networks: Importance, applications and advances, *2013 3rd IEEE International Advance Computing Conference (IACC)*, IEEE, pp.127–131.

[28] **Surendran, R. and Thomas, T.** (2022). Detection of malware applications from centrality measures of syscall graph, *Concurrency and Computation: Practice and Experience*, *34*(10), e6835.

[29] **Irwin, M.D. and Hughes, H.L.** (1992). Centrality and the structure of urban interaction: measures, concepts, and applications, *Social Forces*, *71*(1), 17–51.

[30] **Dablander, F. and Hinne, M.** (2019). Node centrality measures are a poor substitute for causal inference, *Scientific reports*, *9*(1), 6846.

[31] **Li, C.**, **Li, Q.**, **Van Mieghem, P.**, **Stanley, H.E. and Wang, H.** (2015). Correlation between centrality metrics and their application to the opinion model, *The European Physical Journal B*, *88*, 1–13.

[32] **Moghadam, H.E.**, **Mohammadi, T.**, **Kashani, M.F. and Shakeri, A.** (2019). Complex networks analysis in Iran stock market: The application of centrality, *Physica A: Statistical Mechanics and its Applications*, *531*, 121800.

[33] **Coşkun, M. and Koyutürk, M.** (2021). Node similarity-based graph convolution for link prediction in biological networks, *Bioinformatics*, *37*(23), 4501–4508.

[34] **Li, S.**, **Huang, J.**, **Zhang, Z.**, **Liu, J.**, **Huang, T. and Chen, H.** (2018). Similarity-based future common neighbors model for link prediction in complex networks, *Scientific reports*, *8*(1), 1–11.

[35] **Rezaeipanah, A. and Mojarad, M.** (2019). Link prediction in social networks using the extraction of graph topological features, *Int J Sci Res Netw Secur Commun*, *7*(5), 1–7.

[36] **Wang, C.**, **Satuluri, V. and Parthasarathy, S.** (2007). Local probabilistic models for link prediction, *Seventh IEEE international conference on data mining (ICDM 2007)*, IEEE, pp.322–331.

[37] **Muniz, C.P.**, **Goldschmidt, R. and Choren, R.** (2018). Combining contextual, temporal and topological information for unsupervised link prediction in social networks, *Knowledge-Based Systems*, *156*, 129–137.

[38] **Lei, C. and Ruan, J.** (2013). A novel link prediction algorithm for reconstructing protein–protein interaction networks by topological similarity, *Bioinformatics*, *29*(3), 355–364.

[39] **Zhou, T.** (2021). Progresses and challenges in link prediction, *Iscience*, *24*(11), 103217.

[40] **Monti, F.**, **Boscaini, D.**, **Masci, J.**, **Rodola, E.**, **Svoboda, J. and Bronstein, M.M.** (2017). Geometric deep learning on graphs and manifolds using mixture model cnns, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp.5115–5124.

[41] **Cao, W.**, **Yan, Z.**, **He, Z. and He, Z.** (2020). A comprehensive survey on geometric deep learning, *IEEE Access*, *8*, 35929–35949.

[42] **Bronstein, M.M.**, **Bruna, J.**, **LeCun, Y.**, **Szlam, A. and Vandergheynst, P.** (2017). Geometric deep learning: going beyond euclidean data, *IEEE Signal Processing Magazine*, *34*(4), 18–42.

[43] **Cao, W.**, **Zheng, C.**, **Yan, Z. and Xie, W.** (2022). Geometric deep learning: progress, applications and challenges, *Science China Information Sciences*, *65*(2), 126101.

[44] **Bronstein, M.M.**, **Bruna, J.**, **Cohen, T. and Veličković, P.** (2021). Geometric deep learning: Grids, groups, graphs, geodesics, and gauges, *arXiv preprint arXiv:2104.13478*.

[45] **Kipf, T.N. and Welling, M.** (2016). Semi-supervised classification with graph convolutional networks, *arXiv preprint arXiv:1609.02907*.

[46] **Zhang, S.**, **Tong, H.**, **Xu, J. and Maciejewski, R.** (2019). Graph convolutional networks: a comprehensive review, *Computational Social Networks*, *6*(1), 1–23.

[47] **Wu, F.**, **Souza, A.**, **Zhang, T.**, **Fifty, C.**, **Yu, T. and Weinberger, K.** (2019). Simplifying graph convolutional networks, *International conference on machine learning*, PMLR, pp.6861–6871.

[48] **Chen, M.**, **Wei, Z.**, **Huang, Z.**, **Ding, B. and Li, Y.** (2020). Simple and deep graph convolutional networks, *International conference on machine learning*, PMLR, pp.1725–1735.

[49] **Yang, Y.**, **Feng, Z.**, **Song, M. and Wang, X.** (2020). Factorizable graph convolutional networks, *Advances in Neural Information Processing Systems*, *33*, 20286–20296.

[50] **Gao, H.**, **Wang, Z. and Ji, S.** (2018). Large-scale learnable graph convolutional networks, *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp.1416–1424.

[51] **Zhang, S.**, **Tong, H.**, **Xu, J. and Maciejewski, R.** (2018). Graph convolutional networks: Algorithms, applications and open challenges, *Computational Data and Social Networks: 7th International Conference, CSoNet 2018,*

Shanghai, China, December 18–20, 2018, Proceedings 7, Springer, pp.79–91.

[52] **Pei, H.**, **Wei, B.**, **Chang, K.C.C.**, **Lei, Y. and Yang, B.** (2020). Geom-gcn: Geometric graph convolutional networks, *arXiv preprint arXiv:2002.05287*.

[53] **Newman, M.E.** (2001). Clustering and preferential attachment in growing networks, *Physical review E*, *64*(2), 025102.

[54] **Adamic, L.A. and Adar, E.** (2003). Friends and neighbors on the web, *Social networks*, *25*(3), 211–230.

[55] **Jaccard, P.** (1912). The distribution of the flora in the alpine zone. 1, *New phytologist*, *11*(2), 37–50.

[56] **Salton, G. and McGill, M.J.** (1983). *Introduction to modern information retrieval*, mcgraw-hill.

[57] **Zhou, T.**, **Lü, L. and Zhang, Y.C.** (2009). Predicting missing links via local information, *The European Physical Journal B*, *71*(4), 623–630.

[58] **Sorensen, T.A.** (1948). A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on Danish commons, *Biol. Skar.*, *5*, 1–34.

[59] **Geethanjali, P.** (2015). *Fundamentals of Brain Signals and Its Medical Application Using Data Analysis Techniques*, Springer.

[60] **Ravasz, E.**, **Somera, A.L.**, **Mongru, D.A.**, **Oltvai, Z.N. and Barabási, A.L.** (2002). Hierarchical organization of modularity in metabolic networks, *science*, *297*(5586), 1551–1555.

[61] **Katz, L.** (1953). A new status index derived from sociometric analysis, *Psychometrika*, *18*(1), 39–43.

[62] **Fouss, F.**, **Pirotte, A.**, **Renders, J.M. and Saerens, M.** (2007). Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation, *IEEE Transactions on knowledge and data engineering*, *19*(3), 355–369.

[63] **Brin, S. and Page, L.** (1998). The anatomy of a large-scale hypertextual web search engine, *Computer networks and ISDN systems*, *30*(1-7), 107–117.

[64] **Chebotarev, P. and Shamis, E.** (2006). The matrix-forest theorem and measuring relations in small social groups, *arXiv preprint math/0602070*.

[65] **Alvarez-Hamelin, J.**, **Dall'Asta, L.**, **Barrat, A. and Vespignani, A.** (2005). Large scale networks fingerprinting and visualization using the k-core decomposition, *Advances in neural information processing systems*, *18*.

[66] **Alvarez-Hamelin, J.I.**, **Dall'Asta, L.**, **Barrat, A. and Vespignani, A.** (2005). K-core decomposition of internet graphs: hierarchies, self-similarity and measurement biases, *arXiv preprint cs/0511007*.

[67] **Lahav, N.**, **Ksherim, B.**, **Ben-Simon, E.**, **Maron-Katz, A.**, **Cohen, R. and Havlin, S.** (2016). K-shell decomposition reveals hierarchical cortical organization of the human brain, *New Journal of Physics*, *18*(8), 083013.

[68] **Li, Z.**, **Lu, Y.**, **Zhang, W.P.**, **Li, R.H.**, **Guo, J.**, **Huang, X. and Mao, R.** (2018). Discovering hierarchical subgraphs of k-core-truss, *Data Science and Engineering*, *3*(2), 136–149.

[69] **Kong, Y.X.**, **Shi, G.Y.**, **Wu, R.J. and Zhang, Y.C.** (2019). k-core: Theories and applications, *Physics Reports*, *832*, 1–32.

[70] **Liu, J.**, **Xu, C.**, **Yin, C.**, **Wu, W. and Song, Y.** (2020). K-core based temporal graph convolutional network for dynamic graphs, *IEEE Transactions on Knowledge and Data Engineering*.

[71] **Qin, L.**, **Wang, Y.**, **Sun, Q.**, **Zhang, X.**, **Shia, B.C.**, **Liu, C.** *et al.* (2020). Analysis of the covid-19 epidemic transmission network in mainland china: K-core decomposition study, *JMIR public health and surveillance*, *6*(4), e24291.

[72] **Filho, H.A.**, **Machicao, J. and Bruno, O.M.** (2018). A hierarchical model of metabolic machinery based on the k core decomposition of plant metabolic networks, *PloS one*, *13*(5), e0195843.

[73] **Aghdam, M.H. and Zanjani, M.D.** (2021). A novel regularized asymmetric non-negative matrix factorization for text clustering, *Information Processing & Management*, *58*(6), 102694.

[74] **Yang, J.**, **Yang, S.**, **Fu, Y.**, **Li, X. and Huang, T.** (2008). Non-negative graph embedding, *2008 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, pp.1–8.

[75] **Zhang, B.**, **Gong, M.**, **Huang, J. and Ma, X.** (2021). Clustering heterogeneous information network by joint graph embedding and nonnegative matrix factorization, *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *15*(4), 1–25.

[76] **Xing, Y.**, **Hu, L.**, **Zhang, X.**, **Wu, G. and Wang, F.** (2021). Nonnegative Matrix Factorization Based Heterogeneous Graph Embedding Method for Trigger-Action Programming in IoT, *IEEE Transactions on Industrial Informatics*, *18*(2), 1231–1239.

[77] **Zhu, L.**, **Guo, D.**, **Yin, J.**, **Steeg, G.V. and Galstyan, A.** (2016). Scalable Temporal Latent Space Inference for Link Prediction in Dynamic Social Networks, *IEEE Transactions on Knowledge and Data Engineering*, *28*(10), 2765–2777.

[78] **Yu, W.**, **Cheng, W.**, **Aggarwal, C.C.**, **Chen, H. and Wang, W.** (2017). Link prediction with spatial and temporal consistency in dynamic networks., *IJCAI*, pp.3343–3349.

[79] **Tenenbaum, J.B.**, **Silva, V.d. and Langford, J.C.** (2000). A global geometric framework for nonlinear dimensionality reduction, *science*, *290*(5500), 2319–2323.

[80] **Qiu, J.**, **Dong, Y.**, **Ma, H.**, **Li, J.**, **Wang, K. and Tang, J.** (2018). Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec, *Proceedings of the eleventh ACM international conference on web search and data mining*, pp.459–467.

[81] **Huang, S.**, **Zhang, Y.**, **Fu, L. and Wang, S.** (2022). Learnable multi-view matrix factorization with graph embedding and flexible loss, *IEEE Transactions on Multimedia*.

[82] **Agibetov, A.** (2023). Neural graph embeddings as explicit low-rank matrix factorization for link prediction, *Pattern Recognition*, *133*, 108977.

[83] **Perozzi, B.**, **Al-Rfou, R. and Skiena, S.** (2014). Deepwalk: Online learning of social representations, *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp.701–710.

[84] **Grover, A. and Leskovec, J.** (2016). node2vec: Scalable feature learning for networks, *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp.855–864.

[85] **Chen, H.**, **Perozzi, B.**, **Hu, Y. and Skiena, S.** (2018). Harp: Hierarchical representation learning for networks, *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

[86] **Ahmed, N.K.**, **Rossi, R.**, **Lee, J.B.**, **Willke, T.L.**, **Zhou, R.**, **Kong, X. and Eldardiry, H.** (2018). Learning role-based graph embeddings, *arXiv preprint arXiv:1802.02896*.

[87] **Schlötterer, J.**, **Wehking, M.**, **Rizi, F.S. and Granitzer, M.** (2019). Investigating extensions to random walk based graph embedding, *2019 IEEE International Conference on Cognitive Computing (ICCC)*, IEEE, pp.81–89.

[88] **Rahman, T.**, **Surma, B.**, **Backes, M. and Zhang, Y.** (2019). Fairwalk: Towards Fair Graph Embedding, *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, AAAI Press, p.3289–3295.

[89] **Yue, X.**, **Wang, Z.**, **Huang, J.**, **Parthasarathy, S.**, **Moosavinasab, S.**, **Huang, Y.**, **Lin, S.M.**, **Zhang, W.**, **Zhang, P. and Sun, H.** (2020). Graph embedding on biomedical networks: methods, applications and evaluations, *Bioinformatics*, *36*(4), 1241–1251.

[90] **Nguyen, G.H.**, **Lee, J.B.**, **Rossi, R.A.**, **Ahmed, N.K.**, **Koh, E. and Kim, S.** (2018). Dynamic network embeddings: From random walks to temporal random walks, *2018 IEEE International Conference on Big Data (Big Data)*, IEEE, pp.1085–1092.

[91] **Gao, M.**, **Chen, L.**, **He, X. and Zhou, A.** (2018). Bine: Bipartite network embedding, *The 41st international ACM SIGIR conference on research & development in information retrieval*, pp.715–724.

[92] **Chen, J.**, **Zhang, Q. and Huang, X.** (2016). Incorporate group information to enhance network embedding, *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pp.1901–1904.

[93] **Huang, Z.**, **Silva, A. and Singh, A.** (2021). A broader picture of random-walk based graph embedding, *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pp.685–695.

[94] **Pareja, A.**, **Domeniconi, G.**, **Chen, J.**, **Ma, T.**, **Suzumura, T.**, **Kanezashi, H.**, **Kaler, T.**, **Schardl, T. and Leiserson, C.** (2020). Evolvegcn: Evolving graph convolutional networks for dynamic graphs, *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp.5363–5370.

[95] **Veličković, P.**, **Cucurull, G.**, **Casanova, A.**, **Romero, A.**, **Lio, P. and Bengio, Y.** (2017). Graph attention networks, *arXiv preprint arXiv:1710.10903*.

[96] **Chen, J.**, **Ma, T. and Xiao, C.** (2018). Fastgcn: fast learning with graph convolutional networks via importance sampling, *arXiv preprint arXiv:1801.10247*.

[97] **Hamilton, W.L.**, **Ying, R. and Leskovec, J.** (2017). Inductive representation learning on large graphs, *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp.1025–1035.

[98] **He, X.**, **Deng, K.**, **Wang, X.**, **Li, Y.**, **Zhang, Y. and Wang, M.** (2020). Lightgcn: Simplifying and powering graph convolution network for recommendation, *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pp.639–648.

[99] **Geng, X.**, **Li, Y.**, **Wang, L.**, **Zhang, L.**, **Yang, Q.**, **Ye, J. and Liu, Y.** (2019). Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting, *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp.3656–3663.

[100] **Min, Y.**, **Wenkel, F. and Wolf, G.** (2020). Scattering gcn: Overcoming oversmoothness in graph convolutional networks, *Advances in neural information processing systems*, *33*, 14498–14508.

[101] **Chen, J.**, **Zhang, J.**, **Xu, X.**, **Fu, C.**, **Zhang, D.**, **Zhang, Q. and Xuan, Q.** (2019). E-lstm-d: A deep learning framework for dynamic network link prediction, *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.

[102] **Hochreiter, S. and Schmidhuber, J.** (1997). Long short-term memory, *Neural computation*, *9*(8), 1735–1780.

[103] **Sserwadda, A.**, **Ozcan, A. and Yaslan, Y.** (2023). Structural and topological guided GCN for link prediction in temporal networks, *Journal of Ambient Intelligence and Humanized Computing*, 1–9.

[104] **Kingma, D.P. and Ba, J.** (2015). Adam: A Method for Stochastic Optimization, ***Y. Bengio and Y. LeCun***, *editors, 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, volume 3.

[105] **Xu, K.**, **Hu, W.**, **Leskovec, J. and Jegelka, S.** (2018). How powerful are graph neural networks?, *arXiv preprint arXiv:1810.00826*.

[106] **Fix, E. and Hodges Jr, J.** (1955). Significance probabilities of the Wilcoxon test, *The Annals of Mathematical Statistics*, 301–312.

[107] **Goyal, P.**, **Chhetri, S.R. and Canedo, A.M.**, (2020), Capturing network dynamics using dynamic graph representation learning, uS Patent App. 16/550,771.

[108] **Wang, D.**, **Cui, P. and Zhu, W.** (2016). Structural deep network embedding, *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp.1225–1234.

[109] **Shaw, B. and Jebara, T.** (2009). Structure preserving embedding, *Proceedings of the 26th Annual International Conference on Machine Learning*, pp.937–944.

[110] **Bessadok, A.**, **Mahjoub, M.A. and Rekik, I.** (2020). Topology-aware generative adversarial network for joint prediction of multiple brain graphs from a single brain graph, *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, pp.551–561.

[111] **Goyal, P.**, **Kamra, N.**, **He, X. and Liu, Y.** (2018). Dyngem: Deep embedding method for dynamic graphs, *3rd International Workshop on Representation Learning for Graphs (ReLiG)*, *1805*.

[112] **Barabási, A.L. and Albert, R.** (1999). Emergence of scaling in random networks, *science*, *286*(5439), 509–512.

[113] **Zitnik, M.**, **Agrawal, M. and Leskovec, J.** (2018). Modeling polypharmacy side effects with graph convolutional networks, *Bioinformatics*, *34*(13), i457–i466.

[114] **Lee, C.Y. and Chen, Y.P.P.** (2021). Descriptive prediction of drug side-effects using a hybrid deep learning model, *International Journal of Intelligent Systems*, *36*(6), 2491–2510.

[115] **Uner, O.C.**, **Gokberk Cinbis, R.**, **Tastan, O. and Cicek, A.E.** (2019). DeepSide: a deep learning framework for drug side effect prediction, *Biorxiv*, 843029.

[116] **Kumar Shukla, P.**, **Kumar Shukla, P.**, **Sharma, P.**, **Rawat, P.**, **Samar, J.**, **Moriwal, R. and Kaur, M.** (2020). Efficient prediction of drug–drug interaction using deep learning models, *IET Systems Biology*, *14*(4), 211–216.

[117] **Lee, G.**, **Park, C. and Ahn, J.** (2019). Novel deep learning model for more accurate prediction of drug-drug interaction effects, *BMC bioinformatics*, *20*(1), 1–8.

[118] **Wen, M.**, **Zhang, Z.**, **Niu, S.**, **Sha, H.**, **Yang, R.**, **Yun, Y. and Lu, H.** (2017). Deep-learning-based drug–target interaction prediction, *Journal of proteome research*, *16*(4), 1401–1409.

[119] **Ryu, J.Y.**, **Kim, H.U. and Lee, S.Y.** (2018). Deep learning improves prediction of drug–drug and drug–food interactions, *Proceedings of the National Academy of Sciences*, *115*(18), E4304–E4311.

[120] **Pang, S.**, **Zhang, Y.**, **Song, T.**, **Zhang, X.**, **Wang, X. and Rodriguez-Patón, A.** (2022). AMDE: a novel attention-mechanism-based multidimensional feature encoder for drug–drug interaction prediction, *Briefings in Bioinformatics*, *23*(1), bbab545.

[121] **Xu, H.**, **Sang, S. and Lu, H.** (2020). Tri-graph information propagation for polypharmacy side effect prediction, *arXiv preprint arXiv:2001.10516*.

[122] **Yu, Y.**, **Huang, K.**, **Zhang, C.**, **Glass, L.M.**, **Sun, J. and Xiao, C.** (2021). SumGNN: multi-typed drug interaction prediction via efficient knowledge graph summarization, *Bioinformatics*, *37*(18), 2988–2995.

[123] **Sserwadda, A.**, **Ozcan, A. and Yaslan, Y.** (2022). Hierarchical and Centrality driven Polypharmacy Side Effect Prediction Model, *2022 7th International Conference on Computer Science and Engineering (UBMK)*, IEEE, pp.384–387.

[124] **Zitnik, M.**, **Sosic, R. and Leskovec, J.** (2018). BioSNAP Datasets: Stanford biomedical network dataset collection, *Note: http://snap. stanford. edu/biodata Cited by*, *5*(1).

[125] **Menche, J.**, **Sharma, A.**, **Kitsak, M.**, **Ghiassian, S.D.**, **Vidal, M.**, **Loscalzo, J. and Barabási, A.L.** (2015). Uncovering disease-disease relationships through the incomplete interactome, *Science*, *347*(6224), 1257601.

[126] **Kuhn, M.**, **Letunic, I.**, **Jensen, L.J. and Bork, P.** (2016). The SIDER database of drugs and side effects, *Nucleic acids research*, *44*(D1), D1075–D1079.

[127] **Szklarczyk, D.**, **Santos, A.**, **Von Mering, C.**, **Jensen, L.J.**, **Bork, P. and Kuhn, M.** (2016). STITCH 5: augmenting protein–chemical interaction networks with tissue and affinity data, *Nucleic acids research*, *44*(D1), D380–D384.

[128] **Schlichtkrull, M.**, **Kipf, T.N.**, **Bloem, P.**, **Berg, R.v.d.**, **Titov, I. and Welling, M.** (2018). Modeling relational data with graph convolutional networks, *European semantic web conference*, Springer, pp.593–607.

[129] **Yang, B.**, **Yih, W.t.**, **He, X.**, **Gao, J. and Deng, L.** (2014). Embedding entities and relations for learning and inference in knowledge bases, *arXiv preprint arXiv:1412.6575*.

[130] **Leicht, E.A.**, **Holme, P. and Newman, M.E.** (2006). Vertex similarity in networks, *Physical Review E*, *73*(2), 026120.

[131] **Wasserman, S. and Faust, K.** (1994). *Social network analysis: Methods and applications*, Cambridge university press.

[132] **Sullivan, D.** (2007). What is google pagerank? A guide for searchers & webmasters, *Search engine land*.

[133] **Fortunato, S. and Castellano, C.** (2007). Community structure in graphs, *arXiv preprint arXiv:0712.2716*.

[134] **Chattopadhyay, S. and Ganguly, D.** (2020). Community Structure aware Embedding of Nodes in a Network, *arXiv preprint arXiv:2006.15313*.

[135] **Bordier, C.**, **Nicolini, C. and Bifone, A.** (2017). Graph analysis and modularity of brain functional connectivity networks: searching for the optimal threshold, *Frontiers in neuroscience*, *11*, 441.

[136] **Atwood, J. and Towsley, D.** (2016). Diffusion-convolutional neural networks, *Advances in neural information processing systems*, *29*.

[137] **Gasteiger, J.**, **Weißenberger, S. and Günnemann, S.** (2019). Diffusion improves graph learning, *Advances in neural information processing systems*, *32*.

[138] **Zhao, J.**, **Dong, Y.**, **Ding, M.**, **Kharlamov, E. and Tang, J.** (2021). Adaptive diffusion in graph neural networks, *Advances in Neural Information Processing Systems*, *34*, 23321–23333.

[139] **Bavelas, A.** (1950). Communication patterns in task-oriented groups, *The journal of the acoustical society of America*, *22*(6), 725–730.

[140] **Freeman, L.C.** (1977). A set of measures of centrality based on betweenness, *Sociometry*, 35–41.

[141] **Vaswani, A.**, **Shazeer, N.**, **Parmar, N.**, **Uszkoreit, J.**, **Jones, L.**, **Gomez, A.N.**, **Kaiser, Ł. and Polosukhin, I.** (2017). Attention is all you need, *Advances in neural information processing systems*, *30*.

APPENDICES

APPENDIX A : Discussion of supplementary results for the proposed TSC-TLP model

APPENDIX B : Other computational formulae for used algorithms

APPENDIX A : Discussion of supplementary results for the proposed TSC-TLP model

APPENDIX A.1: Assessment of Link prediction AUC based on the various Topological similarity metric features for TSC-TLP model

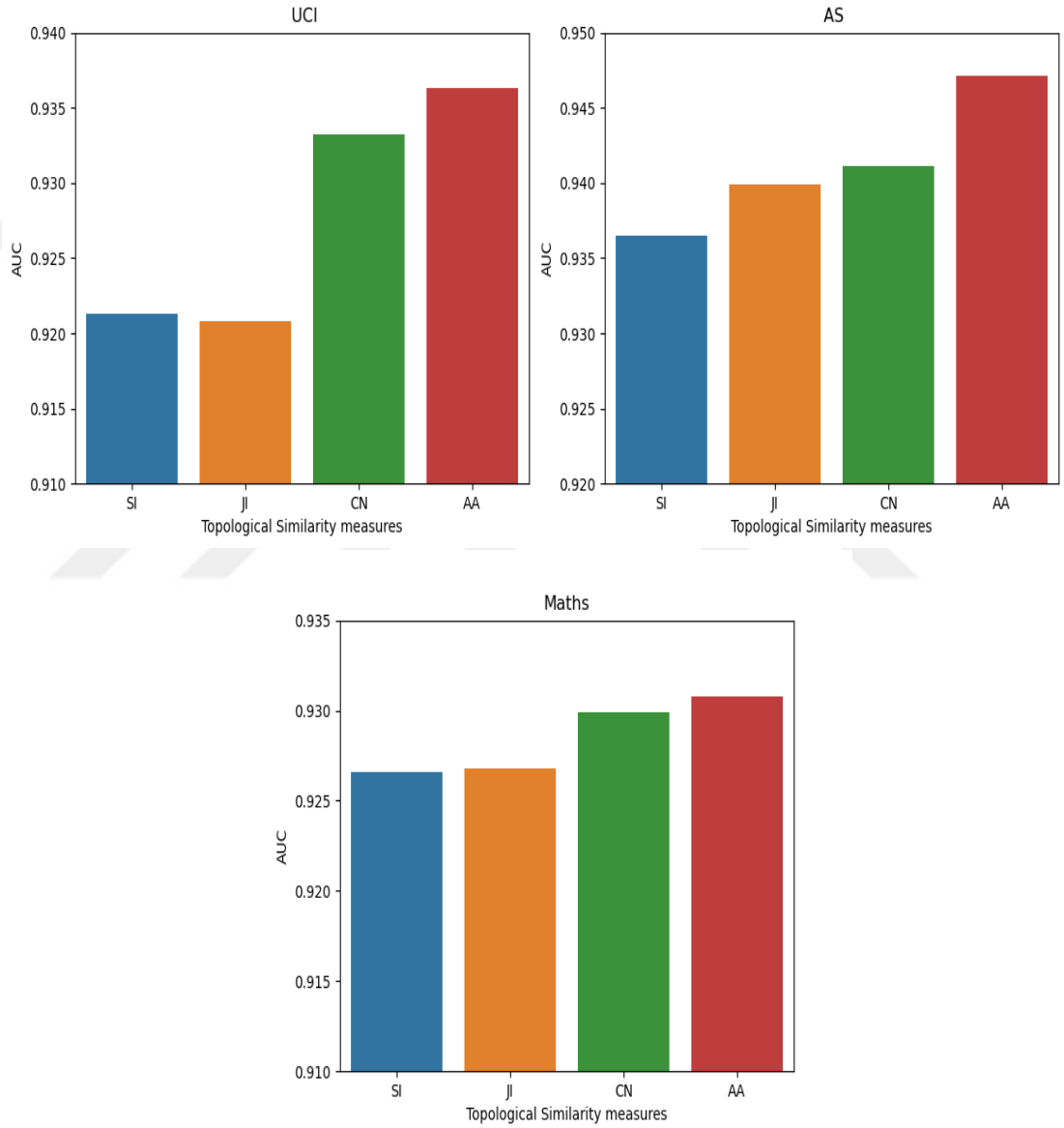

Figure A.1 : AUC link prediction scores for the various datasets based on the features derived by the various topological similarity metrics

The AUC scores from the various topological similarity measures used, including Adamic-Ader (AA), Jaccard Index (JI), Common neighbor (CN), and Salton Index (SI), are shown in Fig.A.1. Note that we report average scores on all graphs for all time stamps. In general, Adamic-adar and Common neighbor yielded the best

results. Thus, we used Adamic-Adar for the remainder of our tests, with the AUC link prediction results shown in Table 4.3. However, we noticed that the choice of topological similarity metric to deploy depends on the dataset. For instance, on some graphs, Common neighbor could perform better than Adamic adder and so apply to other metrics. As another example, the Jaccard index is observed to outperform the Salton index on only 2 of the 3 datasets.

APPENDIX A.2: Examining the conservation of structural Centrality roles of nodes in predicted graphs

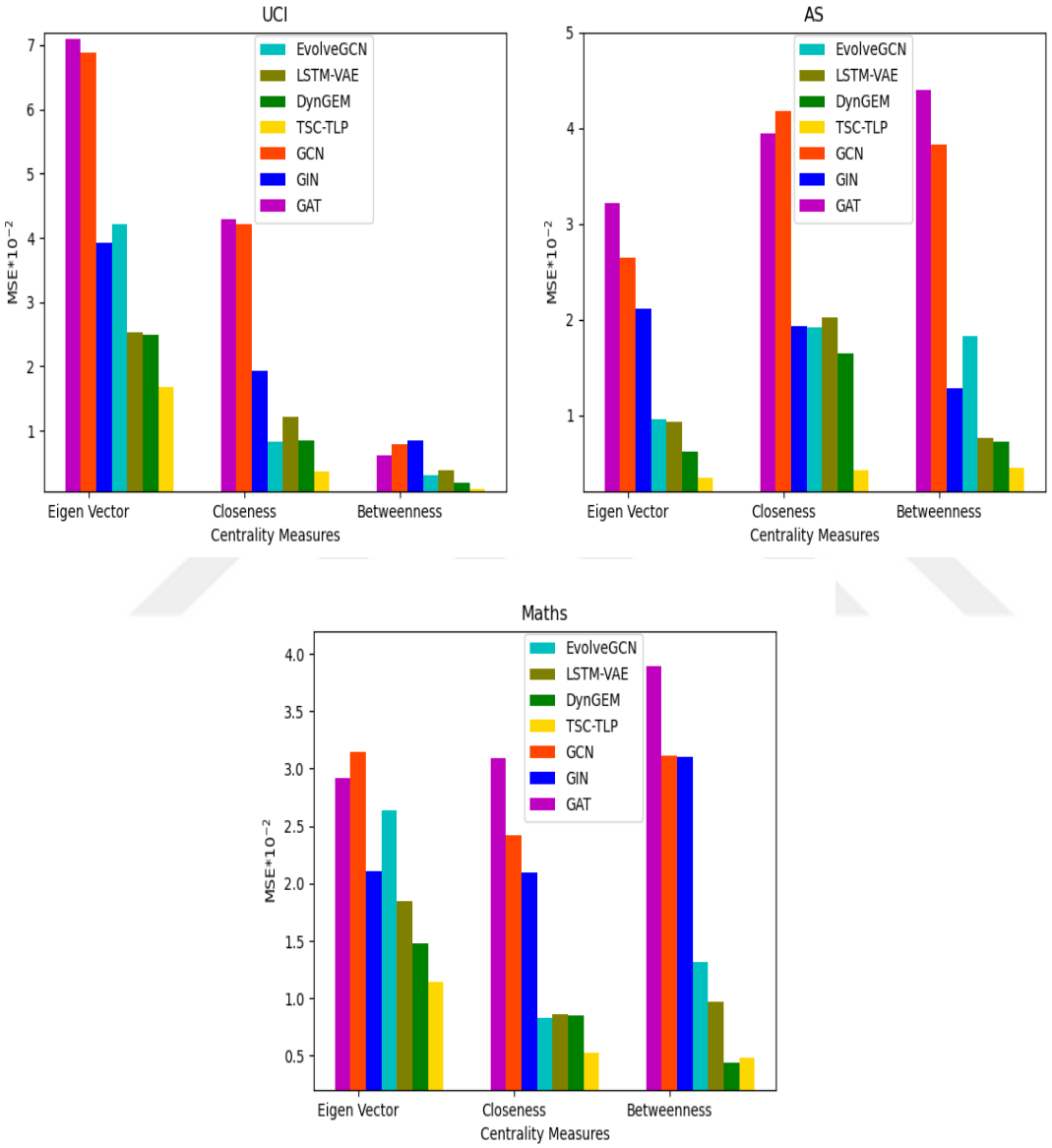

Figure A.2 : Graph centrality prediction MSE score on graphs predicted by the various methods across all datasets.

We compute the MSE between the centralities of nodes in the ground truth test graphs and the same nodes' centralities in predicted graphs to assess the models' capabilities in conserving the nodes' structural roles in their generated embeddings. The average MSE centrality scores of the graphs predicted using embeddings learned

by the different methods across all datasets are presented in Fig. A.2. We evaluated the predicted graphs' nodes using three key centrality metrics including:

- Closeness centrality [139]: This centrality metric estimates the network's information flow speed if the information is routed through the reference node.
- Eigen Vector Centrality [131]: A global version of degree centrality for determining the influence of a node in a network. It scores a node based on the scores of its neighbors. A node has a high eigenvector centrality score if it has links to numerous nodes that also have high scores.
- Betweenness Centrality [140]. It quantifies the frequency with which a node appears on all shortest pathways between any two nodes in a network. Thus, it measures each node's contribution to bridging across nodes and subnetworks.

As seen in Fig. A.2, for all centrality measures on all datasets, the proposed TSC-TPL records the lowest MSE, except for the betweenness centrality prediction on the Maths dataset case, where DynGEM performs somewhat better. TSC-TLP achieves MSE cuts of 59%, 30%, and 12% on the AS, UCI, and Maths datasets, respectively (a reduction of 37% on average) compared to the best benchmark, (DynGEM). This demonstrates how effectively the proposed TSC-TLP conserves the nodes' structural centrality roles during embedding learning. Thanks to the TSC-TLP's centrality and topology-based features as well as the respective network property-based constraints imposed on the model while learning embeddings.

APPENDIX B : Other computational formulae for used algorithms

Recall: To learn node embeddings, the GCN combines the use of local graph structure (i.e., graph connectivity information) and the node-level features. A typical graph convolution operation generates a normalized sum of the node features of its immediate neighbors. Then a reference node is included by adding a self-loop. The GCN representation of the node i at $(l+1)^{th}$ layer is defined as in Eq. B.1

$$h_i^{l+1} = \sigma\left(\sum_{j \in N_i} \frac{1}{c_{ij}} W^{(l)} h_j^{(l)}\right) \tag{B.1}$$

$c_{ij} = \sqrt{|N(i)|}\sqrt{|N(j)|}$: denotes a constant for normalization based on the structure of the graph.
σ: denotes an activation function like ReLU.
$W^{(l)}$: a weight matrix meant for feature transformation.
Thus, a typical GCN layer considers the same age type for all edges across the network. Yet, in the real world, there exist heterogeneous networks with multiple relational edge types. For instance, the Polyphamrcy side effect dataset presented in 5.1.

APPENDIX B.1: Relation graph convolution network (R-GCN)

Motivation: A typical GCN layer defined in Eq. B.1 would pay special attention to the difference in edge relationship types for neighbors of Mustafa when learning Mustafa's embedding vector representations. Yet, it is clear that the edges are actually of opposite (different) types, Esra is unfollowing Mustafa, yet Zeynep is following him. In a knowledge graph, for instance, there exist numerous pairs of triples within the form of topic, relation, and object. Hence, edges encode significant information and have their own unique embeddings that must be learned. Additionally, every given pair may include multiple edges.

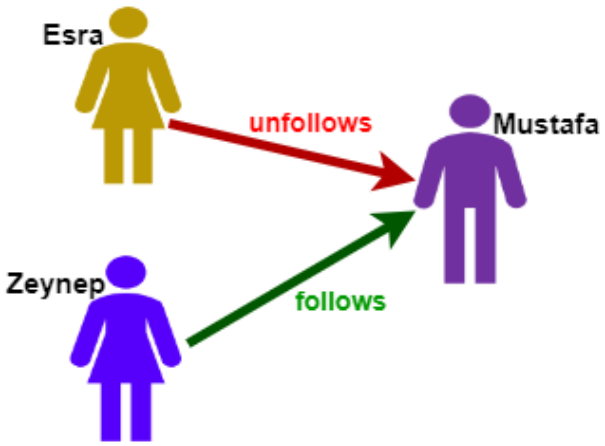

Figure B.1 : R-GCN motivation. Why aggregating Mustafa's neighborhood relations yet they are so different (opposite)?

RGCN: Unlike GCN, RGCN is capable of representing different relation types by leveraging a different projection matrix W for each unique relationship. In the GCN layer B.1, the weight $W^{(l)}$ is shared by all edges. Contrarily, In R-GCN, different edge types take on different weights and only edges belonging to the same relationship type r use the same weight matrix $W_r^{(l)}$. Thus, In R-GCN, the node representation in the $(l+1)^{th}$ layer is computed as in Eq. B.2:

$$h_i^{l+1} = \sigma\left(W_0^{(l)} h_i^{(l)} + \sum_{r \in R} \sum_{j \in N_i^r} \frac{1}{c_{i,r}} W_r^{(l)} h_j^{(l)}\right) \tag{B.2}$$

where
N_i^r: represents the collection of neighbor indices for node i in a relation $r \in R$.
$c_{i,r}$: denotes the normalization constant.
For entity classification, $c_{i,r} = |N_i^r|$

Applying the computation in Eq. B.2 above directly can lead to the rapid growth of several parameters, especially for data with many relationship types. The original work suggests using basis decomposition to decrease model parameter size and avoid overfitting B.3.

$$W_r^{(l)} = \sum_{b=1}^{B} a_{rb}^{(l)} V_b^{(l)} \tag{B.3}$$

Where weight $W_r^{(l)}$ denotes the linear combination of $V_b^{(l)}$, the transformation basis, with coefficients $a_{rb}^{(l)}$. Notably, the total number of bases B is significantly less than the number of relationship types in the knowledge base.

APPENDIX B.2: Graph attention network (GAT)

Intuition: How important node j features are for node i?.
The major difference between the GCN and GAT is the way of aggregating information from the immediate neighborhood. GAT proposes an attention method to replace the statically normalized convolution process in GCN.

The motivation for the attention mechanism is derived from [141] where self-attention by the transformer architecture exhibited overwhelming state-of-the-art performance in machine translation. GAT is essentially an alternative aggregation function, with attention over features of the immediate neighbor rather than a simple average aggregation. Below are the steps and formulas to calculate the node representations $h_i^{(l+1)}$ of layer $l+1$ from the embeddings of the previous layer l in the GAT procedure.

$$z_i^{(l)} = W^{(l)} h_i^{(l)} \tag{B.4}$$

where $z_i^{(l)}$ is a linear transformation on the previous layer $h_i^{(l)}$. $W^{(l)}$ denotes a weight matrix.
An un-normalized pair-wise attention score between two neighboring nodes is computed as follows Eq. B.5:

$$e_{ij}^{(l)} = \text{LeakyReLU}(\vec{a}^{(l)^T}(z_i^{(l)} || z_j^{(l)})) \tag{B.5}$$

where $||$ denotes concatenation operation. The two nodes' z embeddings are first concatenated, then a dot product is taken, followed by a learnable weight vector $\vec{a}^{(l)}$

and finally a leakyReLU is applied as an activation function.
In Eq.B.6, the attention scores on the incoming edges of each node are normalized using softmax.

$$\alpha_{ij}^{(l)} = \frac{\exp(e_{ij}^{(l)})}{\sum_{k\in\{N(i)} \exp(e_{ik}^{(l)})} \tag{B.6}$$

Finally, in Eq. B.7, attention scores are used to scale and aggregate the neighborhood embeddings.

$$h_i^{(l+1)} = \sigma\left(\sum_{j\in N(i)} \alpha_{ij}^{(l)} z_j^{(l)}\right) \tag{B.7}$$

CURRICULUM VITAE

Name Surname : Abubakhari SSERWADDA

EDUCATION:

- **B.Sc.:** 2014, Kyambogo University , School of Science, Department of Computer Science.
- **M.Sc.:** 2017, Istanbul Technical University, Faculty of Computer Engineering, Department of Computer Engineering.

PROFESSIONAL EXPERIENCE AND REWARDS:

- 2018-date: Assistant Lecturer, School of Engineering and Technology, Soroti University
- 2017-2018 Partime Assistant Lecturer, Computer Science Department, Kyambogo University
- 2017-2018 Partime Assistant Lecturer, Computer Science Department, Islamic University In Uganda (IUIU)

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Sserwadda, A.**, Ozcan, A. and Yaslan, Y. (2023). Structural and topological guided GCN for link prediction in temporal networks, *Journal of Ambient Intelligence and Humanized Computing*, 1–9.
- **Sserwadda, A.**, Ozcan, A. and Yaslan, Y. (2023). Topological Similarity and Centrality Driven Hybrid Deep Learning for Temporal Link Prediction, *JUCS - Journal of Universal Computer Science*, 29(5), 470–490.
- **Sserwadda, A.**, Ozcan, A. and Yaslan, Y. (2022). Hierarchical and Centrality driven Polypharmacy Side Effect Prediction Model, *2022 7th Interna- tional Conference on Computer Science and Engineering (UBMK)*, IEEE, pp.384–387

OTHER PUBLICATIONS, PRESENTATIONS AND PATENTS:

- **Sserwadda, A.** and Rekik, I. (2021). Topology-guided cyclic brain connectivity generation using geometric deep learning, *Journal of Neuroscience Methods*, 353, 108988

- **Sserwadda, A.** and Saraç, Ö.S. (2017). Gene selection and classification of pancreatic microarray datasets, *2017 25th Signal Processing and Communications Applications Conference (SIU)*, IEEE, pp.1–4. 49