GRAPH ATTENTION NETWORKS FOR GRAPH LEARNING AND ITS APPLICATIONS

RUNJIE ZHU

A DISSERTATION SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

GRADUATE PROGRAM IN ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
YORK UNIVERSITY
TORONTO, ONTARIO
NOVEMBER 2021

Abstract

This thesis addresses and investigates the recent development of graph attention network (GAT) models in the three following aspects: (1) GATs on single graph learning via the Knowledge Graph Embeddings (KGE) task, (2) GATs on multiple graph learning via the Cross-lingual Entity Alignment (CEA) task, and (3) GATs on on-going real-world problems via the COVID-19 (cell) node classification task. These three aspects of research complement each other in a way that cover a wide range of graph learning tasks to prove the effectiveness and robustness of GAT-based models.

First, GAT has demonstrated its strengths in the KGE task recently beyond its success in many other natural language processing (NLP) tasks. Although GAT has proven to be promising in achieving the state-of-the-art (SOTA) performance in the KGE task, also known as *relation prediction* in KG, the performance of current GAT-based models are still largely restrained due to two reasons: (1) the existing models only take the inbound-directional neighbors of a given entity into account,

while underestimating the rich neighborhood contextual information carried in those outbound-directional neighbors; (2) the SOTA models only apply the k-th hop output to multi-hop embedding learning, which leads to extensive information loss at early-stages (e.g., one-hop). To tackle these two problems stated above, we propose a novel model architecture, bidirectional graph attention network (BiGAT), which fixes these limitations in a unified framework. Specifically, BiGAT leverages the GATs to learn hierarchical neighbor propagation in a bidirectional manner. Extensive experiments conducted on the four publicly available datasets prove the effectiveness and robustness of our proposed model.

Second, past studies of the CEA task of multiple graph learning tend to use traditional approaches, such as translation-based or semantic matching models, to find equivalent entities in the counterpart knowledge graph (KG). These traditional methods share one common limitation of missing important structural information beyond entities in the modeling process. This prevalent issue allows graph neural network (GNN)-based models to step in. Even though these GNN-based methods have achieved some degrees of success in improving model performance, most of them still tend to model KGs of different languages independently for embedding learning. They tend to either underrate the useful pre-aligned links existing between two KGs or utilize only a small amount of pre-aligned entities, served as anchors, to

connect different KGE spaces. These characteristics of prior work largely restrain model performance given the insufficiency of available seed alignments. Moreover, the underestimation of rich contextual information contained within pre-aligned links could further deteriorate the model performance. To address these problems stated above, in this thesis, we propose two novel GAT-based models, Contextual Alignment Enhanced Cross Graph Attention Network (CAECGAT) and the Dual Gated Graph Attention Network with Dynamic Iterative Training (DuGa-DIT), to effectively learn embeddings from different KGs, to capture more neighborhood features, and to propagate more significant cross-KG information through pre-aligned seed alignments. Extensive experiments on the CEA benchmark datasets also reflect the superiority of our proposed models over SOTA methods.

Third, recent studies on the on-going real-world problem of COVID-19 have shown the possibility of leveraging deep learning models to predict COVID-19 disease severity. Thus, another important part of this thesis aims to dive into designing an effective GAT-based model to test out its capability in node classification task. Our proposed method, graph attention capsule network (GACapNet), has delivered significantly better experimental results than baseline models. Moreover, from a biomedical perspective, our study could also indicate predictive features to help close existing knowledge gaps in the pathogenesis of COVID-19 pneumonia.

Acknowledgements

Throughout my PhD journey in the past few years, I have received a great deal of support, guidance and love.

First and foremost, I would like to express my deepest and sincerest gratitude to my supervisor, Professor Jimmy Xiangji Huang. Thank you for taking me under your wing, and introducing me to the world of computer science. You have been a great role model and mentor to me. Your work ethic and positive life outlook have inspired me to become better and stronger. Your encouragement and advice have led me to places I never thought I would go. Your expertise was invaluable in helping me to formulate my research agenda and in navigating the opportunities and challenges in my field of research. Your insightful comments and feedback have helped me sharpen my thinking, writing, and technical skills, and to lift my research onto a higher level.

Second, I would like to thank my thesis committee members, Prof. Aijun An, Prof. Xiangmin Zhang, Prof. George Georgopoulos and Prof. Papagelis, for their

continuous guidance and encouragement throughout my studies. You have provided me with the knowledge, support and tools to choose the right research direction, to overcome difficulties and to complete my dissertation smoothly.

Third, I would like to acknowledge (a) my colleagues from the Information Retrieval and Knowledge Management Research Lab at York University, Canada and (b) my bosses and mentors at AI Singapore for their support. In particular, I would like to single out Jiashu Zhao, Jun Miao, Sadra Abrishamkar, Md Tahmid Rahman Laskar, Lei Liu and many other colleagues in our Research Lab. Thanks for all your collaborations and help over the past few years. I would also like to thank Prof. Ho Teck Hua particularly for giving me the opportunity to work at AI Singapore, and for all the support and care given throughout the journey. I would also like to single out my supervisors at AI Singapore, William Tjhi, Jianshu Weng, Sintia-Teddy Ang and Laurence Liew. Thank you for your continued support and all the opportunities I was given to further my research and to deepen my understanding on AI industry applications.

Finally, I would like to thank my parents for their unconditional love, care, unwavering support and strong belief in me. To my father, thank you for being a steadfast pillar of support, and for your invaluable advice at each major milestone in my life. You have taught me the values of dedication and persistence, and to always take

the long-term view. You have always encouraged me to explore my passions, and have provided me with all I needed throughout my life journey. Because of you, I aspire to lead a life of purpose, through my work and through helping others. To my mother, thank you for being my best friend and showing me how to balance between all aspects of life as a working mom. From you, I learnt the importance of empathy, humility and resilience. I also learnt how to respect different people to gain respect in return. Last but not least, I could not have completed this dissertation without the support of my dearest husband. You have provided sympathetic ears for me all the time, and have engaged in stimulating discussions with me whenever I need. You have also provided me with so many happy distractions to rest my mind outside of my research. Thank you for everything that you have done.

This work was supported by research grants from the Natural Sciences and Engineering Research Council (NSERC) of Canada and York Research Chairs (YRC) program.

Table of Contents

List of Tables

List of Figures

1 Introduction

In recent years, graph attention network (GAT) (170) has shown its effectiveness and has gained more popularity among multi-disciplinary researchers. Compared to its fellow models operating on graph structured data, it has only started to become a mainstream neural network architecture since 2018. Even though it is young, it has already yielded remarkable performance gains in various natural language processing (NLP) tasks, such as language modeling (231), question answering (195), personalized search and recommendation (29, 98, 185, 191). In this thesis, we focus on addressing and investigating the recent advances in GAT models for the following three aspects of graph learning: single graph learning over the Knowledge Graph Embeddings (KGE) task, multiple graph learning over the cross-lingual entity alignment (CEA), and ongoing real-world graph learning over COVID-19 disease prediction. Among these three tasks, they complement each other in a way that cover a wide range of graph learning tasks to prove the effectiveness and robustness of GAT models. Moreover, the KGE also serves as a foundation for the CEA task.

1.1 Background Knowledge

Knowledge graph embeddings (KGEs) aim to embed entities and relations into continuous vector spaces and to generate low-dimensional representations in a knowledge graph (KG). The purpose of embedding is to inherit the original structure of KG entities, and to simplify the process of manipulating them in future. Thus, effective KGEs usually provide a contextualized overview about the given KG that can be utilized to infer missing entities or relations. As the number and the scale of KGs grow rapidly, KGE becomes more important for KG analyses and semantic data modeling. Similar to other tasks in NLP, GAT-based models have demonstrated their strengths in the KGE task itself, as well as in support of the CEA task.

Yago (131, 157), DBpedia (78), and ConceptNet (154) are cross-lingual knowledge bases developed in the past few years which contain cross-lingual KGs and represent different concepts and entities with various relations. As these KGs are usually constructed by information extractions from each monolingual text corpora, there has been a rising need to align and synchronize those information across various languages to complement each other and to compose more comprehensive knowledge bases. Thus, along with the rapid development of these cross-lingual KGs, the NLP task of CEA has also become increasingly important. Specifically, CEA refers to a task of matching equivalent entities with same semantic meanings from KGs of

different languages. However, it still remains as a nontrivial and challenging mission, as those distinct surface forms and heterogeneous structures within KGs can easily pose obstacles to conventional methods.

COVID-19 is an unprecedented global health crisis that not only threatens the current healthcare industry but also triggers subsequent economic and financial crisis. As of November, 2021, there have been more than 261 million confirmed coronavirus cases worldwide, and the total death toll has exceeded 5.21 million people. Public and private medical institutions and other healthcare providers from east to west have been working diligently to study and cure the disease from different medical philosophies and various interdisciplinary approaches. Recent studies have demonstrated the effectiveness of graph neural networks (GNNs)-based models in COVID-19 disease predictions. Thus, another significant component of the proposed dissertation aims to design an effective GAT-based model to test out its capability in the node classification task with single-cell RNA sequencing (scRNA-seq) datasets.

1.2 Problems and Motivation

As aforementioned, even though some existing methods have presented viable solutions to tackle the tasks listed above, there still remain some potential challenges. For the KGE task, it is still challenging for those proposed methods to learn hier-

archical multi-hop embeddings effectively, while avoiding excessive information loss at early stages. For the CEA task, it is still challenging for those existing methods to link up correspondent entities from different KGs with limited pre-aligned seed alignments. And for the COVID-19 disease prediction task, there has been limited number of prior work published due to both the novelty of this research domain and the lack of publicly available high-quality scRNA-seq data.

Given that, the motivation of this thesis could break down into mainly two folds. First, the expressiveness of conventional graph learning methods, such as translation-based methods or semantic matching methods, is largely restrained due to their simple structures, low scalability, and independent pre-assumption of entities and relations embeddings. It would be impactful to propose methods that could tackle extensive and complex KGs structures, model effective hierarchical embeddings by considering interactions between entities and relations, and preserve significant information as much as possible through propagation. Second, even though GAT-based models have achieved great success in many NLP downstream tasks in the past few years, it is still relatively young in the graph learning domain with most of the work published in 2018 and beyond. It would be interesting to see the integration of these two concepts by applying the GATs to graph learning research topics such as KGE relation predictions, CEA and other real-world scenarios.

1.3 Contributions of the Thesis

This thesis aims to contribute new perspectives towards all the problems stated above. In particular, we focus on proposing new GAT-based models for research domains of KGE, CEA, and COVID-19 disease severity predictions. All the experiments carried out in this thesis have demonstrated the effectiveness and robustness of proposed methods over SOTA solutions.

In the single graph learning of KGE domain, we propose a novel bidirectional graph attention network (BiGAT) to learn embeddings in a hierarchical neighboring propagation manner. Specifically, we propose novel techniques, which not only consider resourceful neighboring contextual information from both inbound and outbound directions, but also avoid extensive early-stage information loss by collecting and propagating hierarchical hop embeddings from every single layer. In particular, it takes advantage of the GAT structure for multi-hop neighbor propagation. On one hand, the proposed method captures multi-hop neighborhood information from both inbound and outbound direction for each entity, which empowers the model to leverage more global contextual information. On the other hand, the attention mechanism in the architecture integrates hierarchical hop embeddings information from each BiGAT layer to a unified representation. This design allows our model to perform more robustly compared to other baseline models, and to prevent em-

bedding information loss from early stages. Extensive experiments conducted on the four publicly available benchmark datasets, WN18RR (27), FB15k-237 (167), NELL-995 (207) and Kinship (83), justify the effectiveness and superiority of our proposed method.

In the multiple graph learning of CEA domain, in order to address those relevant problems stated in Section 1.2, we propose two enhanced GAT models by integrating more neighborhood contextual information, propagating more cross-KG information, and enriching new seed alignments iteratively. In order to take full advantage of the contextual alignments for our CEA task, we propose a novel GAT-based Contextual Alignment Enhanced Cross Graph Attention Network (CAECGAT) model as preliminary study. CAECGAT learns the embeddings across different KGs jointly by propagating cross-KG information through pre-aligned seed alignments. Particularly, we propose a new training strategy by dividing seed alignments into two categories, namely the contextual and objective seed alignments, with an iterative manner. This strategy allows our proposed method to capture more cross-KG information effectively. Furthermore, we innovatively develop a new cross graph attention (CGAT) layer, which consists of a cross-KG aggregation and an attention-based cross-KG propagation layer, to learn cross-KG information. The cross-KG aggregation layer serves to transfer entity information across two KGs with pre-aligned seed entities,

and to map these learnt embeddings from different KGs to same semantic space by shared cross-KG information. The attention-based cross-KG propagation layer serves to emphasize and propagate those neighbors with essential cross-KG information to the next layer. Thus, semantic gaps among cross-lingual KGs can be largely alleviated by propagating these cross-KG information. Extensive experiments conducted on three benchmark datasets demonstrate that our proposed CAECGAT achieves remarkable performance gains over SOTA models.

Even though CAECGAT is more effective than SOTA approaches, we find that this preliminary study still faces certain challenges in information propagation stage when the given number of pre-aligned entity pairs is limited. In turn, we further enhance the GAT models and mitigate these problems in a unified framework by introducing a Dual Gated Graph Attention Network with Dynamic Iterative Training (DuGa-DIT). The DuGa-DIT leverages both neighboring features and cross-KG alignment information to learn cross-KG embeddings and to bridge the semantic gaps between cross-lingual KGs. The model architecture comprises several dual KG attention layers where each layer consists of an intra-KG attention layer and a cross-KG attention layer. The intra-KG attention layer serves to learn neighboring features for each KG, and the cross-KG attention layer serves to collect alignment information across multilingual KGs. By stacking multiple dual KG attention layers, the model

is able to propagate local neighboring information and cross-KG information over multi-hop neighbors. Unlike other baseline models, DuGa-DIT can make full use of the pre-aligned entity pairs in both training and prediction stages. Furthermore, we develop a dynamic iterative training process to dynamically update the cross-KG attention score matrices by iteratively adding new seed alignments to the input. This allows more cross-KG information to transfer across different KGs. Similar to the preliminary study, extensive experimental results on the benchmark datasets prove our model's effectiveness, robustness and superiority for CEA task over SOTA methods.

In the real-world graph learning problem of COVID-19 disease prediction domain, we attempt to build a GAT-based hybrid model to facilitate research process for interdisciplinary researchers and scholars, to promote collaborations, and to combat this COVID-19 battle together. Particularly, we focus on identifying patterns of SARS-CoV-2 transcriptome and types of molecular cells that indicate different degrees of disease severity using single cell transcriptomic data, scRNA-seq. According to different gene features that the molecular cells carry, all cells in the scRNA-seq data are classified into different severity levels for cell node classification. We propose a combination of GAT-based models and capsule network techniques to classify the cell nodes. These proposed techniques are integrated into a unified framework, called

a graph attention capsule network (GACapNet). GACapNet stacks multi-head attention networks on top of a capsule network, which consists of a primary capsule layer and a dynamic routing procedure. The multi-head graph attention networks contribute to aggregate contextual features of the center node, while preserving its original node features to the maximum. The capsule layer with dynamic routing also contributes to effectively combine and fuse gene features, allowing more prominent features of infected cells and healthy controlled cells to be presented at the output stage. The experimental results on two scRNA-seq datasets show that our proposed GACapNet is capable of classifying cells according to its severity level effectively. Moreover, it also largely outperforms the SOTA baseline models. To the best of our knowledge, we are the first study that extends GATs and capsule networks to scRNA-seq datasets for COVID-19 disease severity predictions.

In general, the combination of the above three modules composes the main body of this thesis. The work on single graph learning, *Hierarchical Neighbor Propagation with Bidirectional Graph Attention Network for Relation Prediction*, was published at IEEE ACM Trans. Audio Speech Lang. Process. 29: 1762-1773 (2021). The two studies on multiple graph learning, *A Contextual Alignment Enhanced Cross Graph Attention Network for Cross-lingual Entity Alignment* and *Dual Gated Graph Attention Networks with Dynamic Iterative Training for Cross-Lingual Entity Alignment*,

were published at COLING 2020 and ACM Transactions on Information Systems. TOIS. 2021. 40 (3), 1-30, 2021 respectively. Lastly, the work on real-world COVID-19 disease prediction issue, *How severe will your COVID-19 infection be? Predicting SARS-CoV-2 infection with graph attention capsule networks*, was submitted to an ACM journal, still under review. These studies included in this thesis aim to inspire researchers and scholars in relevant fields to come up with more innovative ideas that can better tackle the persisting challenges, integrate them into unified frameworks, and make solid progress in KGE, CEA and COVID-19 research domains.

1.4 Outline

This thesis comprises six chapters. The detailed chapters are organized as follows. Chapter 1 presents a thorough introduction of the research background, and highlights the research problems, motivations as well as the contributions of this thesis. Following that, Chapter 2 introduces the key concepts and provides a thorough literature review of relevant research areas discussed in this thesis. In the next three chapters, one GAT-based advanced KGE model is proposed in Chapter 3, two GAT-based advanced CEA models are proposed in Chapter 4, and one GAT-based capsule network model is proposed for COVID-19 disease severity prediction in Chapter 5. The chapter 3, 4 and 5 are organized in a way that complement each other to cover

a wide range of graph learning tasks to prove the effectiveness and robustness of GAT models. All the modeling, training and experimental details are elaborated in each chapter respectively. Lastly, Chapter 6 concludes the thesis by discussing the implications of this thesis and suggesting several potential directions of future work.

2 Literature Review

In this chapter, we will provide a thorough introduction to the research background of this thesis and explore the prior work on graph learning. We first provide a reference table of the most commonly used notations in Section 2.1. Following that, we introduce the basic concepts of graphs in Section 2.2. Next, we define various machine learning tasks on graphs in Section 2.3 and discuss various traditional (non-deep learning) approaches on graph learning in Section 2.4. After that, we summarize the basics of neural networks in Section 2.5 to serve as a foundation of our deep learning approaches which would be introduced and reviewed in the following chapters.

2.1 Notations

We begin this section by providing a reference notations table that will be extensively applied throughout the following chapters of this thesis. Details are presented in Table 2.1. In addition, other notations will be introduced in each following individual

Table 2.1: List of Notations.

Name	Notation	Description
Knowledge Graph	$\mathcal{G}$	A set of triples in the format $(e_i, r_{i,j}, e_j)$
Entities	$\mathcal{E}$	A set of entities in KG
Relations	$\mathcal{R}$	A set of relations in KG
Input Entity Embedding	$\mathbf{E}$	The input embedding matrix of entities
Input Relation Embedding	$\mathbf{R}$	The input embedding matrix of relations
Output Entity Embedding	$\mathbf{E}'$	The embedding matrix of entities output by our model
Output Relation Embedding	$\mathbf{R}'$	The embedding matrix of relations output by our model
Embedding Size	d	The dimension of entity and relation embeddings
Entity Number	$\lvert\mathcal{N}\rvert$	The number of entities in KG
Relation Number	$\lvert\mathcal{R}\rvert$	The number of relations in KG
Attention Head Number	P	The number of attention head in our model
Hop Number	k	The number of BiGAT layers of our model
Loss Function	$\mathcal{L}$	The loss function of our model

chapter if necessary.

2.2 Basics of Graphs

2.2.1 What is a graph?

In this section, we begin by introducing the basic concepts of graphs related to this thesis. A **Graph ($\mathcal{G}$)** is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ represents a set of vertices

and $\mathcal{E}$ denotes a set of edges, respectively. The two end points, $u \in \mathcal{V}$ and $v \in \mathcal{V}$, of an edge e travel from one to another and are joined by e. Thus, u is called a neighbor of v, and these two vertices are in **adjacent** relation. The edges $\mathcal{E}$ can be either **directed** or **undirected**. A graph $\mathcal{G}$ can also be a **directed graph** containing all directed edges, or an **undirected graph** containing all undirected edges. The **degree** of a vertice refers to the number of edges that are connected with this vertice.

In order to represent a simple graph mathematically, every node in the graph is indexed to a specific row and column of an adjacency matrix. An **adjacency matrix** $A \in R^{|n|\times|n|}$ is a convenient algebra representation of a simple graph with n number of vertices, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where

$$A_{ij} = \begin{cases} 1, & if \mathbf{v_i}, \mathbf{v_j} \in E and i \neq j \\ 0, & otherwise \end{cases} \tag{2.1}$$

If only undirected edges construct the graph stated above, the matrix A is symmetric; otherwise, if directed edges construct the graph, it is not necessarily symmetric. In certain conditions, graphs may contain weighted edges, where the indexed entries of the matrix A are arbitrary real-values rather than simply 0 and 1.

2.2.2 Different Types of Graphs

Beyond the simple graph, we also consider multi-relational graphs with different edge types or relation types r, which is denoted as $(u, r, v) \in \mathcal{E}$. Thus, an adjacency tensor, $A \in R^{|V|\times|R|\times|V|}$, summarizes the graph by a collection of relations $\mathcal{R}$.

Under the multi-relational graphs, there are mainly two subsets, namely the heterogeneous graphs and the multiplex graphs. The heterogeneous graphs are imbued with types. Specifically, they consist of nodes and edges representing different types. For instance, the graph in biomedical domain will have nodes of patients, symptoms, protein, drugs, diseases and doctors, which are linked via different edge types, such as has-effect-on, can-cause-to, has-side-effect-to, has-prescribed-to and has-taken etc.. It is worth noting that, on contrary to heterogeneous graphs, there is a category of homogeneous graph where nodes and edges represent instances and relations of same type. For example, social networks contain all the entities of people, and relations in between these entities are friends. Multipartite graph is considered to be a special subset of heterogeneous graphs. It consists of edges only linking nodes with different types.

The multiplex graphs refer to graphs composed of n number of different layers. Each node exists in each layer which corresponds to a certain relation of the intra-layer edge type. Besides, a same node across different layers can also be connected

by inter-layer edge types. An example to illustrate the multiplex graphs could be the entire transportation network in a country, e.g. Canada. All the cities in Canada (e.g. Toronto, Vancouver, Montreal, Quebec) can be viewed as entities of multiplex graphs, while each layer of the graph could be viewed as a different mode of transportation (e.g. flight, boat, train, car, subway). Thus, cities that are connected by the different modes of transportation are represented as intra-layer edges; and the various options of transportation within the specific city are represented as inter-layer edges.

Finally, multigraphs are another subset of graphs which contain self-loops and multiple edges connecting the same pair of entities. However, graphs that do not contain self-loops and multiple edges are simple graphs as discussed above. In many cases, a graph $\mathcal{G}$ in this thesis refers to **directed graph**, unless specified.

It is worth noting that features and attributes information are commonly associated with graphs in many scenarios, e.g. knowledge graphs. Sometimes, edges of graphs contain additional information like edge types, ranking, properties or weights, and entities of graphs contain attribute features or entity types. These features and attributes usually contain rich context information for graph representation learning.

2.2.3 Knowledge Graph (KG)

A **knowledge graph (KG)**, typically denoted as $\triangle$ and represented as $G = (E, R, T)$, is a data structure comprised of entities E, relations R and triples T, where E stands for a collection of entities, R stands for a collection of relations, and T stands for a collection of triples. As Google announced its Google KG (149) in 2012, the research communities in both industry and academia have been busy pushing this concept towards further advancement. Facebook(115), LinkedIn (49), DBpedia (6), Freebase (13), Wikidata (173), OpenCitations (121), ScriGraph (57), Microsoft Academic KG (34), NELL (101), YAGO (130), BabelNet (105), xlore (187), etc. are also rapidly developed along with the wave of this research heat. In this thesis, we will mainly focus on graph learning on knowledge graphs.

2.3 Machine Learning Tasks on Graphs

In machine learning, one of the first questions we consider solving a particular task is whether it is a supervised or unsupervised task. Machine learning tasks on graphs are similar, but the usual classifications of supervised and unsupervised may not be the most useful or informative when the data is in graph structure. In the past few years, the research on graph learning methods have attracted lots of attentions, including tasks of:

- **Node classification:** Node classification aims to predict type or color of a given node;
- **Relation prediction:** Relation prediction aims to predict if two nodes are linked to each other;
- **Community detection:** Community detection aims to identify densely connected clusters of nodes;
- **Classification, regression or clustering on entire graph:** This task aims to make predictions on each individual graph given multiple different graphs for classification and regression models, and to measure similarities between graphs for clustering models.

2.3.1 Node Classification

Among the four tasks aforementioned, node classification is perhaps the most popular task on graph learning in the past few years (47, 65). Although node classification appears to be a straightforward supervised classification task, it has a fundamental difference from conventional supervised machine learning where data points are assumed to be independently and evenly distributed. Graph nodes obviously break this assumption by requiring us to model a set of interconnected nodes with dependency

considerations. Indeed, the most successful models proposed for node classification use connections between nodes explicitly.

2.3.2 Relation Prediction

Relation prediction, also known as link prediction, is another popular task on graph learning recently. The rationale is to infer missing edges between the nodes given a set of nodes and their incomplete edges. Similar to node classification, relation prediction often refers to both supervised and unsupervised training. Moreover, it supports various downstream applications, such as recommender systems (221), biomedical drug reaction prediction (240), cross database inferences (15, 27, 216), and cross lingual knowledge graphs (48, 117, 203).

In fact, many promising models have been proposed to tackle the entity alignment problem and to predict links on KGs, which are the major focuses of this thesis. One direct solution to entity alignment is to learn the KGE effectively and to leverage the embedded representations to predict missing links. Technically, the existing KGE approaches fall into five main categories: translation-based models, semantic matching models, convolutional neural network (CNN)-based models, neighborhood information-based models, and relation paths/literals-based models. The KGE task started with translation-based methods and semantic matching methods.

More recently, there has been a surge of leveraging CNN and GNN models to embed entities and relations of KGs into continuous vector spaces. Generally, the neighborhood information-based models deliver more promising experimental results by using graph neural network (GNN)-based models (104, 138, 140). However, most of these model performances are still largely restrained to single KG. Details of these prior work will be elaborated in relevant chapters.

2.3.3 Community Detection

Community detection, which aims to identify densely connected clusters of nodes, is an example of unsupervised clustering. We assume the nodes would follow a community structure where nodes belong to the same community would be more likely to be linked to each other.

2.3.4 Classification, Regression or Clustering on Entire Graph

Classification and regression tasks, which aim to make predictions on each individual graph given multiple different graphs, are typical examples of supervised learning. Given the assumption of each data point being individually and evenly distributed over the graph, a classification or regression model aims to map these data points or graphs to labels with labelled training set. Besides, clustering, which compares

similarities between two nodes or graphs, is an example of unsupervised learning. The definition of useful features and attributes to be included in relational structure remains as a difficult and challenging issue in this research domain.

2.4 Traditional Approaches to Graph Learning

In this section, we dive into the traditional machine learning approaches and hand-crafted features for graph data. The goal of traditional graph machine learning pipelines is to leverage d-dimensional vectors to make predictions for a set of objects, such as nodes, sets of nodes, links and entire graphs.

2.4.1 Node-level Task

For node-level tasks, the goal is to characterize the structures and positions of a target node in the massive network. There are several node-level features that we could utilize for this task, which includes node degree, node centrality, clustering coefficient and graphlets. Based on their functions, we could further categorize them into two groups, namely the importance-based features, and the structure-based features.

The importance-based features refer to those features that are able to predict influence of a node in a network. Node degree and node centrality are two features belong to this category. Node degree refers to the number of edges that a specific

node has. The essence of the node degree is to exercise simple counting on edges and to treat all neighboring nodes equally. Node centrality models the node importance among the neighboring nodes. One of the common approaches to calculate node centrality is Eigenvector centrality, denoted as,

$$\mathbf{c}_v = \frac{1}{\lambda} \sum_{\mathbf{u} \in \mathcal{N}_\mathbf{v}} \mathbf{c}_v \tag{2.2}$$

where c is the centrality vector, and u is a node belong to the neighborhood of node v. It assumes that a node is significant if it is surrounded by other significant nodes. Another common node centrality measurement is betweenness centrality, which assumes the significance of a node if it lies on many shortest paths between other nodes. The closeness centrality is also commonly seen to measure node centrality, which views a node to be important if it has small shortest path lengths to all other nodes.

The structure-based features refer to those features that are able to capture topological properties of a node's local neighborhood and to examine a particular role a node plays in a network. Node degree, clustering coefficient and graphlet degree vectors are three features belong to this category. Clustering coefficient measures how connected a node's neighboring nodes are. The common approach is to count the number of triangles that a node touches within a given network. Graphlet degree vectors, which refers to the graphlet-based features for nodes, stand for a count of

occurrences of different graphlets rooted at a given node.

2.4.2 Link-level Task

For link prediction tasks, the goal is to make predictions on missing links or new links based on existing links. Therefore, it is important to design features for a pair of nodes. There are several link-level features that we could utilize for this task, which include the distance-based features, the local neighborhood overlap and the global neighborhood overlap. The distance-based features leverage the shortest path lengths between two nodes but does not learn the degree of neighborhood overlaps. The local neighborhood overlap captures how many neighboring nodes are shared by two nodes. The drawback of this feature is that it could become zero when no neighboring nodes are shared at this moment, however, they could still potentially be connected in future. The global neighborhood overlap tackles this limitation by considering the entire graph. It leverages the Katz Index, which counts the number of paths of all lengths between a pair of nodes, to score two nodes.

2.4.3 Graph-level Task

For graph-level prediction tasks, the goal is to capture features that could characterize the entire graph structure and to measure similarity between two graphs using

graph kernels. The idea of graph kernels is similar to the bag-of-words function for graphs. Traditionally, there are two dominant approaches to learn graph-level features, namely the Graphlet Kernel (144) and the Weisfeiler Lehman Kernel (143). Given two graphs G and G', the Graphlet Kernel is a counting of graphlets computed as:

$$K(\mathbf{G}, \mathbf{G}') = \mathbf{f_G}^T \mathbf{f_{G'}} \tag{2.3}$$

However, when the two graphs have different sizes, the resulting value could be skewed. Thus, features vectors generated in this way need to be normalized. And this approach is also considered to be computationally expensive. To address the problem of Graphlet Kernel, Weisfeiler-Lehman Kernel (WL-Kernel) was introduced to design an efficient graph feature descriptor with bag-of-color concept. The key idea of this WL-Kernel method is to leverage neighborhood structures to enrich nodes vocabularies iteratively using color refinement. Specifically, each node in the given graph is assigned with an initial color. Then, each node goes through the neighbors to aggregate the neighborhood colors. After that, the resulting aggregated colors are hashed into respective hash table iteratively. When the node colors are all refined across the graph, WK-Kernel joins to count the number of nodes with a specific color. As a result, the WL-Kernel value is computed by the dot product of those color count vectors. Comparably speaking, WL-Kernel is more computationally efficient

than Graphlet Kernel as the complexity increases linearly when the number of edges increases.

2.5 Basics of Neural Networks

In the past decade, research in neural network structures has experienced a huge success. Currently, the existing neural network structures can be classified into mainly four categories:

2.5.1 Feedforward Neural Networks

The feedforward neural network (FNN), or multilayer perceptrons(MLPs), is the first and quintessential deep learning architecture of artificial neural network. Typically, a FNN consists of one input layer, one output layer and several hidden layers in between. As its name stated, information within the FNN only travels from input layer, through hidden layer, and exits output layer, in a forward direction. No cycles or loops are allowed in this network.

2.5.2 Convolutional Neural Networks

The convolutional neural networks (CNNs), introduced by Yann LeCun, is a special class of artificial neural networks that mimics the human's vision system. The

system usually consists of convolutional operation layers, pooling layers, flattening layers and fully connected layers. The early version of today's CNN is called LeNet (75) which could recognize handwritten digits. In 2012, AlexNet (69), trained on millions of labelled pictures from ImageNet, showcases the viability of tackling this computer vision task with deep learning which has never been possible before. Additionally, various CNN based approaches (148, 164) have been proposed in the past few years and the CNN architecture has proven to be robust in various downstream applications, such as image recognition, image classification, video recognition, recommender system and other natural language processing related tasks.

2.5.3 Recurrent Neural Networks

Rucurrent neural networks (RNN) is a class of artificial neural networks where connections between neurons follow a temporal dynamic behaviour and effectively process sequences of inputs with memories of historical information. In other words, the output from the previous state carrying memories of all prior states is used as inputs to the next state, while incorporating hidden states. Similar to CNN, the RNN architecture is also widely adopted to many downstream tasks, with emphasis on sequence-based tasks like natural language processing and speech recognition. Note that the RNN architecture usually suffers from long term dependency problem.

Therefore, several variants have been proposed since then to tackle this drawback, including LSTM (51) and GRU (25).

2.5.4 Graph Neural Networks

Graph neural networks (GNNs) is another class of artificial neural networks which is designed for performing inferences on data in graph structure. Social networks, economic networks, communication networks, information networks, knowledge graphs, molecular structures are all examples of networks in graph structure. The key idea of GNN is to learn effective representation of entities that inherit the structure of the graph and preserves the most expressive features.

3 Learning on Single Graph: GATs for Knowledge Graph Embeddings and Relation Predictions

In this chapter, we dive into the first module of this thesis on GATs for single graph learning. We introduce a new bidirectional graph attention network (BiGAT) to learn hierarchical neighbor propagation for the KGE task.

3.1 Introduction

Since early 2010s, many large-scale KGs have been constructed, such as Freebase (13) and DBpedia (78). Typically, a KG refers to a directed graph where entities are represented as *nodes* and relations between entities are represented as *edges*. Specifically, for a collection of factual triples (e_i, r, e_j) of the massive KG, r stands for the relation between the head entity e_i and the tail entity e_j. For instance, among the

factual triple (*Toronto*, *cityOf*, *Canada*), the entities of *Toronto* and *Canada* are the head and tail entity respectively, and the *cityOf* stands for the relation between these two entities. Given those resourceful information existed within its network, KG has been widely adopted in many downstream NLP applications, including semantic search (206), question answering (23, 73), dialogue system (94), as well as machine reading (215).

Despite the applicability of KGs, there still persists several critical challenges in learning on KGs. One of the key problems is the high degree of incompleteness of the existing KGs. Specifically, missing links and relations among KGs impose obstacles to the models and algorithms that aim to learn effective KGE. For instance, the relation of "place of birth" is missing among 71% of the people entities in Freebase and 66% of the people entities in DBpedia; the relation of known parents (150, 192) is missing among 94% of the people entities in Freebase. In order to tackle the challenge stated above, many scholars and researchers have emphasized on the task of *knowledge graph embedding* (KGE), also known as KG *relation prediction*, recently. This task aims to infer missing links and relations within a given KG first, and it also attempts to make predictions on all possible candidate entities (e.g., $(e_i, r, ?)$ or $(?, r, e_j)$).

In the past, traditional translation-based models, such as TransE (15) and its

various extensions of TransR (88), TransD (59), TransH (190) etc., and matrix representations-based models, such as DistMul (216) and ComplEx (167), have been introduced to learn entities and relations embeddings. Even though models taking these conventional approaches achieved some promising results, their attempts to learn embeddings from individual and independent triple weakens the expressiveness of those embedding vectors (116). Later on, convolutional neural networks (CNN)-based models (27, 106) were introduced to enhance embedding expressiveness. However, one of the drawbacks of these CNN-based embedding models is that they still tend to treat entity and relation embeddings as independent vectors at the computation stage. This characteristic largely ignores the rich contextual dependent information existed among the entities and relations.

Lately, many researchers and scholars in this domain has been learning embeddings by neighborhood information aggregations (9, 107, 116, 138, 140, 176). Comparably speaking, these models have achieved remarkable performance improvements for the KGE task, but they still appear to be weak in capturing rich neighborhood information. For example, (9, 107, 138, 140) only take immediate neighbors into account, and (104, 116) only take the top-k hops neighbors away into account. Specifically, KBGAT (104) is the first model that integrated the GAT (170) structure into the KGE task. Even though this approach allows it to achieve SOTA

record performance, the expressiveness of those derived embeddings is still largely restricted due to two reasons: (1) The KBGAT model only considers neighbors from the inbound-direction of the given entity while underestimating the neighbors towards the outbound-direction. This one directional approach limits the embedding expressiveness, and restricts its capability of capturing sufficient neighborhood information; (2) The KBGAT model solely utilizes the k-th hop output in distant nodes embedding learning. This characteristic causes the extensive information loss at early computational stage, such as the one-hop information.

In order to tackle all the modeling problems stated above and to take advantage of the GAT for multi-hop neighbor propagation, in this chapter, we propose a novel bidirectional graph attention network (BiGAT) to learn embeddings in a hierarchical neighboring propagation manner. Explicitly, our BiGAT first concatenates two features produced by both the inbound-directional and the outbound-directional GATs. Then, our model applies a non-linear transformation to get new entities embeddings, and a linear transformation to get new relations embeddings. After that, we exercise the bidirectional neighborhood information propagation for each hop using the multiple BiGAT layers. Thus, these multi-hop features in KGs are encoded layer by layer for aggregation. Lastly, our model applies an attention mechanism to gather the hierarchical hop embedding information and to integrate them from each BiGAT

layer.

In general, the main contributions of this chapter can be ascribed as follows:

- We propose a novel BiGAT model for single graph learning on the KGE task. The BiGAT is capable of capturing multi-hop neighborhood information from both inbound and outbound directions for each entity, which empowers it to leverage more global contextual information in the modeling process.

- An attention mechanism is proposed to integrate the hierarchical hop embedding information generated from each BiGAT layer into an unified representation. This mechanism not only ensures the model robustness but also prevents embedding information loss problems at early stages.

- In order to prove the effectiveness of our proposed model, we conduct extensive experiments on four publicly available benchmark datasets, including WN18RR (27), FB15k-237 (167), NELL-995 (207) and Kinship (83). The experimental results demonstrate the competitiveness of the BiGAT model on various evaluation metrics, as well as its superiority over other SOTA models.

The remainder of this chapter is organized as follows. In Section 3.2, we present the SOTA progress of the KGE task. In Section 3.3, we describe the architecture of our proposed bidirectional graph attention network (BiGAT) model and elaborate

each module in details. In Section 3.4, we report our experimental results along with model comparisons. In Section 3.5, we conclude this chapter with some ideas for future research.

3.2 Related Work

In the past few years, a great number of novel models have been introduced to the KGE task, including several survey papers (39, 61, 84, 181). In this section, we dive into the recent development of KGE and offer an overview of these recent advances in entities and relations representation learning.

In general, the existing literature on KGE task can be divided into five categories, namely the translation-based models, semantic matching models, CNN-based models, neighborhood-based models and transformer-based models. In the subsequent sections, we offer a comprehensive review of each category of methods and discuss their potential benefits and limitations.

3.2.1 Translation-based Models

Translation-based models aim to learn KGE based on translation characteristics. In other words, it considers the relations between head and tail entities as a form of translation. The first model introduced belong to this category is TransE (15).

TransE assumes that the sum of the head entity embedding and the relation vector should be close to the tail entity embedding. Even though it is able to model the hierarchical structure of the input graph, the performance is largely restricted due to the TransE's limitation on one-to-one relation. After TransE was introduced, many extentions and variations of it have been proposed. For instance, in order to address the one-to-many, many-to-one and many-to-many issues, TransH (190) was introduced to project both entities and relations into a hyperplane while preserving its translation operations. TransR (88) attempts to learn both entities and relations embeddings separately first, and project them to the relation space to build translations accordingly. Taking a similar approach, TransD (59) borrows the idea from TransR and extends to project head and tail entities as well as the relation vectors into new space by leveraging dynamic mapping matrices. TransF (32) releases the restrictions on head and tail entities while constructing a flexible translation scoring function to tackle the same issue. TransAt (125) chooses to leverage relation attention mechanism to model the translation embeddings. TransMS (218) transmits the multi-directional semantics with nonlinear functions and linear bias vectors to extend the translation-based methods family. TransG (200) and KG2E (50) introduce Gaussian distribution to their architecture aiming to deal with the uncertainties of entities and relations.

3.2.2 Semantic Matching Models

Semantic matching models refer to a group of methods that use vectors and matrices to calculate semantic similarities. As we commonly agree, SME (14) is the first model that uses linear and bilinear matching blocks to encode entities and relations interactions. DistMult (216) utilizes a simplified bilinear transformation to enforce the relation matrices to be diagonal. RESCAL (111) is the first KGE model which takes an approach of representing triples with three-way tensor factorization. Built on top of that, TuckER (8) extends the model by using three-way Tucker tensor decomposition, which comprises of a core tensor and entities and relations vectors, to learn embeddings. HolE (110), HolEx (213) and ANALOGY (89) are then introduced as methods adopting circular correlation of embeddings and their extensions. Models belong to this category use real-valued space to model representations of entities and relations. Aside from all above, there are also semantic matching models that represent entities and relations in complex space. For instance, ComplEx (167) aims to leverage the Hermitian dot product and to map entities and relations in a complex space for capturing both symmetric and antisymmetric relations. QuatE (226) uses the quaternion inner product with three imaginary components to model factual triples accordingly. RotatE (159) uses element-wise Hadmard product to compute the relations as a rotation from head to tail entities. Other relevant studies belong

to this category include but are not limited to TorusE (30) and D4-Gumbel (209). Even though semantic matching models are fast to train and easy to scale to large KGs, they still learn embeddings with relatively low quality and expressiveness.

3.2.3 CNN-based Models

CNN-based models tap on their advantage of parameter efficiency to employ multi-layer CNNs and to generate embeddings more expressively. ConvE (27) models the local interactions between head entities and relations using a 2D convolution operations. The ConvE is capable of using multiple non-linear feature learning method to express semantic information effectively by reshaping the head and relation embeddings into 2D matrices. As an extension of ConvE, ConvKB (106), on one hand, preserves the transitional characteristic of ConvE by employing CNN to model the global relationships among the same dimensional entries of the embeddings. On the other hand, it replaces the reshaping characteristic of ConvE by concatenating the entities and relations directly. InteractE (168) was proposed to derive higher quality embeddings by increasing number of interactions between entities and relations embeddings. In general, CNN-based methods are more powerful to learn expressive features compared to the first two categories, but they suffer from losing plenty of latent information in KGs when consider each triple independently.

3.2.4 Transformer-based Models

Transformer-based models are considered to be a group of effective and powerful KGE models given the large pool of KG contextual information that they could absorb from in the representation learning process. KG-BERT (220) and MLMLM (26) utilize the pre-trained language model (e.g., BERT (28)) to encode entities and relations existed in KGs. Similarly, CoKE(179) also leverages transformers as encoders for learning sequences of edges and paths.

3.2.5 Neighborhood-based Models

Neighborhood-based models target to learn embeddings by engaging entity neighborhood information. A collection of past studies (9, 107, 138, 140, 176) have demonstrated significant performance gains over the KGE task. However, a majority of neighborhood-based approach still lack of the capability of modeling the multi-hop neighborhood information effectively. Recently, CACL(116) was proposed. Specifically, it leverages connectivity patterns contained in the multi-hop neighbors for the KGE task to build the context-aware CNNs by considering top-k neighborhood entities solely. Besides, KBGAT (104) was also proposed to capture the multi-hop features. The model leverages the attention mechanism from GAT (170) and takes those inbound-directional neighboring information of a given entity into considera-

tion. GAAT (182) aggregates neighborhood features by using an attenuated attention mechanism.

3.2.6 Difference to the Existing Methods

Different from all the methods reviewed above, our proposed BiGAT takes rich neighboring information from both inbound and outbound directions, and propagates bidirectional neighborhood information to learn embeddings in an iterative manner.

In comparison with the prior work (152), which also considers bidirectional neighborhood information, our models differ in mainly two aspects: (1) **Solving the problem of losing early-stage embedding information:** In their study (152), the authors merely takes the k-th hop output into account, which largely sacrifices the embedding information generated at early-stage. Contrarily, our proposed BiGAT solves the early-stage information loss problem by aggregating outputs generated from each single hop. Thus, our model captures multi-hop hierarchical feature embeddings more effectively and efficiently. (2) **The capability of capturing global contextual information through different weighting mechanisms:** In their study (152), equal importance weights are assigned to all neighboring nodes in the learning stage, where the contextual information is largely undervalued. Contrarily,

our proposed BiGAT assigns different weights to different neighbors with an attention mechanism, which leverages both inbound and outbound directional contextual information fully.

For hierarchical embedding aggregations, some existing studies that leverage multi-layered embedding approach fail to consider the model feasibility in large-scale KG training. For instance, DCGCN (45) uses dense connectivities to concatenate all preceding layer outputs as inputs towards each layer to learn multi-layer embeddings. Even though the model architecture seems reasonable, this approach drastically increases the number of parameters which impedes its applicability towards large-scale KGE tasks. In this study, we provide a much more feasible, scalable and elegant solution, BiGAT, to model multi-hop bi-directional neighbors by leveraging bidirectional embedding propagation and hierarchical embedding aggregation.

3.3 The Proposed BiGAT Model

In this section, we first present the problem formulation for the KGE task and list out some important notations that are used in the following chapter.[1] Following that, we introduce the preliminaries of KBGAT (104), which serves as a prior study and strong baseline of GAT model for the KGE task. Last but not least, we present

[1]We use upper-case bold letter, lower-case bold letter and non-bold lower-case letter to represent matrix, vector and scalar, respectively.

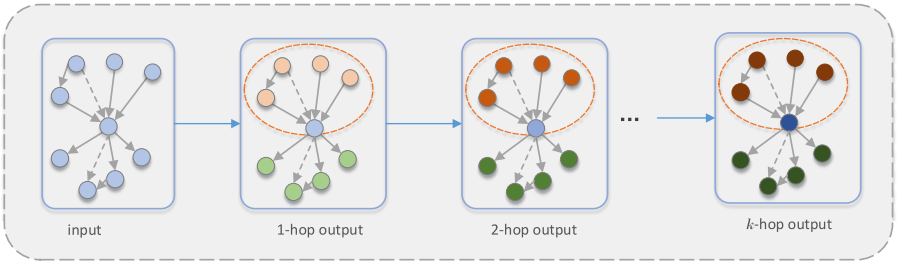


Figure 3.1: The structure of GAT used in KBGAT (104), which just considers the inbound-directional neighbors and k-th hop output to learn the embeddings.

the details of the modules of our proposed BiGAT model.

3.3.1 Problem Formulation

In this study, we make $\mathcal{E}$ and $\mathcal{R}$ denote a set of entities and a set of relations, respectively. A knowledge graph $\mathcal{G}$ refers to a directed graph with a group of factual triples. For each triple in $\mathcal{G}$, it is represented in a form of $(e_i, r_{i,j}, e_j)$, where $e_i, e_j \in \mathcal{E}$ stands for the head and tail entities, and $r_{i,j} \in \mathcal{R}$ stands for a relation of e_i to e_j. Mathematically, we denote the entity embedding matrix in a form of $\mathbf{E} \in \mathbb{R}^{|\mathcal{E}| \times d}$, where $|\mathcal{E}|$ refers to the number of entities in a KG, and d refers to the dimension of each entity embedding. Similarly, we denote the relation embedding matrix in the form of $\mathbf{R} \in \mathbb{R}^{|\mathcal{R}| \times d}$, where $|\mathcal{R}|$ stands for the number of relations existed in the KG. In our BiGAT model, the architecture takes $\mathbf{E}$ and $\mathbf{R}$ as inputs, and uses new embedding matrices $\mathbf{E}'$ and $\mathbf{R}'$ for entities and relations as outputs. The KGE task

aims to predict missing entities of $(e_i, r, ?)$ or $(?, r, e_j)$ with new embedding matrices $\mathbf{E}'$ and $\mathbf{R}'$. The list of notations used in this chapter is presented in Table 3.1.

Table 3.1: A list of notations used in this paper.

Name	Notation	Description
Knowledge Graph	$\mathcal{G}$	A set of triples in the format $(e_i, r_{i,j}, e_j)$
Entities	$\mathcal{E}$	A set of entities in KG
Relations	$\mathcal{R}$	A set of relations in KG
Input Entity Embedding	$\mathbf{E}$	The input embedding matrix of entities
Input Relation Embedding	$\mathbf{R}$	The input embedding matrix of relations
Output Entity Embedding	$\mathbf{E}'$	The embedding matrix of entities output by our model
Output Relation Embedding	$\mathbf{R}'$	The embedding matrix of relations output by our model
Embedding Size	d	The dimension of entity and relation embeddings
Entity Number	$\lvert\mathcal{N}\rvert$	The number of entities in KG
Relation Number	$\lvert\mathcal{R}\rvert$	The number of relations in KG
Attention Head Number	P	The number of attention head in our model
Hop Number	k	The number of BiGAT layers of our model
Loss Function	$\mathcal{L}$	The loss function of our model

3.3.2 Method

Graph attention networks (GATs) (170) have gained its popularity among graph learning in the past few years. Compared to the conventional models, GATs are capable of capturing more information-rich neighboring features and assigning various

attention weights to different nodes according to their importance. However, the vanilla GAT model (170) ignores relation features which serve as an integral part of KGs.

To address the above issue, KBGAT(104) was introduced as an extension of GATs to integrate entity and relation features in an attention mechanism. The Figure 3.1 illustrates the general KBGAT framework. The KBGAT first captures multi-hop neighboring information from the inbound direction of a given entity. After that, it utilizes the k-hop graph attention results generated in the last output layer to learn embeddings. Even though the model is largely optimized compared to the vanilla GAT approach, the KBGAT model performance is still restricted due to two weaknesses. First, the KBGAT model only considers neighbors from the inbound-direction of the given entity while ignoring the neighbors towards the outbound-direction, also known as neighbors that are connected from this entity. This one directional consideration limits the expressiveness of learnt embeddings, and restricts their capability of capturing sufficient neighborhood information. Second, the KBGAT model solely utilizes the k-th hop graph attention output to learn embeddings for those distant nodes. This causes the underestimation of contributions from each single hop, and leads to massive early-stage information loss at graph attention stage. For the purpose of addressing all above limitations of KBGAT, we propose a novel bidirectional

graph attention network (BiGAT) to learn the hierarchical neighbor propagation in this study.

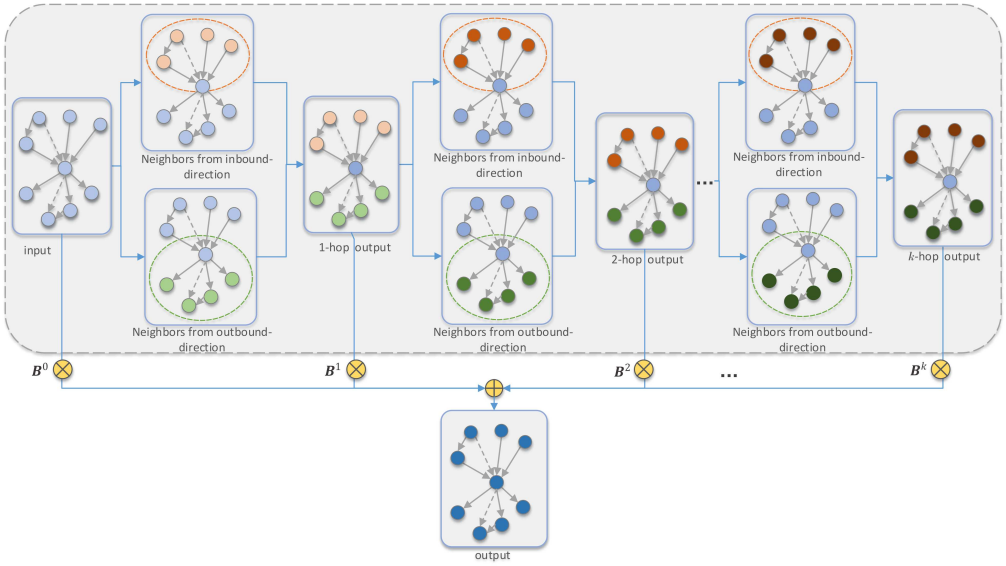


Figure 3.2: The framework of our proposed BiGAT. The orange circles and the green circles represent the neighbors from inbound-direction and outbound-direction respectively. The solid lines denote 1-hop relations while the dash lines denote the auxiliary relations for multi-hop neighbors. $\mathbf{B}^l$ $(l = 0, \cdots, k)$ is the attention weights of the l-th layer for all entities.

3.3.2.1 Bidirectional Embedding Propagation

In Figure 3.2, for each given entity e_i, the neighbors are classified to two different groups according to their directions of relations and edges. For each neighborhood

triple $\tau_{im} = (e_i, r_{i,m}, e_m)$ travels from the outbound-direction, we perform concatenation on their embedding vectors. Then we exercise a linear transformation to get the representation of the triple τ_{im}, computed as:

$$\tau_{im} = \mathbf{W}_1[\mathbf{e}_i||\mathbf{r}_{i,m}||\mathbf{e}_m] \tag{3.1}$$

where $||$ denotes the concatenation operation, $\mathbf{e}_i$, $\mathbf{e}_m$ represent the embedding vectors for entities e_i and e_m and $\mathbf{r}_{i,m}$ is the embedding vector for relation $r_{i,m}$. $\mathbf{W}_1$ is the outbound-directional transformation parameter, which is shared by all neighbors from outbound-direction.

Similarly, for each neighborhood triple $\tau_{ji} = (e_j, r_{j,i}, e_i)$ that travels towards the center node from inbound-direction, the representation is computed by:

$$\tau_{ji} = \mathbf{W}_2[\mathbf{e}_j||\mathbf{r}_{j,i}||\mathbf{e}_i] \tag{3.2}$$

where $\mathbf{W}_2$ is the inbound-directional transformation parameter.

In order to learn those entities in distance, we design an auxiliary edge to connect these k-hop ($k >= 2$) neighbors by leveraging the multi-hop path-based composition introduced in PTransE (85). In Figure 3.2, we depicted the auxiliary edge of 2-hop neighbor as an example in dotted lines. Mathematically, for each multi-hop relation $(e_i \xrightarrow{r_{i,1}} e_1 \xrightarrow{r_{1,2}} e_2 \cdots \xrightarrow{r_{k-1,k}} e_k)$, we denote the direct connection between e_i and e_k as $\tau_{ik} = (e_i, r_{i,1} \circ r_{1,2} \circ r_{k-1,k}, e_k)$, where $\circ$ stands for an addition composition operator

that calculates the sum of all relation vectors in the path to get the composition representation. With the help of the path-based composition, the representation of triple τ_{ik} involving the k-th distant neighbors can be easily computed with Equation (3.1).

We aggregate the neighboring information from both inbound and outbound directions to generate the output embedding at the l-th layer. Then, we update the entity embeddings by concatenating neighboring information from both directions and applying a linear transformation to it. After that, we apply multi-head attention (169) to our BiGAT model, which is denoted as:

$$\mathbf{e}_i^{l+1} = \sum_{p=1}^{P} \mathbf{W}_3^p \left[\sum_{\tau_{im} \in \mathcal{N}_i^{out}} \alpha_{im}^p \tau_{im}^l \Big|\Big| \sum_{\tau_{ji} \in \mathcal{N}_i^{in}} \alpha_{ji}^p \tau_{ji}^l \right] \tag{3.3}$$

where P is the number of attention head, $\mathbf{W}_3^p$ is the parameter for the p-th attention head, and α_{im}^p and α_{ji}^p are the attention weights for τ_{im}^l and τ_{ji}^l, respectively.

In order to compute the attention weights, we apply a linear transformation and a non-linear activation (e.g., LeakyRelu (93)) on them. For purpose of comparing results across different triples, we apply a softmax function to the attention score normalization. Taking the triple τ_{im} as an example, we compute the attention scores using the following formula:

$$\alpha_{im}^p = \frac{\exp(\text{LeakyRelu}(\mathbf{W}_4^p \tau_{im}))}{\sum_{\tau_{im'} \in \mathcal{N}_i^{out}} \exp(\text{LeakyRelu}(\mathbf{W}_4^p \tau_{im'}))} \tag{3.4}$$

where $\mathbf{W}_4^p$ is the linear transformation parameter. We choose LeakyRelu as the activation function since it has been widely used in GAT (170). The attention weight α_{ji} for neighbors from inbound-direction is computed in a similar way.

3.3.2.2 Hierarchical Embedding Aggregation

We collect multi-hop neighborhood information from both the inbound and outbound directions by stacking multiple BiGAT layers. In prior studies, GAT (170) and KBGAT (104) only apply the final aggregated output of the k-hop graph attention to the entity embedding learning process. This could directly lead to underrating the contributions of those neighbors at early hops. As the number of layers increases, some of those significant information from early stage embedding are lost during the graph attention steps. And it directly leads to severe limitations on model flexibility. One simple solution to address the information loss problem above is to concatenate all the preceding outputs as input towards each of the next layers using dense connectivities, as demonstrated in DCGCN (45). However, the dense connectivity operations involve much larger number of parameters, which results in infeasibility to train over large-scale KGs on the KGE task.

In this section, we design an attention mechanism to concatenate the output embeddings from all preceding BiGAT layers, which manages to take full advantage of

each hop embedding information. Thus, it captures more representative hierarchical embedding information rather than the k-th hop solely, and alleviates the problem of excessive information loss at early stages. As illustrated in Figure 3.2, the final entity embeddings of the input KG is computed by using a weighted sum of entity embeddings from all BiGAT layers, which is denoted as:

$$\mathbf{e'}_i = \sum_{l=0}^{k} \beta_i^l \mathbf{e}_i^l \tag{3.5}$$

where $\mathbf{e}_i^l$ is the embedding of i-th entity in the l-th layer and β_l is the attention weight for $\mathbf{e}_i^l$. To compute the attention weights, we consider both the entity embedding and its hop information. We use $k+1$ vectors to represent the hop embeddings which is used to encode the position information of each layer. The hop embedding is denoted as $\{\mathbf{h}^l | l \in (0, 1, \cdots, k)\}$. Then, the attention score is computed as:

$$\beta_i^l = \frac{\exp(\mathbf{W}_5(\mathbf{e}_i^l + \mathbf{h}^l))}{\sum_{l'=0}^{k} \exp(\mathbf{W}_5(\mathbf{e}_i^{l'} + \mathbf{h}^{l'}))} \tag{3.6}$$

where $\mathbf{W}_5$ is the parameter of the attention function.

Given that, the final entity embedding $\mathbf{E'}$, which considers neighboring information from both inbound and outbound directions and captures rich hierarchical features from each hop, is obtained. Formally, the update rule for entity embedding is denoted as:

$$\mathbf{E'} = \sum_{l=0}^{k} \mathbf{B}^l \mathbf{E}^l \tag{3.7}$$

where $\mathbf{B}^l = \{\beta_1^l, \cdots, \beta_{|\mathcal{E}|}^l\}$ is the attention weights of the l-th layer for all the entities, and $\mathbf{E}^l$ is the entity embedding in the l-th layer. Here, $\mathbf{E}^0$ is the input embedding $\mathbf{E}$.

For each BiGAT layer, we apply a linear transformation to update the relation embeddings, which is formally denoted as:

$$\mathbf{R}^{l+1} = \mathbf{W}_6 \mathbf{R}^l \tag{3.8}$$

where $\mathbf{W}_6$ is the parameter of the linear transformation. Here, $\mathbf{R}^0$ is the input relation embedding $\mathbf{R}$ and the final relation embedding is computed with attention mechanism in a similar way:

$$\mathbf{R}' = \sum_{l=0}^{k} \mathbf{B}^{rl} \mathbf{R}^l \tag{3.9}$$

where $\mathbf{B}^{rl} = \{\beta_1^{rl}, \cdots, \beta_{|\mathcal{R}|}^{rl}\}$ is the attention weights of the l-th layer for all the relations.

3.3.3 Training

Once the final entity embedding matrix $\mathbf{E}'$ and relation embedding matrix $\mathbf{R}'$ are derived, we apply a multiclass log-loss (71) to optimize our proposed BiGAT model. Let $\tau = (e_i, r_{i,j}, e_j)$ denote a triple in $\mathcal{G}$, and let τ^- denote an invalid triple by replacing either e_i or e_j with every other entity. The scoring function for each triple is thus computed as:

$$f(\tau) = \text{Relu}(\mathbf{W}_7[\mathbf{e}_i' || \mathbf{r}_{i,j}'])^T \cdot \text{Relu}(\mathbf{W}_8 \mathbf{e}_j') \tag{3.10}$$

where Relu (40) is a non-linear activation function, $\mathbf{W}_7$ is the parameter, $\mathbf{e}'_i$, $\mathbf{e}'_j \in \mathbf{E}'$ are the final entity embeddings for e_i, e_j, and $\mathbf{r}'_{i,j} \in \mathbf{R}'$ is the final relation embedding for $r_{i,j}$.

The final loss function of BiGAT is defined as follows:

$$\mathcal{L} = -f(\tau) + \log\left(\sum_{\tau^- \in \mathcal{G}_\tau^-} \exp(f(\tau^-))\right) \tag{3.11}$$

where $\mathcal{G}_\tau^-$ denotes the collection of invalid triples.

3.4 Experiments

In this section, we list out the datasets, evaluation metrics and parameter settings that are used to evaluate our model first. Then, we present the extensive experimental results and make comparisons between our BiGAT and other SOTA models. Finally, in order to verify the significance of each module component of our BiGAT model, we conduct ablation studies to uncover more details.

3.4.1 Datasets

In this study, we use four publicly available benchmark datasets, WN18RR (27), FB15k-237 (166), NELL-995 (207) and Kinship (83), to test out the effectiveness of our proposed model BiGAT. The WN18RR (27) is a refined version of WN18 (15) which is a subset of WordNet. The FB15k-237 (166) is a revised version of FB15k

(15) which is a subset of Freebase. Some prior studies (3, 27, 166) have found that the original WN18 and FB15k datasets suffer from the problem of inverse relations which can cause data leakage. Thus, the refined versions of both WN18RR and FB15k-237 datasets got publicized to discard these inverse relations, and to make the bidirectional neighboring information more prominent in the KGE task. Each of the four datasets is further decomposed into three subsets for training, validation and testing purposes, as mentioned in (9, 104, 106). The details of the four datasets are presented in the table below.

Table 3.2: Description of the benchmark datasets.

	#Entity	#Relation	#Train	#Valid	#Test
WN18RR	40,943	11	86,835	3034	3134
FB15k-237	14,541	237	272,115	17,535	20,466
NELL-995	75,492	200	149,678	543	3992
Kinship	104	25	8544	1068	1074

3.4.2 Evaluation Metrics

As we mentioned in previous section, the goal of the KGE task is to learn effective low-dimensional representations of entities and relations for link predictions. A com-

mon example of the task could be to predict the head entity e_i given $(?, r_{i,j}, e_j)$ or predict tail entity e_j given $(e_i, r_{i,j}, ?)$. Here, we make predictions on all the head and tail entities of the triples in testing set, by ranking the collection of candidate entities in KG.

In order to evaluate the model performance, we employ three widely used evaluation metrics, namely the mean rank (MR), the mean reciprocal rank (MRR) and the Hits@N (the correct percentage in the top N results e.g., Hits@3, Hits@10), to the study. We evaluate all the models with a "Filtered" setting used in the prior work (15). The setting is able to filter out other correct triples existed in the training, validation and testing sets. Specially, given a triple $(e_i, r_{i,j}, e_j) \in \mathcal{T}$, where the $\mathcal{T}$ is a collection of testing triples, we make $rank_i$ and $rank_j$ to denote the ranking of head entities and tail entities, respectively. Thus, we can compute the evaluation metrics as follows:

$$\text{MR} = \frac{1}{2|\mathcal{T}|} \sum_{(e_i, r_{i,j}, e_j) \in \mathcal{T}} rank_i + rank_j \tag{3.12}$$

$$\text{MRR} = \frac{1}{2|\mathcal{T}|} \sum_{(e_i, r_{i,j}, e_j) \in \mathcal{T}} \frac{1}{rank_i} + \frac{1}{rank_j} \tag{3.13}$$

$$\begin{aligned} \text{Hits@}N = \frac{1}{2|\mathcal{T}|} \sum_{(e_i, r_{i,j}, e_j) \in \mathcal{T}} & I[rank_i \leq N] \\ & + I[rank_j \leq N] \end{aligned} \tag{3.14}$$

where $I[*] = 1$ if the condition is satisfied else $I[*] = 0$. Lower MR, higher MRR and Hits@N indicate the better performance.

3.4.3 Parameter Settings

Embedding: In order to make fair comparisons, the embedding size applied in BiGAT is set to be consistent with the prior work (104), and the dimensions of the entities and relations embeddings are set as 200 for kinship, and 100 for FB15k-237, WN18RR and NELL-995.

Hyper-parameters: We use Adam (64) to optimize the BiGAT parameters, and use the grid search to tune other hyper-parameters on validation set which is determined by the values from the best performed Hits@10 records. Specifically, we search the initial learning rate from $\{0.001, 0.0005, 0.0002, 0.0001\}$, the batch size from $\{128, 256, 512\}$, the dropout rate from $\{0.5, 0.6, 0.7, 0.8\}$, the number of attention heads from $\{1, 2, 3\}$, the number of layers from $\{1, 2, 3, 4\}$. As a result, we select the initial learning rate as 0.0005 for FB15k-237 and kinship, and 0.0002 for WN18RR and NELL-995. The batch size is set as 512. The dropout rate is set as 0.5, the number of attention heads is set as 2, and the number of BiGAT layers is set as 3.

Training strategy: Most of the existing KGE models use negative sampling (99)

as the training strategy, which aims to randomly select some negative triples from the input KG. However, the performance of negative sampling is heavily dependent on the quality of selected negative triples. Instead, we apply a more feasible and effective training method, named 1vsAll training strategy (134), to train our BiGAT. The 1vsAll sees all the other false entities in KGs as negative head or tail entities to compute the objective function.

Experimental environment: In this study, the popular deep learning framework Tensorflow, version 1.14, is used to implement our model. All the experiments in this study are conducted on RTX 2080Ti GPUs. The datasets and codes are available at https://github.com/sherrybbalonsozhu.

3.4.4 Comparison and Results

We compare our proposed BiGAT model with several strong baseline models presented in the related work. The baselines include the translation-based methods (e.g., TransE (15), TransH (190), TransR (88) and etc.), the semantic matching-based models (e.g., Rescal (111), DistMult (216), ComplEx (167) and etc.), the CNN-based models (e.g., ConvE (27), InteractE (168) and etc.), the Transformer-based models (e.g., MLMLM [49]), and the neighborhood-based models (e.g., R-GCN (138) and KBGAT (104)). The experimental results of our BiGAT and the respective baseline

models on the two benchmark datasets (WN18RR and FB15k-237) are reported in Table 3.3. [2]

As illustrated in Table 3.3, our BiGAT consistently achieves the best performance on both the WN18RR and FB15k-237 datasets among various evaluation metrics, except the Hits@3 and Hits@10 on WN18RR and the MR on FB15k-237. In comparison with the conventional methods, including translation-based and semantic matching models of TransE and DistMult, our proposed BiGAT obtains significant improvements. In comparison with the CNN-based methods of ConvE and ConvKB, our proposed BiGAT still achieves remarkable improvements across various metrics. In comparison with the neighborhood-based models of KBGAT especially, BiGAT also achieves great performance gains. These comparison results confirm our observation that both the bidirectional multi-hop neighborhood information and the early-stage embedding information modules play significant role in the KGE task.

Aside from that, we also present our experimental results on NELL-995 and

[2]The recent proposed models (ConvKB, CapsE and KBGAT) suffer from the inappropriate evaluation issue, which has been found and discussed in (162). In order to make a fair comparison, we take the results from (162) for these models. Besides, we also note that the previous work GAAT (182) reports the better results than our BiGAT on FB15k-237. However, the results reported in GAAT (182) still suffer from the inappropriate evaluation issue since it uses a CapsE model as decoder. Moreover, the KBGAT (104) and GAAT (182) involves the test data leakage during negative sampling. Specifically, they will filter out the negative triples which will appear in the test data. This issue has been widely discussed in Github: https://github.com/deepakn97/relationPrediction/issues/28. GAAT does not provide the source code and the details of the implementations, making it difficult to directly verify the results. Therefore, we do not compare GAAT and our BiGAT in Table 3.3.

Table 3.3: KGE results for KGE on WN18RR and FB15k-237 test sets using various evaluation metrics. * indicates that the results are taken from (27), † denotes that the results are taken from (162), all other results are taken from the original papers. The best results are in **bold** while the second best scores are in underline.

Methods	WN18RR					FB15k-237				
	MR	MRR	Hits@1	Hits@3	Hits@10	MR	MRR	Hits@1	Hits@3	Hits@10
TransE (15)*	2300	.243	.042	.441	.532	323	.279	.198	.376	.465
DistMult (216)*	5110	.430	.390	.440	.490	254	.241	.155	.263	.419
ComplEx (167)*	5261	.440	.410	.460	.510	339	.247	.158	.275	.428
ConvE (27)	4187	.430	.400	.440	.520	244	.325	.237	.356	.501
R-GCN (138)	-	-	-	-	-	-	.249	.151	.264	.417
KBGAN (16)	-	-	.213	-	.481	-	.278	-	-	.458
NKGE (176)f	4170	.450	.421	.465	.526	237	.330	.241	.365	.510
CACL (116)	3154	.472	-	-	.543	235	.349	-	-	.487
ConvKB (106)†	3433	.249	-	-	.524	309	.243	-	-	.421
CapsE (108)†	718	.415	-	-	.559	403	.150	-	-	.358
KBGAT (104)†	1921	.412	-	-	.554	270	.157	-	-	.331
TransMS (218)	6523	-	-	-	.460	249	-		-	.445
A2N (9)	-	.450	.420	.460	.510	-	.317	.232	.348	.486
SACN (140)	-	.470	.430	.480	.540	-	.350	.260	.390	.540
TuckER (8)	-	.470	.443	.482	.526	-	.358	.266	.394	.544
MuRE (7)	-	.475	.436	.487	.554	-	.336	.245	.370	.521
pLogicNet (126)	3408	.441	.398	.446	.537	173	.332	.231	.367	.524
GRank (31)	-	.470	.437	.482	.539	-	.322	.239	.352	.489
D4-Gumbel (209)	-	.486	.442	.505	.557	-	.300	.204	.332	.496
RotatE (159)	3340	.476	.428	.492	.571	177	.338	.241	.375	.533
QuatE (226)	2314	.488	.438	.508	.582	**87**	.348	.248	.382	.550
GC-OTE (165)	-	.491	.442	.511	.583	-	.361	.267	.396	.550
InteractE (168)	5202	.463	.430	-	.528	172	.354	.263	-	.535
HAKE (230)	-	.497	.452	.516	.582	-	.346	.250	.381	.542
ATTH (18)	-	.486	.443	.499	.573	-	.348	.236	.384	.540
AutoETER (113)	-	-	-	-	-	170	.344	.250	.382	.538
MLMLM (26)	1603	.502	.439	**.542**	**.611**	411	.259	.187	.282	.403
BiGAT (ours)	**542**	**.505**	**.458**	.521	.594	91	**.366**	**.270**	**.402**	**.558**

Table 3.4: KGE results on NELL-995 and Kinship. The comparisons are from (104). We reproduce the results of KBGAT using the source code provided in (162).

Methods	**NELL-995**					**Kinship**				
	MR	MRR	Hits@1	Hits@3	Hits@10	MR	MRR	Hits@1	Hits@3	Hits@10
TransE (15)	2100	.401	.344	.472	.501	6.80	.309	.009	.643	.841
DistMult (216)	4213	.485	.401	.524	.610	5.26	.516	.367	.581	.867
ComplEx (167)	4600	.482	.399	.528	.606	2.48	.823	.733	.899	.971
ConvE (27)	3560	.491	.403	.531	.613	2.00	.833	**.738**	.917	.981
R-GCN (138)	7600	.120	.082	.126	.188	25.92	.109	.003	.088	.239
KBGAT (104)	1087	.469	.386	.522	.609	2.61	.755	.631	.859	.968
BiGAT (ours)	**761**	**.531**	**.456**	**.574**	**.662**	**1.94**	**.835**	.737	**.925**	**.985**

Kinship in Table 3.4. From Table 3.4, we find the similar patterns and derive similar conclusions as we do from Table 3.3. The results on these two datasets verify the robustness of our BiGAT again.

It is worth noting that our BiGAT model is significantly different from the baseline KBGAT (104) model in two folds: first, rather than gathering neighborhood from one direction solely, our BiGAT model aggregates hierarchical neighbors from both directions; second, rather than training in a pipe line with a GAT as encoder and a ConvKB model as decoder, our BiGAT model can be trained in an end-to-end manner in a unified framework. These differences enable our model to perform better than the KBGAT.

3.4.5 Performance for Different Relations

As illustrated in Table 3.5 and Table 3.6, we analyze the model performance for different relations on both the WN18RR and FB15k-237 datasets. The Table 3.5 contains a list of all details of the Hits@10 results of the KBGAT and our BiGAT model for each relation. The Table 3.6, following (15), contains a list of all details of the four classified relations, namely the one-to-one, one-to-many, many-to-one and many-to-many. It also presents the Hits@10 results for different relation categories. From these experimental results stated above, we can see that our BiGAT model derives better results for many relations on the WN18RR dataset. On the FB15k-237 dataset, our BiGAT model also outperforms KBGAT across all relation categories. These results demonstrate that our BiGAT model is capable of yielding superior performance over various relations.

3.4.6 Ablation Study

Aside from all above, we conduct ablation studies on four benchmark datasets to demonstrate each module's contribution towards the BiGAT model. The results are shown in Table 3.7. The "Inbound GAT" stands for the BiGAT model that only takes in inbound-directional neighbors, while the "Outbound GAT" stands for the BiGAT model that only takes in outbound-directional neighbors. Besides, the "In-

Table 3.5: The relation prediction results for each relation on the WN18RR test set using H@10.

	Predicting Head		Predicting Tail	
Relations	KBGAT	BiGAT	KBGAT	BiGAT
hypernym	**0.210**	0.108	0.304	**0.435**
derivationally_related_form	0.964	**0.966**	0.964	**0.968**
instance_hypernym	**0.339**	0.330	0.750	**0.893**
also_see	0.696	**0.804**	0.609	**0.732**
member_meronym	**0.537**	0.504	**0.272**	0.256
synset_domain_topic_of	0.245	**0.391**	0.609	**0.818**
has_part	0.296	**0.556**	0.195	**0.314**
member_of_domain_usage	0.500	**0.909**	**0.273**	0.227
member_of_domain_region	0.385	**0.808**	0.077	0.077
verb_group	0.974	0.974	0.949	**0.974**
similar_to	1.000	1.000	1.000	1.000

bound GAT (k-th hop)" and the "Outbound GAT (k-th hop)" represent the model performance without hierarchical information attached from inbound and outbound directions respectively. Specifically, the difference between the "Inbound/Outbound GAT" modules and the "Inbound/Outbound GAT (k-th hop)" modules is that the "Inbound/Outbound GAT" utilizes the hierarchical hop embedding information for updating the embedding representations, whereas the "Inbound/Outbound GAT (k-

Table 3.6: The relation prediction results for each relation category on the FB15k-237 test set using H@10. Following [Bordes et al., 2013], we classify relations into four groups: one-to-one, one-to-many, many-to-one and many-to-many.

	Predicting Head		Predicting Tail	
Relation categories	KBGAT	BiGAT	KBGAT	BiGAT
one-to-one	0.473	**0.635**	0.447	**0.598**
one-to-many	0.328	**0.658**	0.091	**0.284**
many-to-one	0.102	**0.444**	0.57	**0.880**
many-to-many	0.309	**0.469**	0.355	**0.596**

th hop)" solely utilizes the k-th hop output for updating the embedding representations.

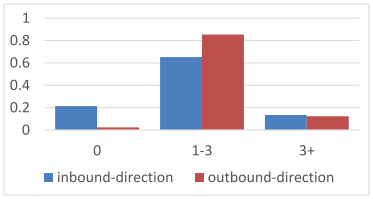
Table 3.7: Ablation study for all datasets.

Methods	**WN18RR**		**FB15k-237**		**NELL-995**		**Kinship**	
	MR	Hits@10	MR	Hits@10	MR	Hits@10	MR	Hits@10
Inbound GAT	584	.590	99	.544	797	.652	2.10	.984
Outbound GAT	602	.591	96	.548	824	.659	2.07	.982
Inbound GAT (k-th hop)	607	.581	107	.531	873	.644	2.74	.965
Outbound GAT (k-th hop)	605	.582	104	.533	866	.643	2.79	.962
BiGAT (k-th hop)	603	.587	102	.536	857	.655	2.15	.980
BiGAT	**542**	**.594**	**91**	**.558**	**761**	**.662**	**1.94**	**.985**

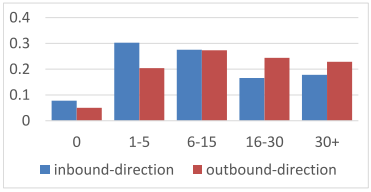
In this study, we report the experimental results of the ablation study using

the same parameter settings. According to the Table 3.7, our proposed BiGAT outperforms both the "Inbound GAT" and "Outbound GAT", which reassures our findings of the significance of neighborhood information over the KGE task. Furthermore, our study also shows that the inclusion of outbound-directional neighborhood information can further strengthen model performance, indicated by the comparisons between "Inbound GAT" vs. BiGAT. This verifies our conclusion that outbound-directional neighborhood information plays a role in better relation predictions. Compared to the "BiGAT (k-th hop)", our BiGAT model obtains remarkable performance gains on all four datasets. Besides, the "Inbound GAT" and "Outbound GAT" outperform both the "Inbound GAT (k-th hop)" and "Outbound GAT (k-th hop)" significantly. These comparisons between stacking of modules show the impact of hierarchical hop embedding on preventing early-stage information loss, as well as the effectiveness of the hierarchical information for the KGE task.

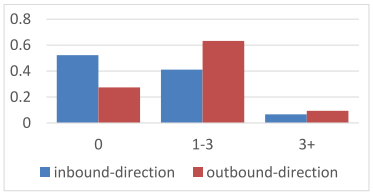
Figure 3.3 reports the neighbor distribution over all datasets. On one hand, as indicated in the figure, a significant amount of nodes hold $>= 1$ neighbors from both directions. This demonstrates the necessity of including the auxiliary edges and the bidirectional neighborhood information for KGE task. On the other hand, we notice that there is a large number of nodes with zero neighbors from the inbound-direction on sparse datasets, such as WN18RR and NELL-995. The nature of sparsity can



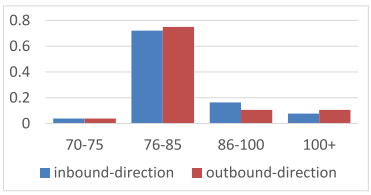
(a) WN18RR.



(b) FB15k-237.



(c) NELL-995.



(d) Kinship.

Figure 3.3: Neighbor distribution from inbound-direction and outbound-direction for all datasets.

be alleviated by adding in neighbors from the outbound-direction. As a result, the BiGAT model can capture richer contextual information and further advance embedding representations.

3.4.7 Complexity Analysis

Even though our BiGAT model has demonstrated its effectiveness and robustness, it is also necessary to involve complexity analysis to show our model efficiency. We compute the number of parameters of the baselines and our BiGAT model to investigate the model complexity of two mechanisms, namely the bidirectional neighborhood

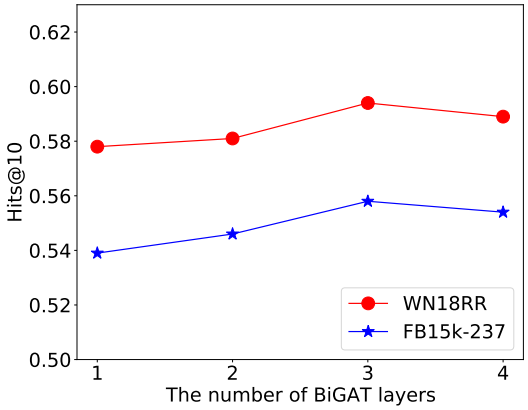


Figure 3.4: Hits@10 results on WN18RR and FB15k-237 with different number of BiGAT layers.

information and the hierarchical hop embedding. As presented in Table 3.8, our BiGAT only needs a slight increase of parameters on all datasets (e.g., 4.805M vs. 4.517M on WN18RR and 2.330M vs. 2.042M on FB15k-237). Although two mechanisms are stacked in our model, our BiGAT is still considered as parameter-efficient. In fact, the computational complexity mainly relies on the entity embeddings, due to the large number of entities in KGs.

3.4.8 Impact of different BiGAT layers

Apart from all above, we also conduct experiments on WN18RR and FB15k-237 to learn how the multi-hop neighborhood information and different BiGAT layers affect

Table 3.8: The number of parameters on four datasets.

Methods	**WN18RR**	**FB15k-237**	**NELL-995**	**Kinship**
Inbound/Outbound GAT	4.518M	2.043M	7.849M	2.473M
Inbound/Outbound GAT (k-th hop)	4.517M	2.042M	7.848M	2.472M
BiGAT (k-th hop)	4.804M	2.329M	8.135M	3.617M
BiGAT	4.805M	2.330M	8.136M	3.618M

the model performance. The Figure 3.4 shows the Hits@10 results with different BiGAT layers settings on the WN18RR and FB15k-237 datasets. As we can see from the figure, our BiGAT model performance increases as the number of BiGAT layers increases from 1 to 3. We conclude that our model can obtain more useful multi-hop information by stacking more BiGAT layers. However, the Hits@10 results stagnates at 3, as nodes further apart could be less relevant and thus introduce more noise that hurts the overall model performance.

3.5 Conclusion and Future Work

In this chapter, we propose a novel bidirectional graph attention network (BiGAT) for single graph learning on the KGE task. Our BiGAT takes in the neighborhood information from both inbound and outbound directions and propagates the bidirectional information to learn multi-hop feature embeddings in an iterative manner.

On top of that, we apply an attention mechanism to integrate the hierarchical hop embedding information from every single BiGAT layer into a unified representation. The hierarchical representation can effectively prevent early-stage information loss and generate more promising embeddings. The experimental results have shown the superiority of our proposed BiGAT model over all SOTA methods on four KGE benchmark datasets.

Last but not least, we believe that there are still two ways to further deepen this study. Firstly, we could extend the proposed BiGAT model to incorporate text information with knowledge embedded in KGs to further improve our model. Secondly, we could also apply our BiGAT to graph-based downstream applications, such as question answering over incomplete KGs (208).

4 Learning on Multiple Graphs: GATs for Cross-Lingual Knowledge Graphs Entity Alignments

In this chapter, we dive into the second module of this thesis on GATs for multiple graph learning. We introduce two advanced GAT-based methodologies, namely the Contextual Alignment Enhanced Cross Graph Attention Network (CAECGAT) and the Dual Gated Graph Attention Network with Dynamic Iterative Training (DuGa-DIT), to tackle the persisting issues in the CEA task.

4.1 Introduction

KG refers to a knowledge base of interlinked entities collection where the data are stored and integrated in graph structure. These entities and relations are represented in the Resource Description Framework (RDF) format. KG has been served

as the core and common denominator of many successful downstream tasks, such as search engines, personalised search recommendation systems (29, 98, 185, 191), social network sites, and common NLP tasks of question answering (195) and language modelling (231). On one hand, open-sourced KGs, including Freebase (13), DBpedia (6), WordNet (100), Wikidata (173), OpenCitations (121), ScriGraph (57) and Microsoft Academic KG (34), are widely adopted to real world applications. On the other hand, as more human capital and crowd-sourcing strategy being invested in KG constructions, many cross-lingual KGs have also grown substantially, such as DBpedia (6), YAGO (130), BabelNet (105) and xlore (187) .

Even though KGs have been applied to many scenarios and downstream tasks, the large amount of missing facts and links still persist in those existing KGs. Thus, the incomplete characteristic of KGs also affects the effectiveness of systems and algorithms that use KGs as the denominator. Under certain sensitive scenarios, such as the clinical decision support system, missing key entities or significant relations between entities could result in serious consequences. In the context of cross-lingual KGs, it is thus important to align those distributed entities among KGs of different languages. Thus, a more vast in scale and higher in-coverage KG would be built to tackle single KG's incompleteness problem (62) and to facilitate other cross-lingual-based NLP tasks.

To address the above issues, learning effective KGE for CEA would be a cut-to-the-point solution. CEA is a task which aims to match equivalent entities across KGs in different languages. This research domain has attracted considerable attention and rapid growth in the past few years. KGE refers to a group of methods that can generate KGE and leverage those embedding representations to make missing link predictions. The research in KGE was kicked off with a group of translation-based models. Bordes et al. introduced TransE (15) in 2013. Shortly after, various extensions of it were introduced, including TransH (190), TransD (59) and TransR (88) etc.. In parallel, various semantic matching-based methods were also introduced, including RESCAL (111), Bilinear (58), DistMult (216), ComplEx (167), SME (14) and NTN (20) etc.. The expressiveness of these two conventional groups of models have always been restricted due to their shallow structures. Moreover, they rely heavily on qualitative machine translation and feature extractions for entity alignment, unless the size of embedding increases together. Later on, the research continues to evolve by introducing multi-layered CNN-based models to tackle the problem. Some examples of the models involved in this category include ConvE (27), ConvKB (106), InteractE (168), RSN (44) and CTEA (214). Comparably speaking, these CNN-based models outperform those conventional ones, despite they still could not model important structural information beyond the entities.

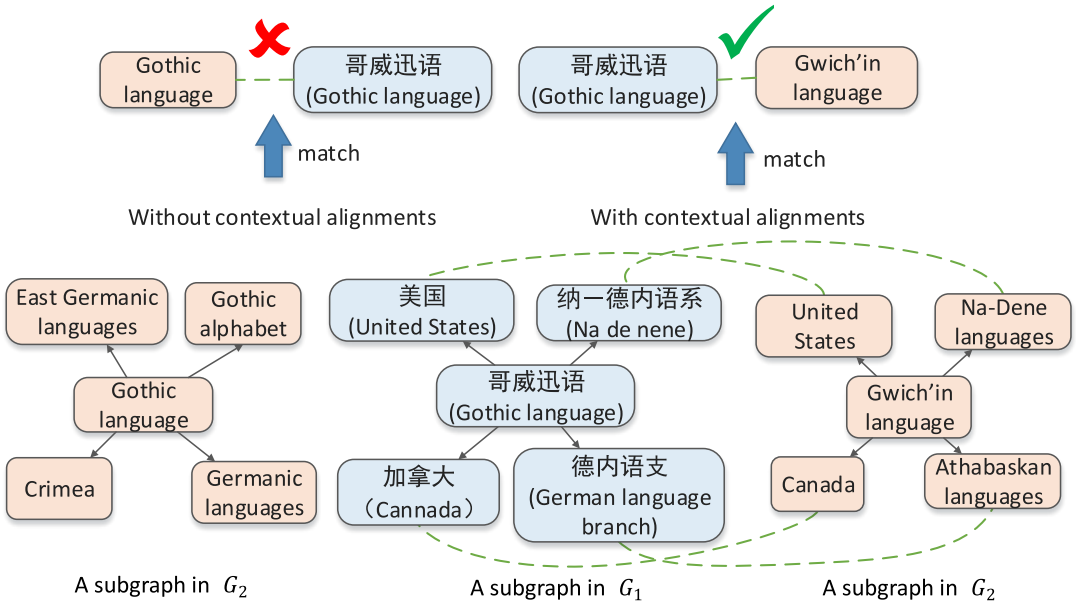


Figure 4.1: An example of cross-lingual entity alignment in DBP15K$_{ZH-EN}$ dataset. The translated surface strings for Chinese entities are provided in original DBP15K$_{ZH-EN}$ dataset (160). The dashed lines represents the pre-aligned neighbors.

Most recently, GNN-based models which could effectively learn and aggregate node's neighborhood information have shown their success in the KGE task. A2N (9), R-GCN (138), KBGAT (104), GraphMatch (212), GCN-Align (188), MRAEA (96), MuGNN (17), HMAN (217), and AliNet (163) are all lately proposed models belong to this category. However, most of these models are restricted by modeling each KG individually with a small number of pre-aligned entities served as seeds to anchor and link different KGE spaces. Some of these models even apply pre-aligned seed alignments in objective function optimization only, without making full use of the seed alignments to capture contextual alignment information. In other words, the information-rich pre-aligned links across KGs of different languages are largely under-used. These characteristics of the prior work could result in two issues: 1) The insufficiency of available seed alignments could restrain their model performances; 2) The rich contextual information among nodes are useful in KGE tasks but greatly under-investigated.

We intuitively believe that the equivalent entities across KGs of different languages should share as many pre-aligned neighbors as possible. As shown in Figure 4.1, although the "*(Gothic language)*" entity in Chinese should match with the "*Gothic language*" entity in English by literal machine translation. The actual entity alignment pair of "*(Gothic language)*" should be "*Gwich'in language*". If we learn

from contextual alignments existed among the DBP15K$_{ZH-EN}$, such as "*(United States)*" and "*United States*", "*(Canada)*" and "*Canada*", "*(Na de nene)*" and "*Na-Dene languages*", "*(German language branch)*" and "*Athabaskan languages*", one could easily infer that pair correctly given the entity's context and neighborhood. And it also proves the necessity of making full use of the contextual alignments for entity alignment task across KGs.

Given the problems stated above, GM-EHD-JEA (211) introduced a two stage reasoning and joint entity alignment model which makes predictions on model-confident alignment as extra input first, and then makes predictions on those remaining hard alignments. Although the algorithm design is rational, it requires large search space and results in extremely high computational cost. In this work, we propose two models to overcome these challenges.

First, we introduce Contextual Alignment Enhanced Cross Graph Attention Network (CAECGAT) to tackle the multiple graph learning on CEA task. The CAECGAT makes full use of pre-aligned seed alignments to propagate information across KGs and jointly learns KGE from KGs of different languages. Specifically, we design a cross graph attention (CGAT) layer, which includes both a cross-KG aggregation and an attention-based cross-KG propagation layer, to learn information across KGs. The cross-KG aggregation layer of CGAT serves to use pre-aligned seed entities to

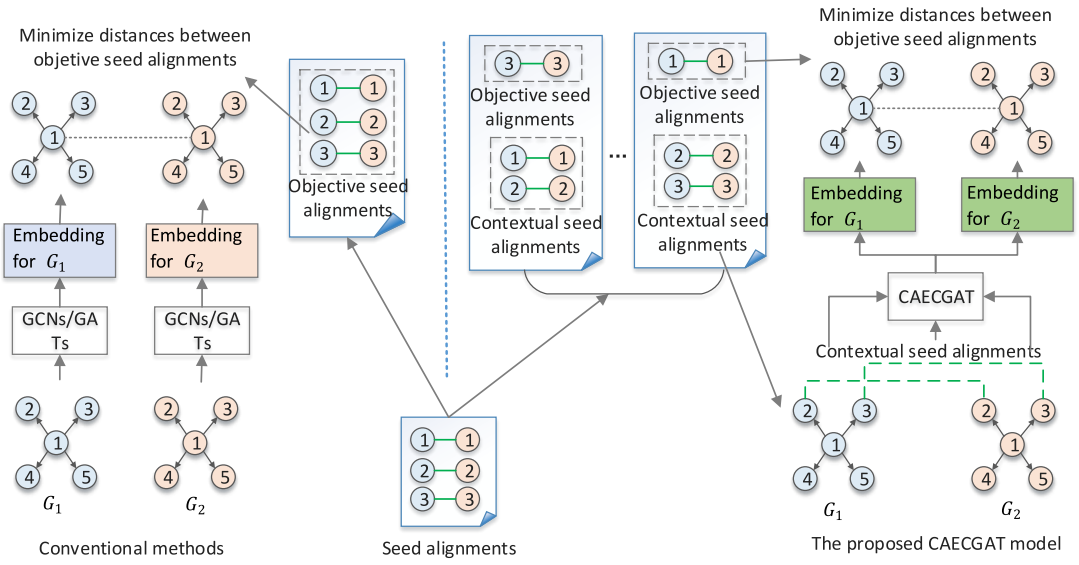


Figure 4.2: The comparison between conventional GNN-based methods and the proposed CAECGAT model. Our aim is to predict whether the $entity_1$ from G_1 and $entity_1$ from G_2 are two equivalent entities.

transfer information across KGs. This allows embeddings from different KGs to be mapped to the same semantic space with shared information. The attention-based cross-KG propagation layer serves to propagate important cross-KG information from entity's neighbors. Thus the semantic gap across KGs of various languages can be minimized.

As illustrated in Figure 4.2, we state that our proposed model is fundamentally different from conventional GNN models. Conventional GNN methods directly feed pre-aligned entity pairs to models as objective seed alignments for training. The contextual alignment information which propagates across KGs of different languages are neglected. However, CAECGAT divides the available seed alignments to both objective and contextual seed alignments at the training stage. The model selects one batch of seed alignments for objective purpose (e.g., $entity_1$ in Figure 4.2) and the remaining for contextual purpose (e.g., $entity_2$ and $entity_3$) iteratively. On one hand, the objective seed alignments still optimizes model parameters effectively. On the other hand, the contextual seed alignments fulfills cross-KG propagation of entity information and provides more pre-aligned information.

Second, based on the preliminary study of CAECGAT, we also extend our research and propose an improved method Dual Gated Graph Attention Network with Dynamic Iterative Training (DuGa-DIT). In Figure 4.1, if neighborhood alignment

of the "*(Gothic language)*" entity and the "*Gothic language*" entity does not exist, it becomes challenging for CAECGAT to perform cross-KG information propagation because of the insufficiency of pre-algined entity pairs. Our second model, DuGa-DIT, mitigates CAECGAT's problems of information propagation across KGs in a unified framework. Specifically, we propose a dual KG attention layer which consists of an intra-KG attention layer and a cross-KG attention layer. The intra-KG attention layer aims to learn features from node's neighborhood from each KG individually. While the cross-KG attention layer aims to consolidate alignment information across KGs of different languages. The combination of this dual KG attention layer allows both local neighborhood information and cross-KG information propagation to be performed from neighbors, even with multi-hops. Furthermore, we include dynamic iterative training process to our model, which allows score matrices of cross-KG attention to be updated dynamically. Thus, more information across KGs of different languages can be seamlessly transferred.

The key contributions of this line of work are summarized as follows.

- We propose a novel method, CAECGAT, to tackle the task of CEA. The CAECGAT learns embeddings and propagates information across cross-lingual knowledge graphs.

- For the purpose of capturing more cross-KG information, we propose a different

training strategy where we iteratively divide seed alignments into contextual seed alignments and objective seed alignments.

- The experimental results on the three benchmark datasets demonstrate that our proposed CAECGAT achieves remarkably better performance than the SOTA methods.

- Based on the preliminary study of CAECGAT, we propose an enhanced method, DuGa-DIT, for CEA. The DuGa-DIT learns cross-KG embeddings to bridge the semantic gaps between different KGs by leveraging both neighborhood features and cross-KG alignment information.

- We also develop a dynamic iterative training process to dynamically update the cross-KG attention score matrices by iteratively adding new seed alignments, which allows more cross-KG information to transfer across different KGs.

- We conduct extensive experiments on two benchmark datasets to evaluate DuGa-DIT. The experimental results demonstrate that our proposed method is effective and robust for the CEA task.

4.2 Related Work

In order to address the CEA task, we hereby discuss mainly two subfields of related work: KGE and CEA. The detailed discussions and analyses of these two groups of existing literature are presented as follows.

4.2.1 KG Embedding

KGE refers to embed entities and relations of KGs into continuous vector spaces. The goal of KGE is to inherit original structures of entities within the KGs, and to simplify the process of future manipulation. As the scale and the number of KGs grow extensively, the KGE task becomes more urgent to support further KG analyses.

4.2.1.1 Translation-based Methods

Translation-based models are one of the earliest and most important approaches to the KGE task. Bordes et al. proposed a pioneer model, TransE (15), to demonstrate the feasibility of KG modeling and to predict missing links, classify various triples, and model large-scale KGs on one-to-one relations. TransE follows an assumption of $h + r \approx t$, for all triples (h, r, t). Given the simple structure of TransE, it is unable to model complex relations among entities, such as one-to-N, N-to-one or N-to-N

relations. Shortly after, various extensions of TransE have been proposed (32, 50, 59, 88, 125, 190, 200, 218). Even though some of these model gain relatively better results, they also suffer from model complexity increase as trade off. TranSparse (60) takes advantage of the translation approach by designing a hybrid model to increase the expressiveness of embeddings from both labelled and unlabelled entities in KGs, and to address the heterogeneity and imbalance nature within KGs. Instead of using conventional transfer matrices, TranSparse leverages adaptive sparse matrices in computation. And it uses the number of entities linked by various relations to calculate the sparse degrees of these matrices.

4.2.1.2 Semantic Matching Methods

Semantic matching methods aim to calculate semantic similarities using vectors or matrices. RESCAL (111) is the first KGE model which leverages a three-way tensor factorization to compute triples. Bilinear (58) is a model where entities are associated with vectors and relations are modeled by matrices. DistMult (216) confirms the fact that the best relational semantics are captured when the respective embeddings are learnt by bilinear objectives. The model also characterizes the composition of relations by the multiplication of matrices. Taking a similar approach as DistMult, ComplEx (167), which is also known as an extension of RESCAL, uses complex

valued embeddings to tackle binary relational problems of symmetry and asymmetry. SME (14) uses both linear and bilinear mattching blocks in neural networks to encode the interactions among entities and relations. NTN (20) exposes the prior study (66) to a multi-lingual perspective. KrompaB (70) enhances experimental results on link prediction tasks by incorporating type-constraints prior knowledge into various SOTA latent variable methods. KR-EAR (87) models correlations and predictions on entities, relations and attributes to distinguish attributes and relations within a single knowledge base. Aside from all above, other prior work belong to this category include but are not limited to HOLE(109), ProjE(145), etc..

4.2.1.3 CNN-based Methods

CNN-based models aim to generate more expressive embeddings given its high parameter efficiency. ConvE (27) models local interactions between head entity and relation with a 2D convolutional operation. The model uses multiple non-linear future learning to reshape the head entity and relation into 2D matrices for semantic expressions. Similarly, by representing triples into a three-column matrix, ConvKB (106) filters out different feature maps. The feature maps are then consolidated to generate feature vector that represents triples before multiplying with weighting vectors to derive final scores. As a result, ConvKB not only captures global re-

lations among the KG, but also captures translation features between entities and relations. InteractE (168) enhances the model performance further by increasing interactions between entities and relations embeddings. RSN (44) and CTEA (214) are example models which utilize CNN-based deep neural network models to tackle the KGE task. Context-aware CNN (116) proposes to improve the KGE results by incorporating neighborhood information. CapsE (108), CoKE (179) and ReInceptionE (202) are also CNN-based KGE models belong to this category. Even though these CNN-based methods show promising results and performance improvement over translation-based and semantic matching models, most models belong to this category also share a common problem of modeling each triple independently. This characteristic causes plenty of latent information loss from KGs.

4.2.1.4 Methods Using Neighborhood Information

Neighborhood information-based models learns embeddings by absorbing entities' neighborhood information. Recently, a number of models released on this track have shown great performance gains (9, 104, 107, 138, 140, 176). A2N (9) obtains query specific neighbors using an attention mechanism. R-GCN (138) is the first model that leverages the GCNs to model multi-relational data among KGs. KBGAT (104) utilizes GAT (170) to assign different importance scores to different neighboring

nodes. As an extension of TransE, TransE-NMM (107) borrows the translation model concepts while taking in neighboring entities in the modelling process. NKGE (176) utilizes semantic and topological features of an entity to tackle the KGE task with a gating mechanism. SACN (140) learns neighborhood information via local aggregation to generate more expressive representations with embedded GCN and ConvE model concepts.

4.2.1.5 Methods Using Relation Paths/Literals

The last category of KGE models takes relation paths or literal approaches to learn and improve embeddings from KGs. DeepPath (207) leverages the reinforcement learning approach to capture multi-hop relation paths and global information. RTransE (37) and PTransE (86) attempt to leverage multi-step relational paths to complete KGs, while SimplE (63) and RotatE (159) tackle the same issue by following same billinear model paths. Among all the models that learn KGE with literals, TransEA (199) jointly learns entity relations and numeric attributes with a combination of attribute embedding and translation-based embedding model. Built on top of the existing latent feature approaches, LiteralE (68) stacks an extra simple module on the model architecture where literals can be incorporated directly into the embedding learning process with a parameterized function. In order to integrate

different latent, relational and numerical feature types, KBLRN (38) proposes to combine neural representation learning with probabilistic product of expert models.

4.2.1.6 Summary and Comparisons

To summarize, the five groups of KGE approaches have their own pros and cons. Translation-based models can preserve structural information well while learning the embeddings. However, as the complexity grows, the number of parameters and computational cost can grow substantially at the same time. Semantic matching models are fast and easy to be scaled towards the entire KG given their shallow structures. However, there is a trade off in between it and the capability of capturing more expressive features. CNN-based models can generally perform better than the above two categories in learning expressive features. However, most of the existing studies model each triplet individually, which causes severe latent information loss problems. Neighborhood information-based models have demonstrated their model superiority over cohorts in the past few years. However, most of the existing models are impeded from further performance improvement due to two reasons. First, many prior studies consider the inbound directional neighbors of a specific entity solely, and tend to underrate the contexts generated from the outbound directional neighbors. Second, most of these models solely apply the k-th hop output to learn multi-hop embed-

dings, which suffers from great early-stage information loss at graph attention steps. Models using relation-path or literals can exploit long-term dependency of entity relations spanning over all the relational paths, however, their feature expressiveness is still restrained.

4.2.2 Cross-lingual Entity Alignment (CEA)

Aside from KGE, another line of related work is to align entities across multiple input KGs of different languages. Traditional approaches to entity alignment are mainly human-centric at the cost of time, labor and flexibility (80, 95, 97, 172). However, as the size of KGs grows substantially over time, those traditional methods become more inapplicable and unable to scale.

Recently, researchers have developed many advanced deep learning-based models to tackle the CEA problem. Based on whether the models take in structural, neighborhood or extra information of the KGs, we can further classify the existing models into three categories, namely the embedding-based methods (Group A), the neighborhood information-based methods (Group B), and the methods using extra information beyond structures (Group C).

4.2.2.1 Embedding-based Methods

Embedding-based models tap on the embedding characteristic to view relations between entities as translations between heads and tails.

For the CEA tasks specifically, JE (48) and MTransE (22) are the pioneer models belong to this category. JE (48) learns a unified vector space for different KGs using KG structures solely. Based on TransE, MTransE (22) proposes to embed entities and relations of each individual KG into an embedding space first. Then, the model learns transitions and minimizes distances between seed alignment embeddings of different KGs to map them into cross-lingual counterparts. Extending from the MTransE model, IPTransE (235) applies iterative methods to add in new alignments as training data for exploiting unlabelled data to a deeper level. After that, it maps those entities and relations of different languages to unified vector space using same parameters.

Given that the limited availability of labelled data among KGs, more studies have tried to exploit unlabelled data using semi-supervised learning, co-training, and bootstrapping approaches. Semi-supervised entity alignment method (SEA)(117) proposes to bring both labeled and unlabeled entities information into analyses for the CEA task. BootEA (161) solves the labelled data limitation problem by improving KGE methods with limit-based loss and bootstrapping techniques. Similar to

IPTransE (235), BootEA also utilizes iterative methods to add in new alignments as training data.

Besides, AKE (82) proposes to use an adversarial knowledge embedding to jointly learn representations, alignment mappings and adversarial modules. MMEA (146) proposes to take a novel multiplicative approach to jointly model entities and relations from different KGs. JarKA (19) aims to use attributes to learn entity embeddings. And OTEA (119) aims to utilize an optimal transport-based method to optimize entity-level and group-level losses simultaneously.

4.2.2.2 Neighborhood Information-based Methods

Lately, more researchers and scholars aim to enrich embeddings by aggregating neighborhood information with graph-based methods. GM (212) proposes to represent entities and neighbors in topic entity graphs by viewing entity alignment as a graph matching task. This method leverages graph attention-based method to practise matching entities across two topic entity graphs. KECG (79) embeds entities from different KGs into a unified vector space with GAT (170) framework of shared parameters. Then, it applies TransE (15) architecture to implicitly complete different KGs in a joint manner.

MuGNN (17) aims to complete KGs by rule-based methods and to use multiple

channel graph attentions to reconcile the structural differences among input KGs. AliNet (163) aggregates multi-hop neighborhood information by gating and attention mechanisms. Specifically, it takes both heterogeneity and differences among counterpart entities neighbors into account, and expands the overlaps between entities neighborhood structures to attention-based neighbors in distance. As a result, the model derives final entity alignment by applying a gating mechanism to its distant and direct neighbors, and a relation loss function for improving entity representations. HGCN-JE (197) utilizes entity embeddings learnt by GCN (65) to approximate relation embeddings. Thus, better entities and relations embeddings are learnt in a joint manner.

MRAEA (96) learns relation-aware representations by using a relation-aware self-attention mechanism and adds in new alignments for training iteratively by adopting a bidirectional iterative training strategy. NAEA (236) gathers neighboring information at subgraph-level with an attention mechanism and proposes a NAEA neighborhood-aware attention representation method for learning effective neighborhood representations. RDGCN (196) takes advantage of the GCN (65) architecture to incorporate both relation interactions across KGs and dual relation counterparts in the computation. GM-EHD-JEA (211) takes an easy-to-hard decoding strategy to predict new alignments iteratively while addressing many-to-one problem using

a joint entity alignment algorithm. NMN (198) captures informative neighborhood features by the application of GCNs and a neighborhood sampling method. REA (118) proposes to deal with noise labels in CEA task with a reinforced training approach. SSP (112) captures both global and local feature by leveraging GCNs and translation-based model architectures. HyperKA (158) proposes to extend GCNs from the Euclidean space towards a hyperbolic space for the purpose of modelling KG hierarchical structures better. AttrGCN (92) proposes to learn attribute and relation triples in a unified framework with an attributed GNN. Other related work of this category include but are not limited to, (9, 104, 116, 138, 227).

4.2.2.3 Methods Using Extra Information Beyond Structures

The third category of methods are those designed to take in extra information, such as attributes, entity descriptions, etc., beyond structures. JAPE (160) aims to learn entity representations from both structural information and attribute information of different KGs in a joint manner. GCN-Align (188) leverages GCN (65) structure to learn entities' neighborhood information and cross-KG equivalent relationships between entities by incorporating entities and attributes information. HMAN (217) learns entity embeddings by employing multiple aspects of structures, relations and attributes. Moreover, they further enhances model performance by leveraging literal

descriptions of entities and fine-tuning pre-trained multilingual BERT (28). The best experimental results from their study are generated by an integration of HMAN to BERT. For the sake of fair comparisons, we only compare our proposed models to the results of HMAN model solely without stacking the BERT model. KDCoE (21) tackles the CEA problem by co-training two parallel embedding models, namely the cross-lingual KGE model and the cross-lingual description embedding model. JarKA (19) also proposes to take in attributes information to perform the CEA task.

4.2.2.4 Summary and Comparison

In summary, all three categories of CEA methods listed above have certain advantages and disadvantages. The embedding-based models enjoy their simple structures, while suffering from model expressiveness restraints due to the shallow structures. The neighborhood information-based models, which aggregate neighboring information of an entity, generally demonstrate promising results on the CEA task. However, most of the existing approaches only capture neighboring information from fixed graphs, which largely limit the capability of obtaining multi-hop neighborhood features for various individual relations. For the methods using extra information beyond structures, one of the biggest obstacles is that those extra information data might not always be available for learning in real world applications. Thus, no matter

Table 4.1: A summary of the CEA models involved in this study.

Groups	Methods	Embedding Model	Extra Information	Initialization
Group A	JE	TransE	No	Random
	MTransE	TransE	No	Random
	IPTransE	TransE	No	Random
	BootEA	TransE	No	Random
	MMEA	ComplEx	No	Random
	OTEA	TransE	No	Random
Group B	KECG	GNN+TransE	No	Random
	MuGNN	GNN	No	Random
	AliNet	GNN	No	Random
	HyperKA	GNN	No	Random
	NAEA	TransE	No	Random
	SSP	GNN+TransE	No	Random
	MRAEA	GNN	No	Random
	GM	GNN	No	Pre-trained
	RDGCN	GNN	No	Pre-trained
	HGCN-JE	GNN	No	Pre-trained
	NMN	GNN	No	Pre-trained
	GM-EHD-JEA	GNN	No	Pre-trained
	CAECGAT	GNN	No	Pre-trained
	AttrGCN	GNN	No	Pre-trained
Group C	JAPE	TransE	Attribute	Random
	GCN-Align	GNN	Attribute	Random
	JarKA	TransE	Attribute	Random
	HMAN	GNN	Attribute	Random
	KDCOE	TransE	Description	Random

how good the models are designed, the feasibility of using these methods is largely affected. Table 4.1 presents and summarizes the key methods belong to these three categories. On contrary to the methods described above, our proposed CAECGAT and DuGa-DIT are able to learn cross-KG embeddings and to bridge semantic gaps between KGs effectively by leveraging both neighborhood features and cross-KG alignment information.

4.3 Preliminaries

Mathematically, we define a **knowledge graph (KG)** as $G = (E, R, T)$ which comprises of E as a collection of entities, R as a collection of relations, and T as a collection of triples. Thus, we refer **Cross-Lingual KG Entity Alignment** as the task of deriving a set of high precision and high recall pairs of entities $(e_1, e_2) \in (E_1 \times E_2)$ from two KGs $G_1 = (E_1, R_1, T_1)$ and $G_2 = (E_2, R_2, T_2)$, in different languages. These two pairs thus can be considered as same real world entities. The list of notation list that are used in the following chapter is presented below.

4.4 Methodology

In this section, we aim to elaborate our proposed CAECGAT and DuGa-DIT model into details.

Table 4.2: Notation List.

Notation	Description
G_1, G_2	The knowledge graphs in different languages.
E_1, E_2	The entities in G_1 and G_2.
R_1, R_2	The relations in G_1 and G_2.
T_1, T_2	The triples in G_1 and G_2.
A	The seed alignments.
A_{ctx}, A_{obj}	The contextual and objective seed alignments.
$A_{ctx}^{new}, A_{obj}^{new}$	The newly added contextual and objective seed alignments.
$\mathbf{E}_1, \mathbf{E}_2$	The initial entity embeddings for G_1 and G_2.
$\bar{\mathbf{e}}_1, \bar{\mathbf{e}}_2$	The final entity vectors for $e_1 \in G_1$ and $e_2 \in G_2$ in CAECGAT.
$\mathbf{E}_1^L, \mathbf{E}_2^L$	The final entity embeddings for G_1 and G_2 in DuGa-DIT.
L	The number of dual KG attention layers.
$\mathbf{e}_1, \mathbf{e}_2$	The initial entity vectors for $e_1 \in G_1$ and $e_2 \in G_2$.
$\bar{\mathbf{e}}_1, \bar{\mathbf{e}}_2$	The final entity vectors for $e_1 \in G_1$ and $e_2 \in G_2$.
$\mathcal{N}_{e_i}$	A set of neighbors for entity e_i.
$\mathbf{M}_1, \mathbf{M}_2$	The intra-KG attention score matrices for G_1 and G_2.
$\mathbf{M}_{12}$	The cross-KG attention score matrices from G_1 to G_2.
$\mathbf{M}_{21}$	The cross-KG attention score matrices from G_2 to G_1.
$\mathbf{H}_1^l, \mathbf{H}_2^l$	The neighborhood features for G_1 and G_2 obtained by the l-th intra-KG attention layer.
$\mathbf{C}_1^l, \mathbf{C}_2^l$	The cross-KG alignment features for G_1 and G_2 in the l-th cross-KG attention layer.
$gate$	The gate mechanism to combine cross-KG embeddings.
$crossAggr$	The function to aggregate cross-KG information.
$crossAtt$	The function to propagate cross-KG information in different KGs.
$\mathcal{L}(\phi; A_{obj})$	The loss function of the proposed CAECGAT
DuGa-DIT model.	
ϕ	The trainable parameters in the model.
$D(\bar{\mathbf{e}}_1, \bar{\mathbf{e}}_2)$	The L_1 distance between entity vectors $\bar{\mathbf{e}}_1$ and $\bar{\mathbf{e}}_2$.

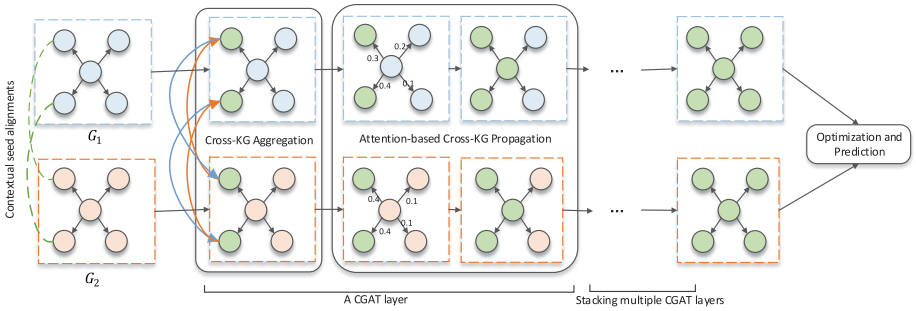


Figure 4.3: The architecture of the proposed CAECGAT model. The cross-KG aggregation layer is used to transfer cross-KG information across different KGs through contextual seed alignments. And the attention-based cross-KG propagation layer is used to propagate the cross-KG information in these two KGs. By stacking multiple CGAT layers, the cross-KG information can be propagated to multi-hop neighbors.

4.4.1 CAECGAT

In this subsection, we first give an overview of the CAECGAT system architecture. Then, we present a CGAT layer which contains a cross-KG aggregation layer and an attention-based cross-KG propagation layer. Lastly, we show the optimization and prediction training processes adopted in our method.

4.4.1.1 Model Overview

In Figure 4.3, we present the architecture of our proposed model, CAECGAT. The CAECGAT comprises of multiple CGAT layers where each of them contains a cross-

KG aggregation layer and an attention-based cross-KG propagation layer. Given a set of pre-aligned entity pairs from two cross-lingual KGs G_1 and G_2, our proposed model uses the set of pre-aligned entity pairs to transfer entity information across different KGs via the cross-KG aggregation layer. This operation helps to share cross-KG entity embeddings by mapping entity embeddings from different KGs to the same semantic space. After that, the CAECGAT gathers important cross-KG and neighborhood information via the attention-based cross-KG propagation layer. This operation allows cross-KG information to propagate across two different KGs. As a result, the entire architecture of CAECGAT can learn multi-hop cross-KG information by stacking the CGAT layers.

4.4.1.2 Cross-KG Aggregation

Zooming into the first layer of the CGAT, our model makes use of the seed alignments to transfer graph information across different KGs via the cross-KG aggregation layer. With the help of those pre-aligned entity pairs, we can predict new entity alignments effectively by fully leveraging pre-aligned neighbors as contextual information. Even though the cross-KG alignment information is effective, it has been largely neglected by many prior methods.

Formally, the two different KGs are represented as $G_1 = (E_1, R_1, T_1)$ and $G_2 =$

(E_2, R_2, T_2), and the collection of seed alignments are represented as $A = \{(e_1, e_2)|e_1 \in E_1, e_2 \in E_2\}$. In this section, we represent the entities from two KGs of different languages as k-dimensional embedding matrices $\mathbf{E}_1$ and $\mathbf{E}_2$. In the training process, we divide all the seed alignments into two groups, namely the contextual seed alignments A_{ctx} and the objective seed alignments A_{obj}. The contextual seed alignments in A_{ctx} aims to transfer information across different KGs. Then, the cross-KG information generated by A_{ctx} are applied as contextual alignment information to make predictions on matching scores for entity pairs in the objective seed alignments, A_{obj}.

For each pre-aligned entity pair $(e_1, e_2) \in A_{ctx}$, our model uses a gating mechanism to update the respective embeddings by aggregating their own embeddings and their counterpart entities of the other KG in different language:

$$\begin{aligned} \mathbf{h}_{e_1}^l &= gate(\mathbf{e}_1^l, \mathbf{e}_2^l) = g_1^l \cdot \mathbf{e}_1^l + (1 - g_1^l) \cdot \mathbf{e}_2^l \\ \mathbf{h}_{e_2}^l &= gate(\mathbf{e}_2^l, \mathbf{e}_1^l) = g_2^l \cdot \mathbf{e}_2^l + (1 - g_1^l) \cdot \mathbf{e}_1^l \end{aligned} \tag{4.1}$$

where $\mathbf{e}_1^l$ and $\mathbf{e}_2^l$ are the vectors of entity e_1 and e_2 in the l-th layer, g_1^l and g_2^l are the gate mechanism to control how much information flow across KGs, which is computed as:

$$\begin{aligned} g_1^l &= \sigma(\mathbf{W}_1^l[\mathbf{e}_1||\mathbf{e}_2] + \mathbf{b}_1^l) \\ g_2^l &= \sigma(\mathbf{W}_2^l[\mathbf{e}_1||\mathbf{e}_2] + \mathbf{b}_2^l) \end{aligned} \tag{4.2}$$

where σ is the sigmoid activation function which can constrain the output values

within the range of 0 to 1, $\mathbf{W}^l_{\{1,2\}}$ and $\mathbf{b}^l_{\{1,2\}}$ are the parameters, $||$ denotes the concatenation operation.

For those entities with no pre-aligned counterparts in the other KG, our model preserves their original embeddings in the cross-KG aggregation layer. Thus, this cross-KG aggregation method allows us to derive cross-KG embeddings $\mathbf{H}^l_1$ and $\mathbf{H}^l_2$, which share entity representations for equivalent entities from different KGs. Formally, we compute the cross-KG embeddings as follows:

$$\begin{aligned} \mathbf{H}^l_1 &= crossAggr(\mathbf{E}^l_1, \mathbf{E}^l_2, A_{ctx}) \\ \mathbf{H}^l_2 &= crossAggr(\mathbf{E}^l_2, \mathbf{E}^l_1, A_{ctx}) \end{aligned} \tag{4.3}$$

where $crossAggr$ denotes the cross-KG aggregation layer, which can be formulated as

$$crossAggr(\mathbf{e}^l_1, \mathbf{E}^l_2, A_{ctx}) = \begin{cases} gate(\mathbf{e}^l_1, \mathbf{e}^l_2), & if \quad (e_1, e_2) \in A_{ctx} \\ \mathbf{e}^l_1, & otherwise \end{cases} \tag{4.4}$$

and $crossAggr(\mathbf{e}^l_2, \mathbf{E}^l_1, A_{ctx})$ is computed in a similar form.

4.4.1.3 Attention-based Cross-KG Propagation

Recently, GNNs have been successfully applied for entity alignment. Those preliminary studies use GNNs to propagate neighborhood information in a monolingual KG but fail to take full use of seed alignments to alleviate the semantic gaps between

different KGs. In this study, we attempt to jointly learn the entity embeddings and to propagate cross-KG information in different KGs.

Zooming into the second layer of the CGAT, our model uses GNNs rather than information from single lingual KG to practice cross-KG information propagation. Inspired by GAT, CAECGAT aggregates neighboring features at the attention-based cross-KG propagation layer. This layer serves to choose the most significant shared neighbors across KGs of different languages to enhance the entity embeddings. Then, our model achieves to alleviate semantic gaps by emphasizing on shared neighbors with cross-KG information and eliminating noise neighbors.

Given the entity embeddings output of the cross-KG aggregation layer, e.g., $\mathbf{H}_1^l$ and $\mathbf{H}_2^l$ for two KGs G_1 and G_2, CAECGAT utlizes a graph attention mechanism to update entity embeddings by aggregating neighborhood information:

$$\begin{aligned} \mathbf{E}_1^{l+1} &= crossAtt(\mathbf{H}_1^l, G_1) \\ \mathbf{E}_2^{l+1} &= crossAtt(\mathbf{H}_2^l, G_2) \end{aligned} \tag{4.5}$$

where $crossAtt$ is the attention-based cross-KG propagation function. For each entity $e_1 \in G_1$, the output feature of $crossAtt$ function is generated by gathering the cross-

KG neighborhood embeddings using a weighted sum function:

$$\begin{aligned}\mathbf{e}_1^{l+1} &= crossAtt(crossAggr(\mathbf{E}_1^l, \mathbf{E}_2^l, A_{ctx}), e_1) \\ &= crossAtt(\mathbf{H}_1^l, e_1) \\ &= \text{Relu}(\mathbf{h}_{e_1}^l + \sum_{e_k \in \mathcal{N}_{e_1}} \alpha_{1k} \mathbf{h}_{e_k}^l)\end{aligned} \tag{4.6}$$

where Relu (40) is the activation function, $\mathcal{N}_{e_1}$ is a set of neighbors of entity e_1, and α_{1k} is the normalized attention weight for the neighborhood entity e_k, which is computed as:

$$\alpha_{1k} = \frac{\exp(s(\mathbf{h}_{e_1}^l, \mathbf{h}_{e_k}^l))}{\sum_{e'_k \in \mathcal{N}_{e_1}} \exp(s(\mathbf{h}_{e_1}^l, \mathbf{h}_{e'_k}^l))} \tag{4.7}$$

where s is the scoring function which is denoted as:

$$s(\mathbf{h}_{e_1}^l, \mathbf{h}_{e_k}^l) = \text{LeakyRelu}(\mathbf{v}^T[\mathbf{h}_{e_1}^l || \mathbf{h}_{e_k}^l]) \tag{4.8}$$

where $\mathbf{v}$ is the attention vector, LeakyRelu (93) is the activation function which is widely used in graph attention mechanism. The *crossAtt* function for the entities in KG G_2 is computed in a similar form.

Formally, we denote the update rule of the cross-KG aggregation and the cross-KG propagation layers in a CGAT layer as follows:

$$\begin{aligned}\mathbf{E}_1^{l+1} &= crossAtt(crossAggr(\mathbf{E}_1^l, \mathbf{E}_2^l, A_{ctx}), G_1) \\ \mathbf{E}_2^{l+1} &= crossAtt(crossAggr(\mathbf{E}_2^l, \mathbf{E}_1^l, A_{ctx}), G_2)\end{aligned} \tag{4.9}$$

From above, we derive new entity embeddings $\bar{\mathbf{E}}_1 = \mathbf{E}_1^L$ and $\bar{\mathbf{E}}_2 = \mathbf{E}_2^L$ from the CGAT layer which contains L cross-KG aggregation layers and L attention-based cross-KG propagation layers. These new entity embeddings include information from both cross-KG and multi-hop neighborhood.

4.4.1.4 Optimization and Prediction

In order to ensure the learnt embeddings from equivalent entities of different KGs are located close to each other, CAECGAT makes full use of seed alignments as contextual alignments in both training and prediction stages.

In the training stage, the objective seed alignments A_{obj} and margin-based ranking loss function are applied to model optimization, which is denoted as:

$$\begin{aligned}\mathcal{L}(\phi; A_{obj}) = &\sum_{(e_1,e_2)\in A_{obj}} [D(\bar{\mathbf{e}}_1, \bar{\mathbf{e}}_2) + \lambda - D(\bar{\mathbf{e}}_1, \bar{\mathbf{e}}_2^-)]_+ \\ &+ \sum_{(e_1,e_2)\in A_{obj}} [D(\bar{\mathbf{e}}_1, \bar{\mathbf{e}}_2) + \lambda - D(\bar{\mathbf{e}}_1^-, \bar{\mathbf{e}}_2)]_+\end{aligned} \tag{4.10}$$

where $[x]_+ = max\{0, x\}$, ϕ denotes the parameters in the model , (e_1, e_2^-) and (e_1^-, e_2) are the negative entity alignment pairs obtained by randomly replacing the positive entity e_1 or e_2, $\bar{\mathbf{e}}_1 \in \bar{\mathbf{E}}_1$ and $\bar{\mathbf{e}}_2 \in \bar{\mathbf{E}}_2$ are the vectors of e_1 and e_2, $D(\bar{\mathbf{e}}_1, \bar{\mathbf{e}}_2) = ||\bar{\mathbf{e}}_1 - \bar{\mathbf{e}}_2||_1$ denotes the L_1 distance function, $\lambda > 0$ is the margin hyper-parameter.

To give a clear picture of our CAECGAT, we present our model algorithm details in Algorithm 1. It is worth noting that the contextual seed alignments A_{ctx} and ob-

jective seed alignments A_{obj} are in dynamic manner during training. In fact, in order to ensure that all the seed alignments can be applied in loss function optimization, the contextual and objective seed alignment sets are set to be changed dynamically. All the seed alignments in A are divided into different batch seed alignments. For each optimization step, our model randomly selects one batch seed alignment for objective purpose A_{obj}, and the rest are naturally grouped as contextual seed alignments A_{ctx}. The operation is repeated until the maximum of training epochs is reached. Details of this process are illustrated in line 2 to 11 in the Algorithm 1.

In the prediction stage, given test entity e_1 (or e_2) of KG G_1 (or G_2), all the respective entities in another KG G_2 (or G_1) are ranked based on the L_1 distance. The L_1 distances are computed by entity vectors $\bar{\mathbf{E}}_1$ and $\bar{\mathbf{E}}_2$ generated from our CAECGAT model. In this stage, all the pre-aligned seed alignments in A can be used as contextual seed alignments, called $A_{ctx} = A$. This process is illustrated in line 12 to 13 in Algorithm 1.

4.4.2 DuGa-DIT

CAECGAT (203) presented above serves as a preliminary study of our DuGa-DIT model. CAECGAT designs the system so that it can jointly learn embeddings by using pre-aligned seeds to propagate information across KGs. However, we dis-

Algorithm 1 CAECGAT Model

Input: Given KGs G_1 and G_2, as well as the initial entity embeddings $\mathbf{E}_1$, $\mathbf{E}_2$ and pre-aligned seed alignments A.

Output: The final entity embeddings $\bar{\mathbf{E}}_1$, $\bar{\mathbf{E}}_2$ and parameters ϕ.

Let $\mathbf{E}_1^0 = \mathbf{E}_1$ and $\mathbf{E}_2^0 = \mathbf{E}_2$.
repeat
    Select a batch seed alignments as objective seed alignments A_{obj} and the rest are used as context seed alignments A_{ctx}.
    for $l = 1$ to L **do**
        $\mathbf{E}_1^{l+1} = crossAtt(crossAggr(\mathbf{E}_1^l, \mathbf{E}_2^l, A_{ctx}), G_1)$;
        $\mathbf{E}_2^{l+1} = crossAtt(crossAggr(\mathbf{E}_2^l, \mathbf{E}_1^l, A_{ctx}), G_2)$;
    end for
    Set $\bar{\mathbf{E}}_1 = \mathbf{E}_1^L$ and $\bar{\mathbf{E}}_2 = \mathbf{E}_2^L$
    Compute loss function $\mathcal{L}(\phi; A_{obj})$.
    Optimize parameters ϕ in CAECGAT model.
until reach the maximum number of training epochs
Set $A_{ctx} = A$.
Use the learned CAECGAT model to predict the final entity embeddings $\bar{\mathbf{E}}_1$ and $\bar{\mathbf{E}}_2$. =0

cover that information propagation for multi-hop neighbors can be important, so as making full use of the pre-aligned seed alignments in both training and prediction stage. Therefore, we aim to optimize our prior study by mainly two ways. First, we introduce Dual Gated Graph Attention Network with Dynamic Iterative Training (DuGa-DIT) to learn cross KGE and to bridge semantic gaps between KGs with more effective information propagation and dynamic iterative training. Second, we aim to run experiments on more benchmark datasets with various settings and scenarios to prove the effectiveness and robustness of our newly proposed model.

In this subsection, we first give an overview of the DuGa-DIT system architecture. Then, we present the two attention layers, namely the intra-KG attention and the cross-KG attention. We also introduce the gated feature update that helps to yield

new entity embeddings from the above two layers. Lastly, we show the optimization, prediction and dynamic iterative training processes adopted in our method.

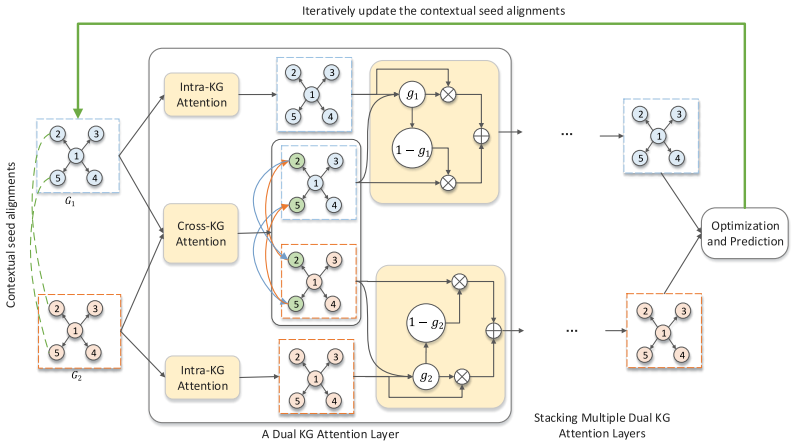


Figure 4.4: The architecture of the proposed Dual Gated Graph Attention Network with Dynamic Iterative Training (DuGa-DIT). The dual KG attention layer consists of a intra-KG attention layer and a cross-KG attention layer. The intra-KG attention layer is used to learn neighborhood features for each KG. And the cross-KG attention layer is employed to collect alignment information across two KGs. By stacking multiple dual KG attention layers, the local neighborhood and cross-KG alignment information can be propagated to multi-hop neighbors.

4.4.2.1 Model Overview

The Figure 4.4 demonstrates the system architecture of our proposed model, DuGa-DIT. The system comprises of multiple dual KG attention layers where each one has

a component of a cross-KG attention layer and an intra-KG attention layer. Given two input KGs G_1 and G_2, and a set of seed alignments across the KGs, the cross-KG attention layer serves to collect useful alignment information across KGs, while the intra-KG attention layer serves to collect those neighboring features. In this way, both intra-KG and cross-KG information are collected. And the multiple dual KG attention layers, as a whole, serve to multi-hop feature aggregation. After that, a dynamic iterative training process is applied to our DuGa-DIT to continuously update the cross-KG entity alignments. And we take advantage from this process to accumulate more cross-KG alignment features.

Given two KGs of different languages $G_1 = (E_1, R_1, T_1)$ and $G_2 = (E_2, R_2, T_2)$, and a collection of seed alignments $A = \{(e_1, e_2)|e_1 \in E_1, e_2 \in E_2\}$, we express the entities of the KGs as k-dimensional embedding matrices, $\mathbf{E}_1$ and $\mathbf{E}_2$. In the training process, we divide the seed alignments into two parts, similar to CAECGAT, namely the contextual seed alignments A_{ctx} and the objective seed alignments A_{obj}. The contextual seed alignments, A_{ctx}, serves to transfer information across two KGs. Therefore, the A_{ctx} provides cross-KG information which can be utilized as contextual alignment information. It further serves to make predictions on entity pairs' matching scores in those objective seed alignments in A_{obj}.

4.4.2.2 Intra-KG Attention

In DuGa-DIT, it designs an attention-based graph attention mechanism to generate features from each single lingual KG, named the intra-KG attention layer.

Given two KGs of different languages G_1 and G_2, $\mathbf{M}_1^l$ and $\mathbf{M}_2^l$ denote the normalized attention matrices respectively for G_1 and G_2. In order to update entity embeddings for every single KG, the model aggregates significant neighboring features through an intra KG attention layer, computed as:

$$\begin{aligned} \mathbf{H}_1^{l+1} &= \sigma(\mathbf{M}_1^l \mathbf{E}_1^l \mathbf{W}^l) \\ \mathbf{H}_2^{l+1} &= \sigma(\mathbf{M}_2^l \mathbf{E}_2^l \mathbf{W}^l) \end{aligned} \tag{4.11}$$

where $\mathbf{W}^l$ is the transformation parameters and σ is the activation function (e.g., Relu (40)), and l denotes the l-th layer.

Note that the normalized attention matrices are computed by GATs. The GATs compute scores for every single neighboring entity first, and normalize scores for all the neighboring entities after completed. Mathematically, each element in $\mathbf{M}_1^l$ is computed as:

$$\mathbf{M}_1^l[i,j] = \frac{\exp(s_{i,j}^l)}{\sum_{j'} \exp(s_{i,j'}^l)} \tag{4.12}$$

where $s_{i,j}^l$ is the attention score between the i-th and j-th entities, which is computed

as:

$$s_{i,j}^{l} = \begin{cases} \text{LeakyRelu}(\mathbf{v}^T[\mathbf{e}_i^l || \mathbf{e}_j^l]) & e_j \in \mathcal{N}_{e_i} \\ -\infty & otherwise \end{cases} \tag{4.13}$$

where LeakyRelu (93) is an activation function that is widely used in graph attention layer, $\mathbf{v}$ is the attention vector, $\mathcal{N}_{e_i}$ is a set of neighbors for entity e_i, $\mathbf{e}_i$ and $\mathbf{e}_j$ are the vectors for entity e_i and e_j, respectively. The normalized attention matrix $\mathbf{M}_2^l$ for G_2 can be computed in a similar manner.

4.4.2.3 Cross-KG Attention

The intra-KG attention layer above only serves to capture structural features locally from single language KG. This is not sufficient for our CEA task. Therefore, our DuGa-DIT stacks it with a novel cross-KG attention layer which utilizes the graph attention across two KGs to perform cross-KG alignment information aggregation. In this manner, our model can use the pre-aligned seed alignments fully and consider these pre-aligned seed alignments as edges to link up the two KGs of different languages.

Mathematically, given the contextual alignments A_{ctx}, our model utlizes cross-KG attention layer to construct the respective cross attention matrices from G_1 to G_2 and from G_2 to G_1. The $\mathbf{M}_{12}$ denotes the attention matrix from G_1 to G_2 and

every single element of $\mathbf{M}_{12}$ is computed as:

$$\mathbf{M}_{12}^{l}[i,j] = \frac{\exp(c_{i,j}^{l})}{\sum_{j'} \exp(c_{i,j'}^{l}) + \delta} \tag{4.14}$$

where $\delta = 1e^{-8}$ is a smoothing factor, $c_{i,j}^{l}$ is the cross-KG attention score between the i-th entity in G_1 and the j-th entity in G_2, which is caculated as:

$$c_{i,j}^{l} = \begin{cases} \text{LeakyRelu}(\mathbf{v}^T[\mathbf{e}_{1i}^{l} || \mathbf{e}_{2j}^{l}]) & (e_{1i}, e_{2j}) \in A_{ctx} \\ -\infty & otherwise \end{cases} \tag{4.15}$$

where e_{1i} is the i-th entity in G_1 and e_{2j} is the j-th entity in G_2, $\mathbf{e}_{1i}^{l}$ and $\mathbf{e}_{2j}^{l}$ is the vector representation of e_{1i} and e_{2j} in the l-th layer, respectively.

Based on above, cross-KG alignment features for G_1 are collected from the attention score matrix $\mathbf{M}_{12}$ after graph attention operation. Details of the operation are denoted as:

$$\mathbf{C}_{1}^{l+1} = \sigma(\mathbf{M}_{12}^{l}\mathbf{E}_{2}^{l}\mathbf{W}_{c}^{l}) \tag{4.16}$$

where $\mathbf{W}_{c}^{l}$ is the linear transformation parameters in cross-KG attention layer.

Similarly, the cross-KG alignment features for G_2 are computed as:

$$\mathbf{C}_{2}^{l+1} = \sigma(\mathbf{M}_{21}^{l}\mathbf{E}_{1}^{l}\mathbf{W}_{c}^{l}) \tag{4.17}$$

where $\mathbf{M}_{21}$ is the attention score matrix from KG G_2 to G_1, which is obtained in a similar way with $\mathbf{M}_{12}$.

With the cross-KG attention in our DuGa-DIT architecture, it allows the equivalent entities across different KGs to have shared cross-KG features, which can be propagated across KGs. This layer of operation largely enhances results of entity alignment task.

4.4.2.4 Gated Feature Update

Intuitively, semantic gaps can be largely reduced by utilizing the neighboring features, which are resourceful factors to model local structures of a mono-lingual KG, and the cross-KG features, which are resourceful factors to transfer alignment features across different KGs. Thus, our proposed DuGa-DIT integrates both features into a gated feature update mechanism which serves to update entity embeddings effectively. Mathematically, the updating rule of our method is formulated as:

$$\begin{aligned} \mathbf{E}_1^{l+1} &= \mathbf{H}_1^{l+1} \cdot g_1 + \mathbf{C}_1^{l+1} \cdot (1 - g_1) \\ \mathbf{E}_2^{l+1} &= \mathbf{H}_2^{l+1} \cdot g_2 + \mathbf{C}_2^{l+1} \cdot (1 - g_2) \end{aligned} \tag{4.18}$$

where $\cdot$ is the element-wise multiplication, g_1 and g_2 are the gate mechanism for G_1 and G_2, respectively, which are obtained by concatenating the neighborhood features and cross-KG alignment features, and followed by a non-linear transformation:

$$\begin{aligned} g_1 &= \theta(\mathbf{W}_g[\mathbf{H}_1^{l+1}||\mathbf{C}_1^{l+1}] + \mathbf{b}_g) \\ g_2 &= \theta(\mathbf{W}_g[\mathbf{H}_2^{l+1}||\mathbf{C}_2^{l+1}] + \mathbf{b}_g) \end{aligned} \tag{4.19}$$

where θ denotes the sigmoid function, $\mathbf{W}_g$ and $\mathbf{b}_g$ are trainable parameters.

In our work, L number of dual KG attention layers are stacked to generate both neighborhood and cross-alignment features propagation. As a result, DuGa-DIT can effectively generate new entity embeddings of $\mathbf{E}_1^L$ and $\mathbf{E}_2^L$.

4.4.2.5 Optimization and Prediction

In this work, DuGa-DIT aims to apply the contextual seed alignments as input and to apply the objective seed alignments as parameters optimizations. The loss function is defined as:

$$\begin{aligned}\mathcal{L}(\phi; A_{obj}) = &\sum_{(e_1,e_2)\in A_{obj}} [D(\bar{\mathbf{e}}_1, \bar{\mathbf{e}}_2) + \lambda - D(\bar{\mathbf{e}}_1, \bar{\mathbf{e}}_2^-)]_+ \\ &+ \sum_{(e_1,e_2)\in A_{obj}} [D(\bar{\mathbf{e}}_1, \bar{\mathbf{e}}_2) + \lambda - D(\bar{\mathbf{e}}_1^-, \bar{\mathbf{e}}_2)]_+\end{aligned} \tag{4.20}$$

where $[x]_+ = max\{0, x\}$, ϕ denotes the parameters in the DuGa-DIT model , (e_1, e_2^-) and (e_1^-, e_2) are the negative entity alignment pairs obtained by randomly replacing the positive entity e_1 or e_2, $\bar{\mathbf{e}}_1 \in \mathbf{E}_1^L$ and $\bar{\mathbf{e}}_2 \in \mathbf{E}_2^L$ are the vectors of e_1 and e_2, $D(\bar{\mathbf{e}}_1, \bar{\mathbf{e}}_2) = ||\bar{\mathbf{e}}_1 - \bar{\mathbf{e}}_2||_1$ denotes the L_1 distance function, $\lambda > 0$ is the margin hyper-parameter.

4.4.2.6 Dynamic Iterative Training

In real life scenarios, it is difficult to train alignment models due to the insufficiency of available seed alignments. This characteristic leads to sparsity problems of the cross-KG attention score matrix $\mathbf{M}_{12}$ and $\mathbf{M}_{21}$. It can lead to the fact of capturing less expressive cross-KG alignment features. Prior works (96, 161, 211, 235) have proposed to feed in new alignments predicted by training model iteratively. In this study, our proposed DuGa-DIT dynamically updates both contextual and objective seed alignments with iterative methods. One assumption is that if and only if e_i and e_j are the nearest entity to each other mutually, this entity pair (e_i, e_j) is defined as a new alignment. Aligning with prior work (96), DuGa-DIT implements the bidirectional iterative strategy to select the new alignment pairs. Significantly different from prior work, our proposed method applies the newly added alignment pairs to update both cross-KG attention score matrices and objective functions dynamically.

Mathematically, A_{ctx}^{new} denotes the newly added contextual seed alignments, and every attention score $c_{i,j}^{l}$ of the cross-KG attention score matrix $\mathbf{M}_{12}^{l}$ is updated by:

$$c_{i,j}^{l} = \begin{cases} \text{LeakyRelu}(\mathbf{v}^T[\mathbf{e}_{1i}^{l}||\mathbf{e}_{2j}^{l}]) & (e_{1i}, e_{2j}) \in A_{ctx} \bigcup A_{ctx}^{new} \\ -\infty & otherwise \end{cases} \tag{4.21}$$

Similar to above, the cross-KG attention matrix $\mathbf{M}_{21}^{l}$ can also be updated this way. As a result, our model is capable of generating more resourceful cross-KG

alignment information and adding new objective seed alignments A_{obj}^{new} to update the objective function, represented in $\mathcal{L}(\phi, A_{obj} \bigcup A_{obj}^{new})$.

To give a clear picture of our DuGa-DIT, we present our model algorithm details in Algorithm 2. The A^{new} is set as an empty set at the 1st iteration, shown in Line 1. In the training stage, DuGa-DIT dynamically splits seed alignments into two groups, A_{ctx} and A_{obj}. This categorization allows us to take full advantage of the seed alignments to train the model, shown in Line 5 to 6. Line 7 to 12 shows the process of updating entity embeddings with L layers of dual KG attention. Then, line 13 to 14 demonstrate how objective function computation and parameter optimizations are done with A_{obj} and A_{obj}^{new} in the DuGa-DIT model. After the training stage of DuGa-DIT is done, new seed alignments are predicted to update A^{new}. And the A^{new} is used in the next iterations.

In the prediction stage, as shown in Line 18, $A_{ctx} = A \bigcup A^{new}$ was set. Given a test entity e_1 (or e_2) from KG G_1 (or G_2), all entities in the respective KG G_2 (or G_1) are ranked based on the L_1 distance computed by entity vectors $\mathbf{E}_1^L$ and $\mathbf{E}_2^L$ calculation done by our DuGa-DIT model.

Algorithm 2 DuGa-DIT Model

Input: Given two KGs G_1 and G_2, the initial entity embeddings $\mathbf{E}_1$, $\mathbf{E}_2$ and pre-aligned seed alignments A.

Output: The final entity embeddings $\mathbf{E}_1^L$ and $\mathbf{E}_2^L$.

Initialize A^{new} as an empty set
repeat
  Let $\mathbf{E}_1^0 = \mathbf{E}_1$ and $\mathbf{E}_2^0 = \mathbf{E}_2$.
  repeat
    Randomly choose some seed alignments from A as A_{obj} and the rest as A_{ctx}.
    Randomly choose some seed alignments from A^{new} as A_{obj}^{new} and the rest as A_{ctx}^{new}.
    for $l = 1$ to L **do**
      Perform intra-KG attention and cross-KG attention and obtain H_1^l, H_2^l, C_1^l and C_2^l
      Update entity embeddings:
      $\mathbf{E}_1^l = \mathbf{H}_1^l \cdot g_1 + \mathbf{C}_1^l \cdot (1 - g_1)$;
      $\mathbf{E}_2^l = \mathbf{H}_2^l \cdot g_2 + \mathbf{C}_2^l \cdot (1 - g_2)$;
    end for
    Compute loss function $\mathcal{L}(\phi; A_{obj} \bigcup A_{obj}^{new})$.
    Optimize parameters ϕ in DuGa-DIT model.
  until reach the maximum number of training epochs
  Predict new seed alignments and update A^{new}
until reach the maximum number of iterations
Set $A_{ctx} = A \bigcup A^{new}$.
Use the learned DuGa-DIT model to predict the final entity embeddings $\mathbf{E}_1^L$ and $\mathbf{E}_2^L$.
=0

4.5 Experiment

In this section, we validate our proposed methods, CAECGAT and DuGa-DIT respectively, on widely used benchmark datasets for the CEA task. Then we elaborate on the implementation details and model comparisons. Lastly, we present our experimental findings, and make further detailed analyses.

4.5.1 CAECGAT

In this subsection, we present our datasets, experimental settings and experimental results for the CAECGAT model first.

4.5.1.1 Datasets

We conduct extensive experiments to test the effectiveness of our proposed CAECGAT on three widely used cross-lingual datasets of DBP15K (160), including DBP15K$_{ZH-EN}$ (Chinese-English), DBP15K$_{JA-EN}$ (Japanese-English) and DBP15K$_{FR-EN}$ (French-English). These datasets are extracted from multilingual DBpedia (78), including four KGs of different languages (English, Chinese, Japanese and French). Each dataset contains 15,000 inter-lingual links connecting equivalent entity pairs in KGs with different languages. The statistics of the datasets are summarized in the table below.

4.5.1.2 Implementation Details

Following the prior work (197, 212) which have achieved promising results, we initialize the entity embeddings of our proposed method with the sum of word vectors of entities' literal or surface format. Beyond that, we also share the same 30% for training and 70% for testing data split with the prior works (197, 212). We further

Table 4.3: Statistics of the datasets.

Dataset		Entities	Relations	Rel.Triples
DBP15K_{ZH-EN}	Chinese	66,469	2,830	153,929
	English	98,125	2,317	237,674
DBP15K_{JA-EN}	Japanese	65,744	2,043	164,373
	English	95,680	2,096	233,319
DBP15K_{FR-EN}	French	66,858	1,379	192,191
	English	105,889	2,209	278,590

split the training set into 10% and 90%, where the 10% serves as the development set for selecting parameters and the 90% serves for training purposes. Adam (64) is applied to find model parameters optimizations. And the model was trained up to $5,000$ epochs.

We utilize grid search to select hyper-parameters in this study, which were derived from the Hits@1 results on development set. The number of negative pair of entities are set as $\{10, 20, 30, 50\}$; the margin parameter λ in Equation 4.20 is set as $\{1, 3, 5\}$; the number of CGAT layers is set as $\{1, 2, 3\}$, the dropout rate is set as $\{0.1, 0.2, 0.3\}$, and the batch size is set as $\{500, 1000, 1500, 2000, 3000\}$, the learning rate is set as $\{0.002, 0.001, 0005\}$, respectively. In the end, for each positive pair of entities, we randomly select a sample of 30 negative pairs, and the margin parameter λ in

Equation 4.20 is selected to be $\lambda = 3$.

As described in methodology, our CAECGAT stacks two CGAT layers multi-hop cross-KG information propagation. The learning rate is set as 0.002. The dropout rate is set as 0.2. The batch size is set as $2,000$. In order to evaluate the performance of CAECGAT, the standard metrics of Hits@1 (H@1), Hits@10 (H@10) and MRR (Mean Reciprocal Rank) are applied in this study. And all experiments reported in this study are conducted on NVIDIA GTX 1080Ti GPUs and the codes are implemented with TensorFlow. Details are available online at https://github.com/sherrybbalonsozhu.

4.5.1.3 Models for Comparison

To investigate the efficiency and robustness of our proposed model, CAECGAT, we compare it with mainly two fields of existing baseline models, namely the embedding-based models and the GNN-based models.

The embedding-based models involved in our study as baselines are JE (48), JAPE (160), BootEA (161), MTransE (22), and IPTransE (235). And following the line of GNN-based work, we include the following GNN-models as baselines for comparison: GCN-Align (188), KECG (79), MuGNN (17), AliNet (163), HGCN-JE (197), HMAN (217), MRAEA (96), NAEA (236), RDGCN (196) and GM-EHD-JEA

Table 4.4: Experimental results on entity alignment. The results of the baselines are directly taken from the original papers. The best results are in **bold**, and the second best results are in underline.

Methods	DBP15K_{ZH-EN}			DBP15K_{JA-EN}			DBP15K_{FR-EN}		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
JE (48)	21.27	42.77	-	18.92	39.97	-	15.38	38.84	-
MTransE (22)	30.83	61.41	0.364	27.86	57.45	0.349	24.41	55.55	0.335
JAPE (160)	41.18	74.46	0.490	36.25	68.5	0.476	32.39	66.68	0.430
IPTransE (235)	40.60	73.50	0.516	36.70	69.30	0.474	33.30	68.50	0.451
BootEA (161)	62.94	84.75	0.703	62.23	85.39	0.701	65.30	87.44	0.731
GCN-Align (188)	41.25	74.38	0.549	39.91	74.46	0.546	37.29	74.49	0.532
GM (212)	67.93	78.48	-	73.97	87.15	-	89.38	95.24	-
KECG (79)	47.77	83.5	0.598	48.97	84.4	0.61	48.64	85.06	0.61
MuGNN (17)	49.40	84.40	0.611	50.10	85.70	0.621	49.50	87.00	0.621
AliNet (163)	53.90	82.60	0.628	54.90	83.10	0.654	55.20	85.20	0.657
RDGCN (196)	70.75	84.55	-	76.74	89.54	-	88.64	95.72	-
HGCN-JE (197)	72.03	85.7	-	76.62	89.73	-	89.16	96.11	-
HMAN (217)	56.20	85.10	-	56.70	86.90	-	54.00	87.10	-
NAEA (236)	65.01	86.73	0.720	64.14	87.27	0.718	67.32	89.43	0.752
MRAEA (96)	**75.70**	92.98	**0.827**	75.78	93.38	0.826	78.04	94.81	0.849
GM-EHD-JEA (211)	73.58	-	-	79.15	-	-	92.43	-	-
CAECGAT (ours)	75.55	**93.38**	0.818	**83.58**	**95.59**	**0.881**	**94.68**	**99.18**	**0.965**

(211).

4.5.1.4 Main Results

The experimental results of CAECGAT and baseline models on cross lingual entity alignment are presented in Table 4.4. These results listed demonstrate that the effectiveness and robustness of CAECGAT is superior than baseline models. As illustrated in Table 4.4, our proposed CAECGAT gains the best results across all

metrics on both DBP15K$_{JA-EN}$ and DBP15K$_{FR-EN}$ dataset. Meanwhile, it scores the best for the H@10 result and the second best for the H@1 and MRR results on DBP15K$_{ZH-EN}$. While in comparison with other GNN-based models, especially the GCN-Align (188), our proposed method achieved significantly better results and improved the H@1 score on DBP15K$_{ZH-EN}$ by 34.3%, the H@1 score on DBP15K$_{JA-EN}$ by 43.67%, and the H@1 score on DBP15K$_{FR-EN}$ by 57.39%. These remarkable improvements can be ascribed to our model design. It takes seed alignments as contextual information, allowing cross-KG information to propagate across KGs in different languages. Generally, our experimental results strengthen the necessity of involving contextual seed alignments into the CEA task. And our CAECGAT is a successful showcase model that utilizes contextual seed alignment and GAT approach to bridge language gaps between cross-lingual KGs.

4.5.1.5 Ablation Study

As shown in Table 4.5, we conduct ablation study on CAECGAT model to investigate the effect of each model component on the final model performance. In general, we have three variants that compose the overall architecture of CAECGAT. These are "BASELINE", "GAT" and "CrossGCN". The "BASELINE" is a simple model which utilizes the learnt embeddings from the sum of word vectors to identify equivalent

counterparts across KGs of different languages. The"GAT" is the conventional GAT network which stands for our proposed model without the cross-KG aggregation layer. The "CrossGCN" stands for the model which uses multiple GCN layers to replace the attention-based propagation layer. Lastly, models with different numbers of CGAT layers are represented as subscripts of $L = 1, 2, 3$.

From the results shown in Table 4.5, our observation is that, by taking surface names into consideration, the BASELINE model can achieve good results. This has already been proven by existing related works (211, 212). From the GAT and BASELINE results, we can see that performance improves further by adding GNN layers. While comparing GAT and CAECGAT$_{L=2}$, we can easily draw conclusion on the importance of cross-KG aggregation component of our proposed model. It has helped to achieve great model performance improvement over GATs on all datasets. Specifically, it has generated 11.69% performance improvement on H@1 DBP15K$_{ZH-EN}$, 11.23% performance improvement on H@1 DBP15K$_{JA-EN}$, and 7.54% performance improvement on H@1 DBP15K$_{FR-EN}$.

In comparison with CrossGCN, our proposed CAECGAT$_{L=2}$ also outperforms the CrossGCN line of work significantly. The attention mechanism adopted in cross-KG propagation layer is capable of selecting neighbors with more important cross-KG information. Thus, it helps to propagate cross-KG information better and to shrink

Table 4.5: Ablation Study.

Methods	DBP15K$_{ZH-EN}$			DBP15K$_{JA-EN}$			DBP15K$_{FR-EN}$		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
BASELINE	56.09	71.55	0.614	67.1	78.25	0.712	85.48	91.92	0.878
GAT	63.86	91.92	0.709	72.35	94.64	0.815	86.24	98.70	0.895
CrossGCN	65.45	92.85	0.757	72.07	94.9	0.809	93.79	98.98	0.958
CAECGAT$_{L=1}$	72.25	92.03	0.796	81.9	93.68	0.862	93.55	98.5	0.954
CAECGAT$_{L=2}$	**75.55**	**93.38**	**0.818**	**83.58**	95.59	**0.881**	**94.68**	**99.18**	**0.965**
CAECGAT$_{L=3}$	74.95	**93.38**	**0.818**	82.46	**96.21**	0.877	93.78	98.77	0.956

the semantic gap between KGs of different languages. The ablation study results clearly demonstrate the importance of each component involved in the CAECGAT model.

In terms of different layers used in CAECGAT models, we can see that CAECGAT$_{L=1}$ which captures one-hop neighbors performs the worst. Both the two CGAT layered, CAECGAT$_{L=2}$, and the three CGAT layered, CAECGAT$_{L=3}$, are able to propagate multiple-hop neighbors perform better. It is worth noting that performance stop further improving by adding one more layer from CAECGAT$_{L=2}$ to CAECGAT$_{L=3}$. As stacking more layers also increases number of parameters involved in the modeling process, thus, we choose CAECGAT$_{L=2}$ as the optimal number of CGAT layers.

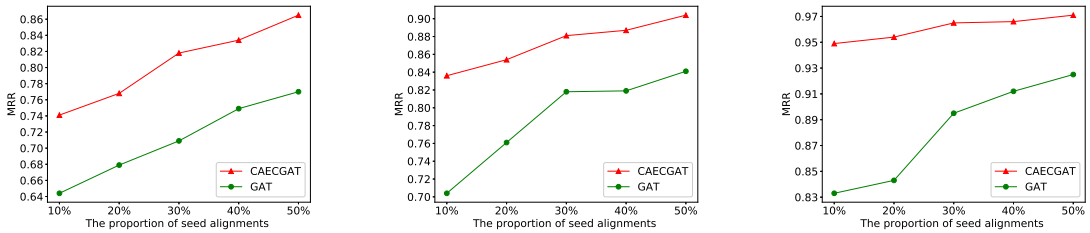
4.5.1.6 Performance and Different seed alignments sizes

Aside from the ablation study, we also conduct investigation on the effects of proportions of pre-aligned seed alignments on model performance. Specifically, we evaluate CAECGAT's performance when the proportions of training sets were set to 10% to 50%, with each step of change being 10%. Therefore, all the remaining entity alignments are used for testing (e.g., from 10% for training and 90% for testing to 50% for training and 50% for testing).

As illustrated in Figure 4.5, we make comparisons between CAECGAT with baseline GAT model without the cross-KG aggregation layer. Generally, the model performance improves gradually as the proportions of training sets increase on all datasets. Our proposed CAECGAT has demonstrated its effectiveness by consistently outperforming GAT models with large gains in all proportions. It reassures us the CEACGAT powerfulness on capturing cross-KG features and reducing the semantic gaps between KGs of different languages.

4.5.2 DuGa-DIT

Built on top of the preliminary study above, in this subsection, we present our datasets, experimental settings and experimental results in details for the enhanced DuGa-DIT model.



(a) MRR on DBP15K$_{ZH-EN}$ (b) MRR on DBP15K$_{JA-EN}$ (c) MRR on DBP15K$_{FR-EN}$

Figure 4.5: The MRR results on DBP15K using different proportions of seed alignments.

4.5.2.1 Datasets

To evaluate the effectiveness of our proposed model, we apply two benchmark datasets DBP15K (160) and WK31-60K (21, 22) with rich nodes and edges of different relations to the experiments. Details of these two datasets are presented in Table 4.6. Specifically, the DBP15K dataset comprises of four KGs of different languages, namely the English, Chinese, Japanese and French. These nodes and relations are extracted from DBpedia (78). Given these four KGs, cross-lingual subsets are built as DBP15K$_{ZH-EN}$ (Chinese-English), DBP15K$_{JA-EN}$ (Japanese-English) and DBP15K$_{FR-EN}$ (French-English). In total, 15,000 cross-lingual links are connected between two KGs.

The WK31-60K dataset comprises of three KGs of different languages from DBpedia, namely the English, French, and German. Every single KG consists of

Table 4.6: Statistics of the datasets.

Dataset		Entities	Relations	Rel.Triples	Training/Dev/Test
DBP15K$_{ZH-EN}$	Chinese	66,469	2,830	153,929	4,050/450/10,500
	English	98,125	2,317	237,674	
DBP15K$_{JA-EN}$	Japanese	65,744	2,043	164,373	4,050/450/10,500
	English	95,680	2,096	233,319	
DBP15K$_{FR-EN}$	French	66,858	1,379	192,191	4,050/450/10,500
	English	105,889	2,209	278,590	
WK31-60K$_{FR-EN}$	French	45,255	277	258,337	14,095/1,566/36,544
	English	64,539	458	569,393	
WK31-60K$_{FR-DE}$	German	43,503	172	244,647	14,451/1605/37,467
	English	64,539	458	569,393	

around 60K entities, and two cross-lingual entity alignment subsets are built as WK-60K$_{FR-EN}$ (French-English) and WK-10k$_{FR-DE}$ (German-English).

4.5.2.2 Implementation Details

Hyper-parameters Following the prior work of (160, 217), we share the same 30% for training and 70% for testing data split with the prior works (160, 217). We further split the training set into 10% and 90%, where the 10% serves as the de-

velopment set for selecting parameters and the 90% serves for training purposes. Table 4.6 is an illustration of the statistics of our data split. Adam (64) is applied to find model parameters optimizations. And the model was trained up to $3,000$ epochs. Same as CAECGAT, we utilize grid search to select hyper-parameters in this study, which were derived from the Hits@1 results on development set. The number of negative pair of entities are set as $\{10, 20, 30, 40, 50\}$; the margin parameter λ in Equation 4.20 is set as $\{1, 3, 5\}$; the number of dual KG attention layers are set as $\{1, 2, 3\}$, the dropout rate is set as $\{0.1, 0.2, 0.3\}$, and the batch size is set as $500, 1000, 1500, 2000, 3000,$ the learning rate is set as $\{0.004, 0.002, 0.001, 0005\}$, respectively. In the end, for each positive pair of entities, we randomly select a sample of 30 and 50 negative pairs for DBP15K and WK31-60 respectively. The margin parameter λ in Equation 4.20 is selected to be $\lambda = 3$.

As described in methodology, our DuGa-DIT model stacks dual KG attention layers for information propagation. The learning rate is set as 0.002 and 0.004 for DBP15K and WK31-60K respectively. The dropout rate is set as 0.2. The batch size is set as $2,000$ and $3,000$ for DBP15K and WK31-60K respectively. We select 2 as the maximum iteration number for dynamic iterative training. For the best performed hyper-parameters of our proposed model, please refer to Table 4.7.

Initialization The random initialization and the initialization with pre-trained

Table 4.7: Hyper-parameter setting. "#neg" denotes the number of negative entity pairs. "lr" denotes the learning rate. "#iter" denotes the number of dynamic iterative training process.

Datasets	λ	L	#neg	dropout	lr	batch_size	#iter
DBP15K$_{ZH-EN}$	3	2	30	0.2	0.002	2,000	2
DBP15K$_{JA-EN}$	3	2	30	0.2	0.002	2,000	2
DBP15K$_{FR-EN}$	3	2	30	0.2	0.002	2,000	2
WK31-60K$_{FR-EN}$	3	2	50	0.2	0.004	3,000	2
WK31-60K$_{DE-EN}$	3	2	50	0.2	0.004	3,000	2

entity name word vectors are the two most commonly used methods for the entity alignment model initialization. In this study, we use both ways to perform the entity embedding initialization. The random initialization aims to initialize entity embeddings randomly without taking entity names into consideration. However, entity names usually contain valuable information for our CEA task, especially when the number of entity neighbors available are limited. As a result, most methods proposed later on include entity names in the initialization process. Specifically, they translate the target entity name into English, before calculating the sum of entity name word vectors (196, 197, 198, 212). In the DuGa-DIT scenario, when our specific words in the entity belong to out-of-vocabulary category in the pre-trained

embedding, our model initializes these words of entities with random vectors.

Evaluation In this study, we apply two major standard metrics, Hits@N (H@N) and MRR, to measure our DuGa-DIT model performance. The higher these two metrics are, the better the model performance is. Hits@N (H@N) is an evaluation on correctly aligned proportion of entities ranking in the top N list. And the MRR stands for the mean reciprocal rank of the entity alignments results. In order to make comparisons with prior works, our study reports the Hits@1 (H@1), Hits@10 (H@10) and MRR results on DBP15K dataset, and reports an additional metric Hits@5 (H@5) on WK31-60K dataset.

Implementation

We conduct all experiments of this study on NVIDIA GTX 1080Ti GPUs. All our source codes are implemented on TensorFlow [3]. For the purpose of facilitating the experimental results reproduction, all datasets and source codes are available online at https://github.com/sherrybbalonsozhu.

4.5.2.3 Models for Comparison

To investigate the efficiency and robustness of our proposed model, DuGa-DIT, we compare it with mainly three fields of existing baseline models, namely the

[3]https://tensorflow.org

embedding-based models (Group A), the neighborhood information-based methods (Group B) and the methods using extra information beyond structures (Group C).

Embedding-based Methods (Group A) The embedding-based models refer to those methods that aim to learn low-dimensional vectors of entities from different KGs and to match them by computing entity vectors similarity scores. The list of models for comparison include, JE (48), MTransE (22), MMEA (146) and OTEA (119). For methods belong this category but leverage bootstrapping methods to exploit the unlabelled data, we include BootEA (161) and IPTransE (235) for baseline comparisons. Other embedding-based methods involved in this study include MMEA (146) and OTEA (119).

Neighborhood Information-based Methods (Group B) Experimental results from prior work in this category have shown promising results in feature learning. The list of models involved in this study for comparison include, KECG (79), MuGNN (17), AliNet (163), HyperKA (158), NAEA (236), SSP (112), GM (212), GM-EHD-JEA (211), RDGCN (196), HGCN-JE (197), NMN (198), MRAEA (96), AttrGCN (92), and CAECGAT (203).

Methods using Extra Information Beyond Structures (Group C) The Group C involves methods which use extra information beyond structures to complete the task. The list of models for comparisons include, JAPE (160), GCN-Align (188),

JarKA (19) and HMAN (217), and KDCoE (21).

All methods mentioned above are included for comparison except AKE (82) and REA (118), because these two models were run on different datasets and experimental setups.

4.5.2.4 Main Results

The Table 4.8 illustrates our experimental results on DBP15K. We split our main results analyses on model comparisons by different groups.

Group A. Group A includes various embedding-based methods for comparisons. Among the five models in this category, MMEA achieves the best results on both H@1 and H@10 across the DBP15K datasets of different languages. The superior performance comes from its architecture that defines multiplication-based scoring functions. The functions are capable of dealing 1-N, N-1 and N-N relations. This flexible characteristic allows it to perform better than other baseline models in this category. It is worth noting that IPTransE and BootEA models also achieve relatively promising results. Both of these two approaches adopt bootstrapping strategy to feed in all possible entity pairs to train iteratively.

Table 4.8: Entity alignment results on DBP15K dataset. The results of the baselines are directly taken from the original papers. ‡ denotes the models using pre-trained word vectors as initialization method. DuGa-DIT (rand) is the model using the random initialization. The best results are in **bold**, and the second best results are in underline.

		DBP15K_{ZH-EN}			DBP15K_{JA-EN}			DBP15K_{FR-EN}		
Groups	Methods	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
Group A	JE	21.27	42.77	-	18.92	39.97	-	15.38	38.84	-
	MTransE	30.83	61.41	0.364	27.86	57.45	0.349	24.41	55.55	0.335
	IPTransE	40.60	73.50	0.516	36.70	69.30	0.474	33.30	68.50	0.451
	BootEA	62.94	84.75	0.703	62.23	85.39	0.701	65.30	87.44	0.731
	MMEA	68.07	86.74	-	65.53	85.90	-	67.70	89.01	-
Group B	KECG	47.77	83.50	0.598	48.97	84.40	0.610	48.64	85.06	0.610
	MuGNN	49.40	84.40	0.611	50.10	85.70	0.621	49.50	87.00	0.621
	AliNet	53.90	82.60	0.628	54.90	83.10	0.654	55.20	85.20	0.657
	HyperKA	57.20	86.50	0.678	56.40	86.50	0.673	59.70	89.10	0.704
	NAEA	65.01	86.73	0.720	64.14	87.27	0.718	67.32	89.43	0.752
	SSP	73.90	92.50	0.808	72.10	93.50	0.800	73.90	94.70	0.818
	MRAEA	75.70	92.98	0.827	75.78	93.38	0.826	78.04	94.81	0.849
	DuGa-DIT (rand)	75.62	85.78	0.805	78.11	91.36	0.829	83.73	91.97	0.867
	GM‡	67.93	78.48	-	73.97	87.15	-	89.38	95.24	-
	GM-EHD-JEA‡	73.58	-	-	79.15	-	-	92.43	-	-
	RDGCN‡	70.75	84.55	-	76.74	89.54	-	88.64	95.72	-
	HGCN-JE‡	72.03	85.7	-	76.62	89.73	-	89.16	96.11	-
	NMN‡	73.30	86.90	-	78.50	91.20	-	90.20	96.70	-
	CAECGAT‡	75.55	**93.38**	0.818	83.58	**95.59**	0.881	94.68	99.18	0.965
	AttrGCN‡	79.60	92.93	**0.845**	78.33	92.08	0.834	91.85	97.77	0.910
	DuGa-DIT ‡	**80.73**	88.17	0.832	**91.44**	95.21	**0.928**	**98.17**	**99.22**	**0.985**
Group C	JAPE	41.18	74.46	0.490	36.25	68.5	0.476	32.39	66.68	0.430
	GCN-Align	41.25	74.38	0.549	39.91	74.46	0.546	37.29	74.49	0.532
	HMAN	56.20	85.10	-	56.70	86.90	-	54.00	87.10	-
	JarKA	70.58	87.81	0.766	64.58	85.50	0.708	70.41	88.81	0.768

Group B. Group B includes various methods from mainly two different settings, namely the random initialization and the pre-trained word vectors as initialization. Based on the results presented in Table 4.8, GNN-based models, such as SSP, MRAEA, GM-EHD-JEA, HGCN-JE, CAECGAT etc., achieve better results as they can make full use of the neighborhood features effectively. Those methods, which adopt bootstrapping strategies to train their models iteratively, can further improve model performances, such as CAECGAT and DuGa-DIT. Besides, under the random initialization setting, we observe that the DuGa-DIT (rand) we proposed achieves comparable results as the best reported MRAEA. For methods using pre-trained word vectors as initialization, our proposed model, DuGa-DIT, scores the highest for H@1 on DBP15K$_{ZH-EN}$, the highest for H@1 and MRR on DBP15K$_{JA-EN}$, and the highest for H@1, H@10 and MRR on DBP15K$_{FR-EN}$. Among the two settings, most existing approaches use pre-trained word vectors as initialization which generally performs better than random initialization.

Group C. Group C involves a group of methods utilizing extra information, including attribute and entity description. The JAPE, JarKA and KDCoE take the embedding-based approach to encode the KG structure. And the GCN-Align and HMAN adopt GNN-based approach. Surprisingly, our experimental results have

shown that all approaches belong to this category do not have significant performance gains. It can be derived that those extra information could contain a lot of noise, which hurt model performances to some extent.

Comparison Across Groups. Based on the experimental results presented in Table 4.8, we derive to three findings as follows: (a) Comparing Group A to Group B, we believe neighborhood information-based methods, which take advantage of the resourceful neighborhood features, achieve generally better performance than naive embedding-based methods; (b) Comparing Group B to Group C, we believe that the literal or surface names of entities are more significant than extra attribute information; (c) Comparing BootEA vs. MTransE from Group A to GM-EHD-JEA vs. GM from Group B, we believe that models adopt iterative approach can generate greater performance gains than those do not adopt iterative approach.

Analysis of our DuGa-DIT model. Given that our DuGa-DIT aggregates neighboring features using a GAT approach, our proposed method should belong to Group B. For the purpose of fair comparisons, we include DuGa-DIT with random initialization, DuGa-DIT (rand), as one setting reported in the end results. Compared to other existing methods with random initialization settings (e.g.SSP and

MRAEA), the model performance of DuGa-DIT (rand) is comparable across all the datasets. Whereas our DuGa-DIT with pre-trained word vectors as initialization achieves significantly better results than SOTA methods across all the datasets, except the H@10 and MRR on DBP15K$_{ZH-EN}$ and the H@10 on DBP15K$_{JA-EN}$. As shown in the Table 4.8, DuGa-DIT outperforms our preliminary study, CAECGAT, by 7.68% on H@1 for DBP15K$_{JA-EN}$ and achieves great improvement of 3.49% on H@1 for DBP15K$_{FR-EN}$. The results of the statistical significance test, t-test, also demonstrate the superiority of DuGa-DIT over preliminary study CAECGAT and DuGa-DIT (rand) on H@1 with confidence of $p < 0.5$.

As a result, our proposed DuGa-DIT model is capable of capturing neighborhood features from intra-KG and cross-KG alignment information using cross-KG attention. By incorporating the dynamic iterative training into the architecture, DuGa-DIT dynamically updates the cross-KG attention score matrices effectively and provides more cross-KG alignment information. These characteristics of our model allow it to obtain better entity representations for the CEA task.

Impact of the Initialization. Prior works (196, 197, 198, 203, 212) have proved the literal or surface forms of entities as resourceful information. Therefore, many existing models, including GM, GM-EHD-JEA, RDGCN, HGCN-JE, NMN, AttrGCN

and CAECGAT, initialize entity embeddings with word vectors of entities' surface forms. Aligning with these prior work, DuGa-DIT also takes the same approach. DuGa-DIT (rand) and DuGa-DIT aim to investigate the impact of various initialization settings. Based on the results from Table 4.8, such as HGCN-JE vs. AliNet and DuGA-DIT vs. DuGa-DIT (rand), we can tell that models without considering entities names achieve less promising results than those involved entities names. As equivalent entities from different languages always contain shared words in their translated surface forms, those surface forms can be resourceful information for entity pairs alignment. It is worth noting that our DuGa-DIT also outperforms those methods adopting entity names, such as CAECGAT vs. DuGa-DIT and AttrGCN vs. DuGa-DIT on H@1 metric.

As presented in Table 4.9, we also run experiments on WK31-60K, which is larger in size and more challenging in task dataset than the DBP15K. For models which do not contain results on WK31-60K, such as RDGCN, HGCN-JE, NMN, SSP and CAECGAT, we ran source codes to obtain the results. Table 4.9 demonstrates the superiority of our proposed DuGa-DIT over the SOTA models. Compared to our preliminary study CAECGAT, DuGa-DIT achieves a significant performance improvement of 6.45% on H@1 for WK31-60K$_{FR-EN}$ and a great improvement of

Table 4.9: Entity alignment results on WK31-60K dataset. ‡ denotes the models using pre-trained word vectors as initialization methods. DuGa-DIT (rand) is the model using the random initialization.

		WK31-60K$_{FR-EN}$				WK31-60K$_{DE-EN}$			
Groups	Methods	H@1	H@5	H@10	MRR	H@1	H@5	H@10	MRR
Group A	MTransE	13.95	20.25	-	0.177	3.37	10.07	-	0.072
	JAPE	16.85	35.41	-	0.271	14.71	23.86	-	0.192
	BoostEA	33.31	51.14	-	0.425	23.28	39.29	-	0.316
	OTEA	36.07	54.08	-	0.447	26.97	43.97	-	0.352
Group B	GCN-Align	21.47	37.81	-	0.293	13.8	24.55	-	0.190
	MRAEA	54.34	76.03	-	0.646	42.76	62.77	-	0.524
	SSP	62.67	76.64	81.11	0.693	60.55	77.01	80.46	0.684
	DuGa-DIT (rand)	66.12	78.32	80.84	0.718	59.67	75.12	77.64	0.669
	RDGCN‡	77.91	91.72	93.31	0.843	70.87	85.37	86.74	0.776
	HGCN-JE‡	76.59	90.26	93.85	0.834	71.91	85.67	86.71	0.781
	NMN‡	77.63	91.25	92.66	0.841	70.91	85.59	86.75	0.778
	CAECGAT‡	78.24	92.23	95.23	0.852	71.41	**87.20**	**89.26**	0.788
	DuGa-DIT‡	**84.69**	**95.06**	**95.47**	**0.897**	**73.67**	85.99	86.84	**0.796**
Group C	KDCoE	48.32	-	56.95	0.496	33.52	-	45.47	0.349

2.26% on H@1 for WK31-60K$_{DE-EN}$. These improvements are statistically significant with $p < 0.5$ on a t-test. These improvements double confirm the effectiveness and robustness of our model on entity alignment tasks. Between the DuGa-DIT and DuGa-DIT (rand), we find the superiority of pre-trained entity embedding over random initialization on WK31-60K performance. Thus, in the rest of this study, we mainly report the DuGa-DIT results with the pre-trained entity embeddings.

4.5.2.5 Efficiency Analysis

We make comparisons between our proposed model and baseline models on performance and prediction times to evaluate the efficiency of our DuGa-DIT. The analyses are run on both DBP15K$_{FR-EN}$ and WK31-60K$_{FR-EN}$ datasaets, presented in Table 4.10.

We notice that the differences in parameter settings and environment running the models might have an impact on the time costs. But we try our best to provide an objective overview of these models by using the settings they reported in original papers. Table 4.10 shows that GM-EHD-JEA is the most time-consuming model on DBP15K$_{FR-EN}$, as its architecture adopts the time consuming joint entity alignment approach for prediction. Significantly different from GM-EHD-JEA, our proposed method shows much higher efficiency. As shown in Table 4.10, GM-EHD-JEA re-

Table 4.10: Performance and prediction time (seconds) for different models on DBP15K$_{FR-EN}$ and WK31-60K$_{FR-EN}$.

Methods	DBP15K$_{FR-EN}$		WK31-60K$_{FR-EN}$	
	Time	H@1	Time	H@1
GM-EHD-JEA	1,474	92.43	-	-
RDGCN	45	88.64	366	77.91
HGCN-JE	50	89.16	392	76.59
NMN	84	90.20	407	77.63
CAECGAT	42	94.68	382	78.24
DuGa-DIT	44	98.17	385	84.69

quires 1,474 seconds[4] on DBP15K$_{FR-EN}$ while our proposed model only requires 44 seconds. Compared to the preliminary study, CAECGAT, and other baselines, our DuGa-DIT still enjoys higher efficiency and more promising model performance. Even though on the WK31-60K$_{FR-EN}$ dataset, the time costs rise dramatically across all models due to the increasing number of parameters and test samples, our proposed model still managed to perform efficiently and effectively. This efficiency analysis reconfirms the superiority of our proposed model over other SOTA baselines.

[4]The results are reported in the original paper. As GM-EHD-JEA did not release the source code, we also do not report the model results on WK31-60K$_{FR-EN}$.

Table 4.11: Ablation Study.

Methods	DBP15K$_{ZH-EN}$			DBP15K$_{JA-EN}$			DBP15K$_{FR-EN}$		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
BASELINE	56.09	71.55	0.614	67.10	78.25	0.712	85.48	91.92	0.878
DuGa	69.17	89.96	0.765	81.29	94.58	0.861	94.42	98.96	0.961
DuGa w/o C	63.86	**91.92**	0.709	72.35	94.64	0.815	86.24	98.70	0.895
DuGa w/o I	57.98	76.62	0.644	76.78	86.39	0.812	91.29	96.30	0.935
DuGa-DIT$_{iter=1}$	78.48	89.54	0.822	89.34	95.14	0.913	97.61	99.21	0.982
DuGa-DIT$_{iter=2}$	**80.73**	88.17	**0.832**	**91.44**	**95.21**	**0.928**	**98.17**	**99.22**	**0.985**

4.5.2.6 Ablation Study

In this subsection, we conduct a series of ablation studies to learn the significance of the three main components of DuGa-DIT, namely the intra-KG attention layer, the cross-KG attention layer and the dynamic iterative training process.

In Table 4.11, the "BASELINE" refers to a naive model which aims to capture equivalent counterparts across KGs using the sum of word vectors from entities surface names as embeddings. The "DuGa" refers to the DuGa-DIT model without dynamic iterative training. The "DuGa w/o C" refers to the DuGa model without the cross-KG attention layer. The "DuGa w/o I" refers to the DuGa model without the intra-KG attention layer. The "DuGa-DIT$_{iter=1}$" and "DuGa-DIT$_{iter=2}$" refer to

our DuGa-DIT model with dynamic iterative training of 1 and 2 iterations respectively.

From Table 4.11, we note that, as demonstrated by prior works (211, 212), the "BASELINE" gets good experimental results as it absorbs surface names. The "DuGa" achieves better results than "BASELINE", which proves the significance of both the intra-KG and cross-KG attention mechanisms. The "DuGa w/o C" achieves less promising results than the "DuGa" which shows the necessity of including the cross-KG attention layer. Similarly, the "DuGa w/o I" also performs less promising than the "DuGa", because eliminating the intra-KG attention layer also hurts the model performance. The combination of our model with dynamic iterative training has shown great model performance improvement. The "DuGa-DIT$_{iter=1}$" outperforms "DuGa" greatly, while the "DuGa-DIT$_{iter=2}$" lifts the performance even more as we increase the number of iterations. This ablation study further strengthens the necessity of including each module in our DuGa-DIT architecture.

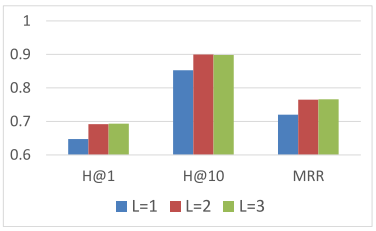
4.5.2.7 Impact of the Number of Layers

We conduct further experiments to learn the impact of number of dual KG attention layers on model performance with no dynamic iterative training. In Figure 4.6, we present the experimental results of DuGa with different number of layers, $L = 1$ to

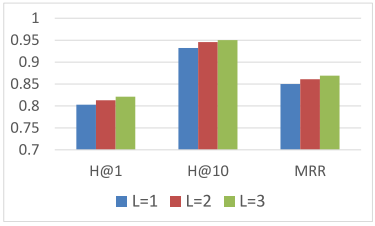
$L = 3$. We note that the DuGa model with two layers (e.g., $L = 2$) performs the best among the three different number of layers. The model performance of $L = 2$ is superior than $L = 1$ on all datasets which shows that stacking more layers can improve the model performance. The reason is that the added layer allows multi-hop neighborhood information and cross-KG alignment features to be propagated effectively. However, the $L = 3$ does not further improve the results which shows that further neighbors do not provide additional resourceful information for our entity alignment task. Moreover, stacking more layers also further increases the number of parameters in the model, which could hurt model efficiency. Therefore, we set 2 as optimal number of dual KG attention layers.

4.5.2.8 Impact of Different Proportions of Seed Alignments for Training

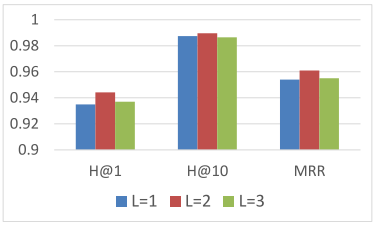
Aside from above, we also conduct experiments to learn the impact of different proportions of seed alignments for training. We choose 10%, 20%, 30%, 40% and 50% as the proportions of training, and the remaining entity alignments 90%, 80%, 70%, 60% and 50% as testing sets. The experimental results between DuGa-DIT and other baselines are shown in Figure 4.7. The DuGa is the version of DuGa-DIT without aggregation layer of the cross lingual KG. The general trend of model performances over all datasets follow an uprising pattern as the number of seed alignments goes



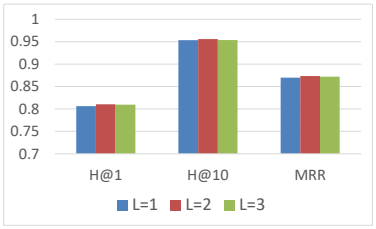
(a) DBP15K$_{ZH-EN}$



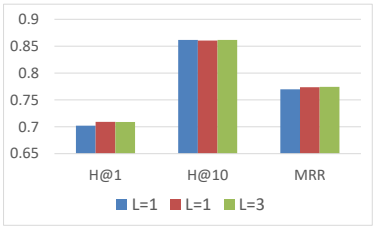
(b) DBP15K$_{JA-EN}$



(c) DBP15K$_{FR-EN}$



(d) WK31-60K$_{FR-EN}$



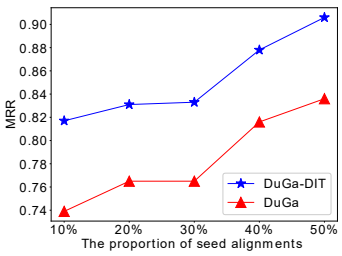
(e) WK31-60K$_{DE-EN}$

Figure 4.6: The results on DBP15K and WK31-60K with different number of layers.

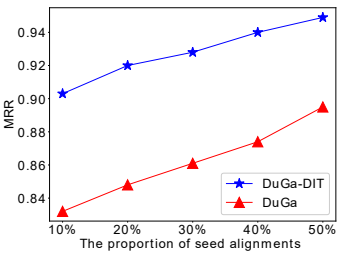
up. Our proposed model have demonstrated its effectiveness and robustness on the CEA task by adopting an iterative approach and adding in new seed alignments dynamically. The DuGa-DIT trendy lines in blue color are much smoother compared to the ones in red representing DuGa. This implicates the robustness of our proposed model with limited seed alignments.

4.5.2.9 Results for Entities with Different degrees

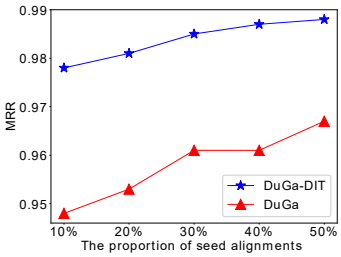
The degree of entities plays a significant role in GNN-based model performance. In our proposed model, neighboring features aggregation is done within the intra-KG



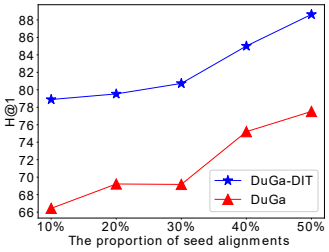
(a) MRR on DBP15K$_{ZH-EN}$



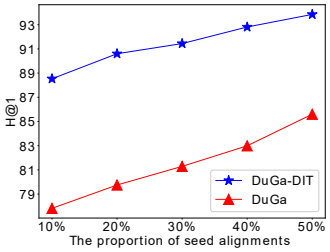
(b) MRR on DBP15K$_{JA-EN}$



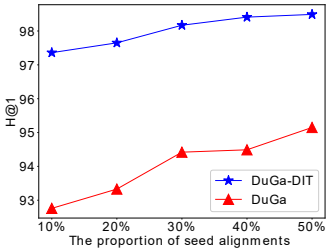
(c) MRR on DBP15K$_{FR-EN}$



(d) H@1 on DBP15K$_{ZH-EN}$



(e) H@1 on DBP15K$_{JA-EN}$



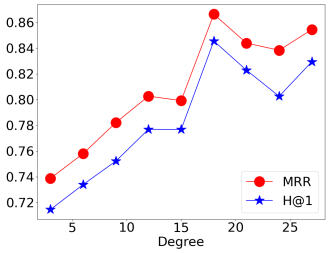
(f) H@1 on DBP15K$_{FR-EN}$

Figure 4.7: Performance on DBP15K using different proportions of seed alignments.

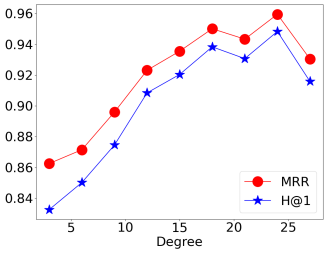
attention layer. The degree of entities is linked to the richness of the neighboring features. Figure 4.8 is a good illustration of the impact of entity degrees on Hits@1 and MRR metrics of DBP15K and WK31-60K. For DBP15K$_{ZH-EN}$ and DBP15K$_{JA-EN}$, both MRR and H@1 metrics increase as the degree increases, and both metrics decrease slightly as the degree decreases slightly. For the DBP15K$_{FR-EN}$ specifically, the metrics increase rapidly as the degree reaches 10, and keep stable at around the 0.96-0.98 level. Intuitively, we believe that large degree-entities would have more neighbors. Thus, we can capture more of their neighboring features, which are key to the CEA task, leveraging the intra-KG attention layer. The DBP15K$_{FR-EN}$ dataset contains similar entities surface forms between English and French which makes the entity alignment relatively easy. Thus, evaluation metrics can reach high accuracies on DBP15K$_{FR-EN}$ with fewer degrees. Similarly, we can also observe similar trends on the WK31-60K dataset.

4.5.2.10 Impact of the Number of Negative Samples

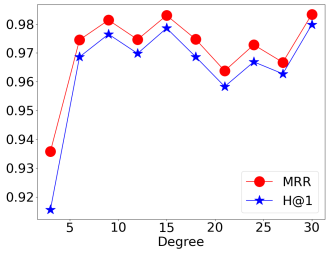
In this study, the number of negative samples is considered as an important hyper-parameter. We sample a number of negative entity pairs to perform the objective function computation for model training. In order to investigate the impact of the number of negative samples, we compare the experimental results with various num-



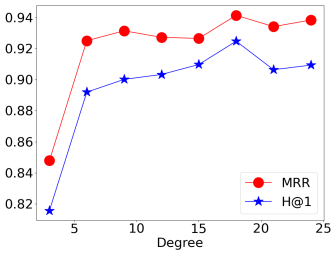
(a) DBP15K$_{ZH-EN}$



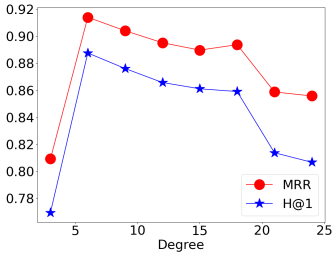
(b) DBP15K$_{JA-EN}$



(c) DBP15K$_{FR-EN}$



(d) WK31-60K$_{FR-EN}$



(e) WK31-60K$_{DE-EN}$

Figure 4.8: The results on DBP15K and WK31-60K for entities with different degrees.

Table 4.12: Results on DBP15K with different number of negative samples.

	DBP15K_{ZH-EN}			DBP15K_{JA-EN}			DBP15K_{FR-EN}		
Negative Samples	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
10	68.67	88.92	0.762	80.76	94.67	0.858	93.44	98.95	0.957
20	68.85	89.90	0.763	80.74	94.40	0.857	93.93	98.87	0.957
30	**69.17**	**89.96**	**0.765**	81.29	94.58	0.861	**94.42**	98.96	**0.961**
40	68.76	89.85	0.763	81.41	**94.64**	0.862	94.16	98.09	0.959
50	68.72	89.68	0.761	**81.58**	94.57	**0.863**	94.29	**99.07**	0.960

bers of negative samples on the DBP15K dataset. Table 4.12 shows the experimental results of different number of negative samples, ranging from 10 to 50, on DuGa-DIT. The general trend can be summarized as follows: as the number of negative samples increases, the model performance improves respectively. However, it is worth noting that the degree of improvement is restricted once it surpasses 30. Therefore, taking both the effectiveness and efficiency of the model into consideration, the optimal number of negative samples is set to be 30.

4.6 Conclusions and Future Work

In this chapter, we propose both CAECGAT and DuGa-DIT with dynamic iterative training for multiple graph learning on the CEA task. Both approaches make full use

of the pre-aligned seed alignments to try alleviate semantic gaps between different KGs.

In the CAECGAT model, we transfer entities information across different KGs over a cross-KG aggregation layer. It enables the embeddings from various KGs to share cross-KG information. After that, we gather important neighboring cross-KG information over an attention-based cross-KG propagation layer. The stacking architecture of multiple CGAT layers allows cross-KG information propagation to happen between two KGs.

In the DuGa-DIT model, we aggregate local neighboring features with intra-KG attention layer and consolidate cross-KG alignment information with a cross-KG attention layer. The embeddings of two KGs are thus updated by merging these two features together. Besides, we update the cross-KG attention score matrices with dynamic iterative training. This allows us to capture richer cross-KG information.

We run extensive experiments on benchmark datasets for both models. As analyzed above, these experimental results demonstrate our model effectiveness and robustness over SOTA methods on the CEA task.

Lastly, we would like to present several possible directions of future work, presented as follows:

- The proposed models in this chapter simply consider entity alignments across

two KGs. In future, the task could increase difficulty by practising entity alignments across three or more KGs. This would require us to learn auto-alignments and to take more entity information into consideration.

- The second promising research direction could be to design a distributed auto alignment method. In practice, the size of KGs is normally extensive, especially for downstream tasks like personalised recommendations or personalised search. It thus becomes necessary to have a distributed auto alignment method to tackle this potential issue.

- We could also continue applying our model to the CEA task in other languages, such as English to Spanish, English to Portuguese, or English to South East Asia-based languages.

5 Learning on Real-World Problems: Cell Node Classification for COVID-19 Disease Prediction

In this chapter, we dive into the third module of this thesis on GATs for an ongoing real-world problem, the COVID-19 disease prediction. We introduce a new GNN-based method which leverages single-cell RNA sequencing (scRNA-seq) data for predictions on the Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2) infection. We develop a novel graph attention capsule network (GACapNet) model to perform the node classification task on infected cells of different severity levels. Specifically, we consider genes of the scRNA-seq data as cell features to input into our model. As a result, our GACapNet effectively identifies the patterns of SARS-CoV-2 transcriptome and associates various types of molecular cells with different degrees of disease infectiousness.

5.1 Introduction

The month of December, 2021 marks the two-year anniversary since the World Health Organization announced the global outbreak of COVID-19. This COVID-19 pandemic has spread across the globe and has hit us hard with severe public health and economic consequences. The infection of the Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2), similar to its same virus family members of Severe Acute Respiratory Syndrome (SARS) and Middle East Respiratory Syndrome CoV (MERS-CoV), can lead to fatal pneumonia that is in association with rapid replication of the deadly virus, an elevation of proinflammatory cytokines, as well as an infiltration of immune cells. While people from all over the world are anticipating the roll-out of a third dosage of effective coronavirus vaccines, a potentially more deadly, contagious and transmissible variant, Omicron, was found by various countries by the end of 2021[5]. The cost of human capital continues to mount over the past few months, with more than 265.2 million cases confirmed worldwide, more than 20.9 million active cases and more than 5.26 million deaths as of December, 2021[6].

During the course of two years, a great amount of COVID-19 related data from a wide range of sources and formats have been accumulated. These data collections

[5]https://www.channelnewsasia.com/business/imf-chief-warns-omicron-covid-19-variant-could-slow-global-growth-2357731

[6]https://www.worldometers.info/coronavirus/

are considered as valuable and key theoretical basis and evidence for further clinical research and biomedical analyses. Since the disease belongs to the department of respiration by nature, most of the existing COVID-19 related data collections are constructed and presented in a graph structure. Indeed, there is a great portion of important real-world information and data that have long been presented in a form of graph structure before COVID-19. These graph structured data include person-person relationship in social networks (41, 115), product knowledge graph for e-commerce (67, 123, 175), medical knowledge graph in healthcare industry (122, 127, 142, 151, Lally et al.), and those publicly available COVID-19 related knowledge graphs (10, 141, 178, 180, 193, 225) etc.. Even though various evidences have hinted it as an important research direction for machine learning to extend neural networks to process graph structured data, this research domain has not caught high attention until very recently.

Graph Attention Networks (GATs) (170) is a lately introduced neural network architecture that is capable of operating graph structured data and supporting predictive tasks, such as node classifications and link predictions. In the past few years, a great number of studies have been published using the GAT and conventional GNN approach. Specifically, these models have been widely adopted to explicit relational data structures such as knowledge graphs (46, 189), non-explicit relational

data structures such as texts (170), and other applications such as generative models (12). Although these existing studies have shown promising results, they also share one deficiency that they usually model each interactions individually without considering complex relations between them enough. This characteristic of these past studies restrict them from absorbing more relevant and useful neighborhood features, and transforming them into more useful outputs. In the context of COVID-19, it is important for us to empower our current medical research with these advanced machine learning methodologies to uncover secrets hidden in the infection and pathogenesis of SARS-CoV-2, and to suggest viable therapeutic directions.

Given that the counts of transcripts are correlated with expressions of genes, the technology of single-cell RNA sequencing (scRNA-seq) is often used to capture thousands of expressions of cell genes under different circumstances to comprise large data collections (56, 155, 233). Even though massive single-cell data can be captured effectively, the high sparseness, high heterogeneity and high dimensions characteristics of these single-cell data pose challenges in sorting out significant features that can formulate causal relations to the pathophysiological trajectory and interactive reactions.

Our research interest thus arises from two recently published work (129, 139). The first paper applies GATs to disease states prediction with scRNA-seq data

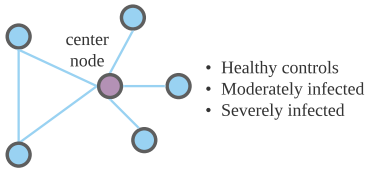


Figure 5.1: An example of sub-graph in the COVID-19 patients (81) dataset. The purple node represents the center cell node and the blue nodes represent the neighboring cell nodes. All cells are classified into three infectious levels, namely the healthy controls, moderately infected and severely infected.

(128). While the second paper aims to apply quantitative methods to understand the characterizations of genes, cells, as well as changes in cell states interrelated with SARS-CoV-2 infection in the human airway.

Aligning with the work above, we focus on identifying the patterns of SARS-CoV-2 transcriptome and the types of molecular cells linked to the degree of disease infectiousness and severity with single cell transcriptomic data collections. However, different from all prior works, in this paper, we take a new approach by proposing a graph attention capsule network (GACapNet) which stacks the multi-head attention networks above a capsule network, comprising of a primary capsule layer and a dynamic routing procedure. To the best of our knowledge, we are the first one to extend GATs and Capsule networks to scRNA-seq datasets for COVID-19 disease

state predictions.

The proposed architecture GACapNet has a strong advantage in dealing graph structured data and generating more useful and representative node features for enhancing GATs' performance on cell node classification. To align with the prior work (139), we use the processed and clustered dataset in graph structure as inputs to our model. The prior work constructed the graphs from cells by utilizing batch-balanced weighted KNN graph (124) with matrices of gene features. In other words, the structure of the graph was constructed by finding the closest cell nodes with KNN. The task of our study is to identify and classify SARS-CoV-2 infected cells that are linked to different degrees of infectiousness. For example, Figure 5.1 shows an example of a sub-graph in the dataset which aims to group cells into different levels of cell node infectiousness, namely the severely infected class, the mildly infected class, and the healthy controls. All cells are grouped under the assumption that the cells with similar gene features should be closer to each other. If the two cell nodes belong to the same category, an edge is formed in between these two nodes.

For a given constructed graph $\mathcal{G}$, we first use multi-head graph attention to aggregate features from the neighbors of a center node, shown as the purple node in Figure 5.1. The center node stands for a cell with a sequence of gene features. Each cell in the dataset can be a center node, and the surrounding nodes are defined

as neighboring cell nodes, shown as the blue nodes in Figure 5.1. In order to preserve the center node feature (or the cell's feature), we use a simple feed-forward network on the node features to obtain a transformed node feature from the center node. Then, the neighborhood features generated from different attention heads and those transformed node features are fed into a capsule network. The capsule network comprises of a primary capsule layer and a dynamic routing process. From the experimental results on two scRNA-seq datasets, we prove and reason that our proposed GACapNet is able to classify cells according to its severity level effectively. In other words, it outperforms other SOTA models in extracting insightful data of SARS-CoV-2 transcriptomic patterns and molecular cell types that are linked to the COVID-19 severity prediction.

This work makes four primary contributions in general, which are summarized as follows:

- We apply advanced GATs and Capsule networks to the COVID-19 domain of research. To the best of our knowledge, our GACapNet is the first one to extend the GATs and Capsule networks models to scRNA-seq data, aiming to uncover the secrets hidden in the infection of SARS-CoV-2, to understand the COVID-19 pathogenesis, and to suggest the therapeutic directions and development by identifying the patterns of SARS-CoV-2 transcriptome and the

types of molecular cells that are linked to the degree of disease infectiousness and severity;

- We propose to use multi-head graph attention networks in our model architecture to aggregate more information-rich neighborhood features, and to preserve its original node feature to the maximum;

- We incorporate a capsule layer with dynamic routing into our model to effectively combine and fuse the gene features, and to allow more prominent features of the severely infected cells and healthy controlled cells to present in the output;

- The outstanding experimental results of our model on two scRNA-seq data collections can be used as basic hypotheses for the future COVID-19 research and clinical validations.

The remainder of this chapter is organized as follows. Section 2 gives a thorough literature review on the relevant research background. Section 3 describes the architecture of our proposed GACapNet model in details. Section 4 illustrates our experimental results and presents our model comparisons. Lastly, in Section 5, we conclude our chapter with some ideas for future research.

5.2 Related Work

To address our proposed approach, we elaborate on mainly two fields of related studies, namely the modeling graphical data using Graph Neural Networks (GNNs) and machine learning for COVID-19.

5.2.1 Modeling Graphical Data using GNN

In this section, we introduce the concept of GNN, a non-spectral learning method, following by some of its downstream applications, such as text classification, relation extraction, image classification, knowledge graph, graph generation, as well as the drug and protein interaction mining etc..

Scarselli et al. (137) were the first researchers who proposed the concept of GNN. GNN extends the existing neural network models to serve the processing needs of data in graph structures. In general, the aim of a GNN model is to capture neighborhood information of each node and to generate a state embedding of $h_i \in \mathcal{R}^m$, where the h_i stands for a hidden state of node and R^m is the m-dimensional Euclidean space. Several variants of the original GNN model have been proposed recently, and one of the variants is the GAT model. GATs have gained quite amount of attention by the machine learning research community in recent years. It is introduced by Veličković et al. (170) in 2018. Its structure of stacking masked self-attentional layers, which

assigns different weighting scores to different neighborhood features, overcomes the shortcomings of prior models and lowers computational costs.

Recently, GNN-based methods have achieved great success in processing graph structured data. Tasks such as link prediction on general knowledge graphs (202), entity alignment on cross lingual knowledge graphs (204), personalised recommendation on product knowledge graphs (174)and biomedical knowledge graph for drug discovery (201, 232) have gained significantly more promising results. Lately, CapsGNN (205) was proposed to extract features of nodes with capsules so that the routing mechanism could be applied to capture graph-level significant information from various aspects. Even though CapsGNN (205) is also an integration of GNNs and capsules, CapsGNN uses GCN (65) rather than GAT (170) in the model architecture. Moreover, CapsGNN utilizes the output of each layer as capsule whereas our proposed GACapNet is an improvement and optimization of the GAT model, where capsules in our model refer to the output of multi-heads attentions.

The most commonly seen downstream applications for these GNN models comprises of text classification (5, 35, 47, 65, 120, 170, 219, 228), relation extraction (102, 153, 229), image classification (36, 76, 186), knowledge graph (46, 189), graph generation (12, 114, 222), graph classification (11) as well as the drug and protein interaction mining (33, 132, 240). The comprehensive discussion about these works

are out of the scope of this paper. Here, we mainly focus on utilizing GAT methods to effectively predict disease state and to derive links between our single-cell data and the infection of SARS-CoV-2. Up to date, we find limited collection of prior work in this field, which consists of mainly (129, 139). This provides opportunities for more advanced machine learning models to step in.

5.2.2 Machine Learning for COVID-19

Machine learning refers to a sub-field of artificial intelligence studying computer algorithms which are capable of building, improving and automating analytical models through learning experience with minimal human intervention. Despite the data in medical domain are able to provide insights to diagnosis and treatments, they have not been well-understood for long because of its nature of high volume, incompleteness, heterogeneity, and complexity. These unique characteristics of medical data leave huge space for machine learning, especially deep learning, models to show their talents in uncovering unknown knowledge.

In fight with the global community against COVID-19, our machine learning fellows have mostly focused on three areas of research: 1) text analytics, machine translation and social media studies for COVID-19 literature; 2) image studies of CT scans and X-rays for COVID-19 diagnosis and detection; and 3) disease prediction

with various relevant features.

The first area of research aims to build text analytics tools, machine translation and social media studies. An end-to-end recommender system for academic research paper was introduced by Shen et al. (141) to match these papers with potential downstream COVID-19 use-cases. A COVID-19 knowledge graph (CKG) was constructed by Wise et al. (193) to illustrate complex relations between enriched entity information and graph structured COVID-19 scientific articles, utilizing Amazon Web Services (AWS) technologies and latent schemas. An EVIDENCEMINER system was presented by Wang et al. (184) to facilitate the COVID-19 literature text mining process for researchers and scholars. Aside from that, a series of empirical studies on the retrieval results of the COVID-19 from both Google and Amazon were conducted and compared to academic prototypes evaluated by the TREC-COVID track (133). Other related literature in COVID-19 text analytics and knowledge bases include (2, 4, 43, 156).

The second area of research aims to assist to diagnose and treat patients in clinical scenarios with deep learning techniques, CT scans and X-rays. COVID-Net (177) is an open-sourced disease diagnosis and detection system specializing in COVID-19 cases using chest X-ray images. Similarly, Gozes et al.(42) applies 2D and 3D CNN algorithms to develop an automated analytical tool of CT Chest scan for COVID-19

diagnosis and detection. Other related publications include but are not limited to CT images screening(183), auto-detection of coronavirus using X-ray and CNN(103), and classification of COVID-19 X-ray images using DeTraC and CNN (1). However, these work mainly focus on image-based deep learning approaches. We exclude them as out of the research scope of this specific study.

The third area of research aims to practise disease prediction with various features. Given the scope of this study, we mainly focus on molecular and cellular features clustering and classification, which is as significant as the other areas aforementioned. Ravindra et al. (128) present a deep learning approach to predict disease state with GAT applied on Multile Sclerosis (MS) patients and single-cell data. Sehanobish et al. (139) propose to combine edge features generated from a set transformer and node features generated from GNN architectures to effectively practice node classification task in a self-supervised learning manner. The purpose is to identify the key cells and genes from scRNA-seq datasets that could lead to severe infectiousness of COVID-19. Other studies in this field include but are not limited to (81, 129, 234). Ravindra's study (129) performs single-cell RNA sequencing of infected human bronchial epithelial cells (HBECs) and presents the major targets of infections in their study. Liao et al. (81) collects and classifies bronchoalveolar lavage fluid immune cells from COVID-19 patients, who are of different severity lev-

els, to acquire a landscape of bronchoalveolar immune cells across different patients' profiles for disease prediction. Zheng et al. (234) proposes a hybrid AI model with traditional epidemic model, a NLP module and a long short-term memory (LSTM) network to predict COVID-19 diseases. In summary, our proposed study pioneers the research in both machine learning and medical domain by extending the GATs and Capsule networks to COVID-19 disease state prediction with scRNA-seq data. Significantly different from the preliminary studies (129, 139), our proposed study takes a novel approach by applying a multi-head graph attention network to learn local structure features in a graph first. Then, it uses a capsule layer to transform node feature vectors into multiple capsules and combines them through dynamic routing procedure for more representative and significant output.

5.3 Methods

In this section, we first introduce the overview of our proposed GACapNet architecture. Following that, we present the three major compositions of our model: (i) the multi-head graph attention networks to learn local structure features in a graph; (ii) the primary capsule layer of the capsule networks to transform each node feature vector from the multi-head graph attention layer into multiple capsules using feed-forward networks; (iii) the dynamic routing procedure of the capsule networks to

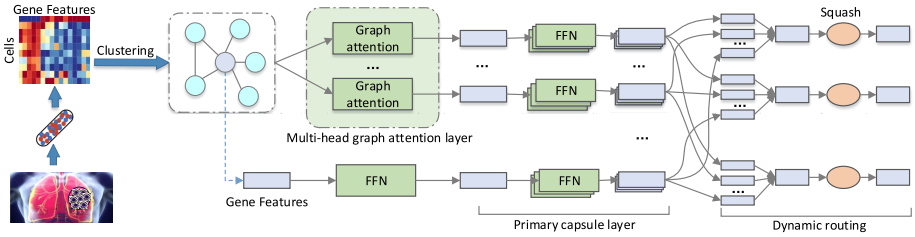


Figure 5.2: The architecture of the proposed graph attention capsule network (GACapNet). The model consits of multi-head graph attention layer, primary capsule layer and dynamic routing layer. The multi-head attention layer is used to generate features from different aspects. In the primary capsule layer, we transform the node features into multiple capsule vectors. Then, the dynamic routing layer is applied to obtain output capsule vectors.

combine different input features and to determine which information to be generated as output capsules.

5.3.1 Overview

Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{V}, Y)$, where $\mathcal{V}$ denotes a set of nodes, $\mathcal{E}$ denotes a set of edges, $\mathbf{V}$ and Y denote the node attribute matrices and node labels, respectively. The goal of the node classification task is to learn node representations and a classification function which can be used to map the node vectors into corresponding node labels.

The architecture of our proposed GACapNet is illustrated in Figure 5.2. The

proposed GACapNet consists of a multi-head attention layer and a capsule networks layer. For a given graph $\mathcal{G}$, we first use the multi-head graph attention to aggregate features from neighbors of the center node. To preserve the features of this center node, we also use a simple feed-forward network on the node features to obtain a transformed node feature from the center node itself. Then, the neighborhood features from different attention heads and the transformed node feature are fed into a capsule network with primary capsule layer and dynamic routing process.

5.3.2 Sequence to Graph

The input graph structure of our proposed GACapNet was created from cells by gene counts feature matrices which stem from the collection of single-cell samples. Sehanobish et al. utilized Scanpy (136, 194) to preprocess all single cell samples using the standard preprocessing pipeline of single cell RNA-seqencing. Then, a batch-balanced weighted KNN (BB-KNN) (124) method was applied to construct the KNN graph by identifying the top-K nearest neighbours of the center cell node in the batch. Specifically, the gene feature matrices are mapped into low-dimensional principle component analysis (PCA) space. Then, for two nodes v_i and v_j, the similarity score is calculated by using the Euclidean distances between the PCA

vectors of these two nodes:

$$d_{ij} = ||\mathbf{x}_i - \mathbf{x}_j||_2 \tag{5.1}$$

where $\mathbf{x}_i$ and $\mathbf{x}_j$ are the PCA vectors for nodes v_i and v_j, respectively. Based on the distance between nodes, k nearest neighbors were generated by applying BBKNN algorithm and the distance between respective nodes was generated as edge weights.

Conventionally, the KNN algorithm identifies the k nearest neighbors for each cell node across the entire data point collections. These cell nodes thus become massively connected network, which is hard to link up cell types across various batches, for further studies. Instead of dealing with the entire data collection directly, BBKNN identifies each cell's relatively smaller k nearest neighbors by batch. These batch-based KNN results are integrated to the final cell node neighboring list. This allows the analogue cells to be connected from different batches without making changes in counts and PCA space.

5.3.3 Multi-head Graph Attention

GNNs have shown promising results on modeling graph-structured data in the past. In this study, we use GATs (170) to learn the local structure features in a cell graph. Each node $v_i \in \mathcal{V}$ in the graph can be represented as a vector $\mathbf{v}_i$. As shown in Figure 5.2, we use multi-head graph attention layer to model the given graph. For

each graph attention head, the input is the node features in the graph and the output is a set of updated node features. Formally, for a node v_i, we can update its features by aggregating the neighbors using a weighted sum function:

$$\mathbf{h}_i^m = \sigma\left(\sum_{v_j \in \mathcal{N}_{v_i}} \alpha_{ij}^m \mathbf{W}^m \mathbf{v}_j\right) \tag{5.2}$$

where $\mathcal{N}_{v_i}$ is a set of neighboring nodes of v_i, $\mathbf{v}_j$ is the vector representation of the neighboring node $v_j \in \mathcal{N}_{v_i}$, $\mathbf{h}_i^m$ is the updated node feature for v_i using the m-th head, σ is an activation function, $\mathbf{W}^m$ is the transformation parameters, and α_{ij}^m is the attention weight for v_j using the m-th head, which is computed by normalizing the attention scores of the neighboring nodes:

$$\alpha_{ij}^m = \frac{\exp(s_{ij}^m)}{\sum_{v_j' \in \mathbf{N}_{v_i}} \exp(s_{ij'}^m)} \tag{5.3}$$

where s_{ij}^m is the attention score for node v_j, which is computed as:

$$s_{ij}^m = \text{LeakyRelu}(\mathbf{W}_1^m[\mathbf{v}_i||\mathbf{v}_j]) \tag{5.4}$$

where $||$ is a concatenation operation, $\mathbf{W}_1^m$ is a trainable parameter and LeakyRelu is the activation function.

To better preserve the original feature of the node, we also apply a simple feed-forward network (FFN) to update the node representation, which is denoted as:

$$\mathbf{h}_i^0 = \sigma(\mathbf{W}_1^0 \mathbf{h}_i + \mathbf{b}^0) \tag{5.5}$$

where $\mathbf{W}_1^0$ and $\mathbf{b}^0$ are the trainable parameters.

By applying multiple graph attention heads and the FFN layer, we can obtain a set of new node features for each node v_i, which is denoted as $V_i = \{\mathbf{h}_i^0, \mathbf{h}_i^1, \mathbf{h}_i^2, \cdots, \mathbf{h}_i^M\}$.

5.3.4 Primary Capsule Layer

The node features obtained from multi-head graph attention layers can be viewed as different aspects of this node. We use a capsule layer with a primary capsule layer and dynamic routing process to compute the output features of the node.

As shown in Figure 5.2, in the primary capsule layer, each node feature vector from the multi-head graph attention layer can be transformed into multiple capsules using feed-forward networks. Formally, a capsule vector produced in the primary capsule layer can be computed as:

$$\mathbf{z}_i^t = \text{squash}(\mathbf{W}_z \mathbf{h}_i^m + \mathbf{b}_z) \tag{5.6}$$

where squash (135) is a non-linear activation function, which is denoted as:

$$\text{squash}(x) = \frac{||x||^2}{1 + ||x||^2} \frac{x}{||x||} \tag{5.7}$$

By applying the primary capsule layer, we can obtain a list of primary capsule vectors for the node v_i, which is denoted as $Z_i = \{\mathbf{z}_i^1, \mathbf{z}_i^2, \cdots, \mathbf{z}_i^T\}$.

5.3.5 Dynamic Routing

Each capsule vector represents a specific property of the node. The dynamic routing layer is used to combine these input features and to determine what information should be used to generate the output capsules. As shown in Figure 5.2, a vote vector is computed by using a linear transformation from each input capsule vector. It is denoted as:

$$\hat{\mathbf{z}}_i^{k|t} = \mathbf{W}_{kt}\mathbf{z}_i^t \tag{5.8}$$

where $\mathbf{W}_{kt}$ denotes the transformation parameter from the t-th input capsule vector to the k-th output capsule vector.

Then, all the vote vectors are combined using a weighted sum function to obtain a new capsule s_i^k:

$$\mathbf{s}_i^k = \sum_t c_i^{kt}\hat{\mathbf{z}}_i^{k|t} \tag{5.9}$$

where $c_i^{kt} = \frac{\exp(b_i^{kt})}{\sum_t \exp(b_i^{kt})}$ is a coupling coefficient between input capsule $\mathbf{z}_i^t$ and the output capsule $\mathbf{s}_i^k$, and it is determined by a dynamic routing process with a zero initialized logits b_i^{kt}.

After that, we apply a non-linear squashing (135) to each capsule vector $\mathbf{s}_i^k$:

$$\mathbf{u}_i^k = \text{squash}(\mathbf{s}_i^k) \tag{5.10}$$

In the dynamic routing process, the logits b_i^{km} can be iteratively updated by

measuring the agreement between the output $\mathbf{u}_i^k$ and the vote vector $\hat{\mathbf{h}}_i^{k|m}$:

$$b_i^{kt} \leftarrow b_i^{kt} + \hat{\mathbf{z}}_i^{k|t} \cdot \mathbf{u}_i^k \tag{5.11}$$

5.3.6 Model Optimization

In the dynamic routing capsule layer, there is a total K capsules obtained, where K stands for the number of node labels in the node classification task. Thus, we use the length of each output capsule to compute the probability of each class:

$$p_i^k = \frac{\exp(||\mathbf{u}_i^k||)}{\sum_k \exp(||\mathbf{u}_i^{k'}||)} \tag{5.12}$$

To optimize the parameters in the model, we use a cross-entropy loss as the objective function:

$$\mathcal{L} = -\sum_i \sum_k y_i^k \log(p_i^k) \tag{5.13}$$

where $y_i^k = 1$ if the node v_i is an instance of the k-th class, otherwise $y_i^k = 0$.

5.4 Experiments

5.4.1 Datasets

In order to evaluate the effectiveness of our model, we run experiments on two scRNA-seq data collections, namely the SARS-CoV-2 infected organoids and the COVID-19 patients.

The SARS-CoV-2 infected organoids (129) is a dataset on infected human bronchial epithelial cells which were infected by SARS-CoV-2 and co-cultured for 1, 2, and 3 days-post-infection (dpi). A cell is considered to be infected if more than 10 transcripts are aligned to the SARS-CoV-2 genome in the cell. The cells that were not infected in 1, 2, 3 dpi samples are labeled as bystander cells. Besides, there are some controlled samples, labeled as "Mock", which stands for neither bystander nor infected. Given that, all the cells can be grouped into 7 classes according to whether the cells are infected or bystander in the 3 time-stamp points, namely the Mock, infected_1dpi, bystander_1dpi and etc.. Aligning with the prior work, our study constructed on the assumption that all SARS-CoV-2 infected organoid cells show transcriptomic signatures of them being infected or bystander cells, which differentiate them from mock-infected or control sample cells.

The COVID-19 patients (81) is a dataset collected from Shenzen Third People's Hospital in Guangdong Province, China. The dataset supports the node classification task on COVID-19 infected cells, and identifies those cells that are highly expressed by severely infected patients and other cells that are preferentially expressed by healthy controls and moderately infected patients. Specifically, the dataset, consisting of 12 patient samples, allows us to use 10x genomics to measure scRNA-seq to classify different lung immune microenvironment in the bronchoalveolar lavage

Table 5.1: Statistic of the datasets.

Dataset	Nodes(train/val/test)	Node features	Edges(train/val/test)	Classes
SARS-CoV-2 infected organoids	54353/11646/11648	24714	1041226/230429/228630	7
COVID-19 patients	63486/13604/13605	25626	2746280/703217/707529	3

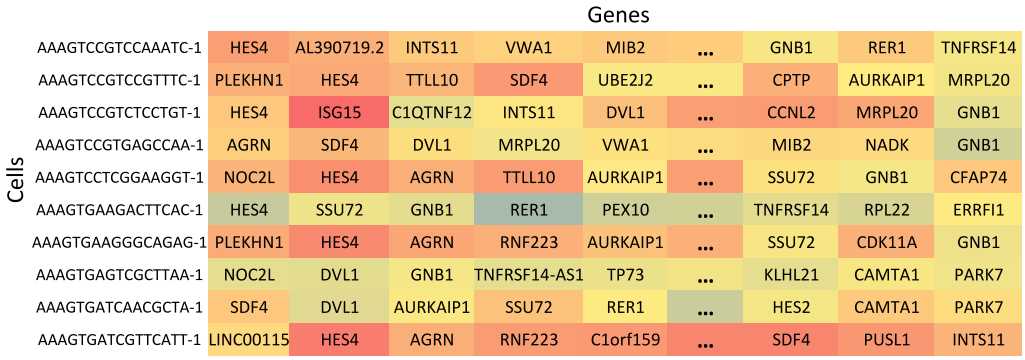


Figure 5.3: Some examples of the cell nodes in the graph.

fluid (BALF) comprehensively. Among these 12 samples, more than 25,000 node features are collected from 3 healthy control samples, 3 moderately infected samples and 6 severely infected samples. The categorization comes from the Diagnosis and Treatment Protocol of COVID-19 (the 7th Tentative Version) released by the National Health Commission of China. Aligning with the prior work, we further split the dataset into 63,486 nodes of cells for training, 13,604 nodes of cells for validation, and 13,605 nodes of cells for testing. All cells taken from patient's sample were labelled in accordance with a severity level.

The details of these datasets are summarized in Table 5.1. For the purpose of presenting the formats of the datasets clearly, we show some examples of the cells in

Figure 5.3. The input graph structure of our proposed GACapNet is the processed and clustered dataset from prior work (139). Sehanobish et al. (139) constructed the cell graphs by utilizing batch-balanced weighted KNN graph (124) with matrices of gene features. The collection of gene features is used as cell features to input into the model. For more details about the data collections, please refer to (139). As illustrated in Figure 5.3, each row represents a cell which contains some gene features, and each column denotes the gene features represented in the corresponding cells. For example, "AAAGTCCGTCCAAATC-1" is the cell name, "HES4", "AL390719.2", ..., "TNFRSF14" represent the specific gene features contained in the cell "AAAGTCCGTCCAAATC-1". Thus, these gene features representing the characteristics of cells reflect the ones that were infected by SARS-CoV-2 and the ones that were healthy controls in the RNA sequences. The edges between these cells are constructed by identifying the top k nearest neighbors, which aligns with the prior work (139).

5.4.2 Parameter Settings

We optimize the parameters of our model using the Adam (64) optimizer with an initial learning rate of 0.001. For each dataset, we use grid search to select hyper-parameters according to the accuracy on validation dataset. Specifically, we select

the number of heads in multi-head graph attention layer from $\{2, 4, 6\}$, the hidden size of each attention vector from $\{8, 16, 32\}$, the number of primary capsules from $\{1, 2, 3, 4\}$, the size of each capsule vector from $\{4, 8, 16\}$, and the number of iterations in dynamic routing from $\{2, 3\}$. Finally, the number of heads in multi-head graph attention layer is set to 2 for SARS-CoV-2 infected organoids and the COVID-19 patients, since these parameter settings obtain the best performance on the validation dataset. The hidden size of each vector in multi-head attention layer is set to 32. Each node feature in graph attention layer can be transformed into 4 primary capsule vectors and the dimension of each capsule is set to 8. The number of iterations in dynamic routing is set to 3. The batch size is set to 128. The dropout rate is set 0.4 and the weight decay is set to 4e-5.

The code of our model is implemented using Pytorch and all the experiments are conducted on a server with a single NVIDIA GTX 1080Ti GPU [7].

5.4.3 Comparing Methods

We compare our model with a series of strong baselines, summarized as follows.

- ClusterGCN (24) uses a graph clustering algorithm to sample a dense subgraph, and then performs GCN (65) on the subgraph.

[7]We will release the code and datasets for research community once the paper is accepted at https://github.com/sherrybbalonsozhu.

- GAT (170) uses self-attention mechanism to aggregate neighborhood features.
- The methods ClusterGCN + DeepSet, ClusterGCN + Set2Set, and ClusterGCN + Set Transformer use ClusterGCN to aggregate neighborhood features and use different methods to encode edge features (e.g., Deep Set (224), Set2Set (171) and Set Transformer (77)).
- Similarly, the methods GAT + DeepSet, GAT + Set2Set, GAT + Set Transformer use GAT to learn neighborhood features and incorporate edge features using Deep Set (224), Set2Set (171) and Set Transformer (77).
- GIN + EdgeConv (52) incorporates edge features into Graph Isomorphism Network (GIN) (210).
- EdgeConditionedConvolution (147) uses graph convolutions with edge-conditioned filters to incorporate edge features.
- GraphSAGE (47) uses rich node features, such as node attributes, node degrees to learn better node embeddings.
- CapsGNN (205) uses GCN and capsule neural network to extract high-level features.

5.4.4 Main Results

In this section, we present the experimental results of our proposed study. The main experimental results on the test sets are summarized in Table 5.2. We use the accuracy metric on the test sets to measure the performance. For each dataset, we run the model 10 times with different random seeds and report the average accuracy as well as the minimum and maximum accuracies in the bracket.

We compare our model with some strong baselines listed in Table 5.2. Besides the model GraphSAGE and CapsGNN, the results of other baselines are taken from (139). The proposed GACapNet outperforms all the baseline models on both datasets. On COVID-19 patients dataset, our model achieves the best result with an accuracy of 99.21%, outperforming all the baselines. On SARS-CoV-2 infected organoids dataset, our model also shows superior performance and achieves the best result among all the SOTA methods. It is worth noting that GAT+Set Transormer model incorporates many edge features pre-trained by predicting metadata in the graph, which is not always available in practice. Instead, the proposed GACapNet obtains better performance without using any additional edge features. Furthermore, our GACapNet outperforms the CapsGNN by 4.13% on SAR-CoV-2 organoids dataset and 5.38% on COVID-19 patients respectively. It proves that even though both models used the integration of GNNs and capsules, our proposed GACapNet,

Table 5.2: Experimental Results on node classification task in terms of accuracy evaluation metric. The bold fonts show the best results, and the underlined fonts show the second best results. We run the model 10 times with different random seeds and report the average accuracy as well as the minimum and maximum accuracy in the bracket.

Models	SARS-CoV-2 infected organoids	COVID-19 patients
ClusterGCN	65.43 (65.21-65.65)	89.26 (89.06-89.47)
ClusterGCN + DeepSet	79.75 (78.75-80.75)	87.2 (87.02-87.38)
ClusterGCN + Set2Set	71.65 (69.89-73.42)	88.34 (87.89-88.79)
ClusterGCN + Set Transformer	81.61 (79.34-83.87)	92.84 (91.95-93.74)
GAT	73.10 (70.93-75.27)	92.25 (91.27-93.24)
GAT + DeepSet	79.45 (77.98-80.92)	75.99 (74.8-77.68)
GAT + Set2Set	82.95 (81.75-84.15)	92.87 (92.62-93.12)
GAT + Set Transformer	<u>89.8</u> (88.89-91.71)	95.12 (94.02-96.22)
GIN + EdgeConv	63.36 (62.53-64.19)	89.56 (88.54-90.58)
EdgeConditionedConvolution	46.15 (34.72-57.59)	88.63 (86.07-91.20)
GraphSAGE	89.51 (88.13-90.47)	<u>96.71</u> (95.61-97.42)
CapsGNN	87.84 (84.87-89.89)	93.86 (91.36-95.33)
GACapNet	**91.97** (90.58-93.45)	**99.24** (98.66-99.46)

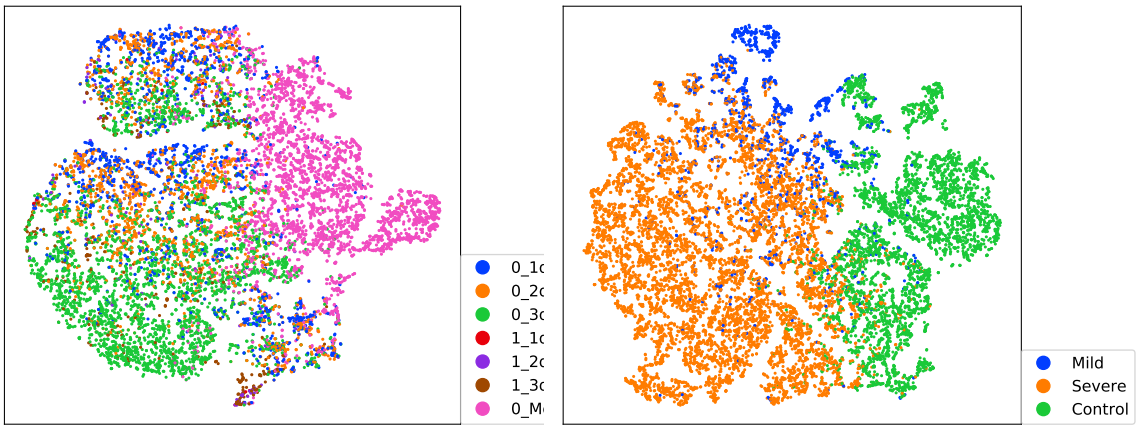
which utilizes GAT and the output of multi-heads attentions as capsules, is significantly more effective than the CapsGNN. In general, these experimental results demonstrate the superiority and robustness of our proposed GACapNet.

5.4.5 Node Visualization

The task of node classification aims to learn good node representations which can be used to distinguish different node types. To understand the results of our GACapNet model more intuitively, we concatenate the output capsules as the vector for each node and apply t-SNE (74) to map the the node vector into a 2-D space for visualization. As illustrated in Figure 5.4, we compare our GACapNet with the GAT model. In Figure 5.4, different color indicates different categories of nodes. For both the SARS-CoV-2 infected organoids and the COVID-19 patients datasets, our GACapNet is able to map the nodes with different labels into different areas. Moreover, the nodes belonging to the same categories are grouped in the same areas. This demonstrates the model ability of label-discrimination.

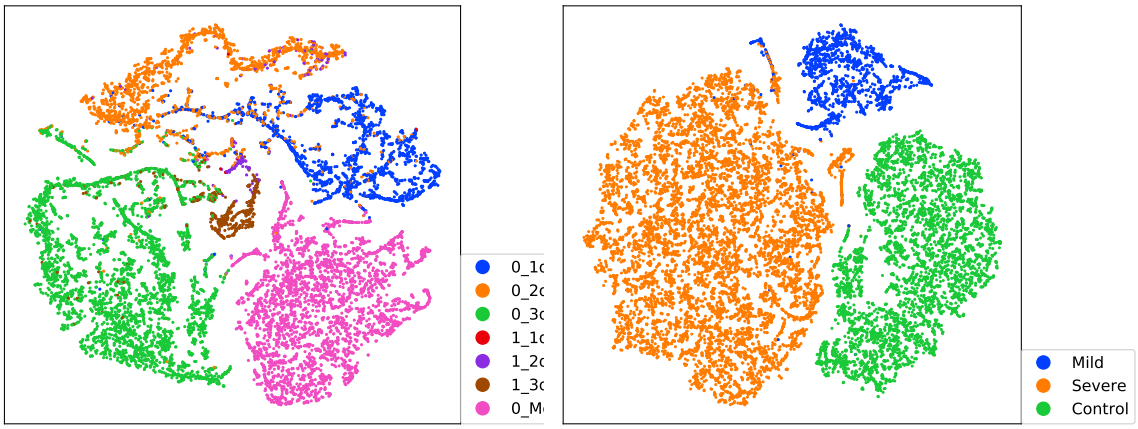
5.4.6 Analysis of Model Convergence

To estimate the convergence of our model on node classification task, we analyse the accuracy score curve with different training epochs. As shown in Figure 5.5,



(a) GAT on SARS-CoV-2 infected organoids

(b) GAT on COVID-19 patients

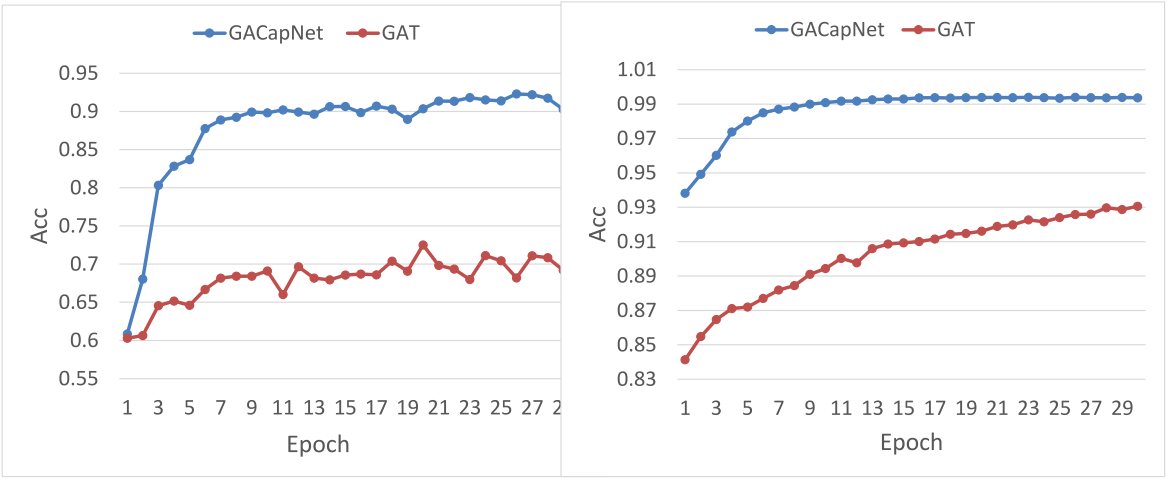


(c) GACapNet on SARS-CoV-2 infected organoids

(d) GACapNet on COVID-19 patients

Figure 5.4: Visualization of node representations learned by the proposed GACapNet vs. GAT model. Different colors correspond to different node classes. 0_1dpi and 1_1dpi denote infected_1dpi or bystander_1dpi, respectively, and other classes in SARS-CoV-2 infected organoids are represented in a similar manner.

we compare our model with a strong baseline GAT model. We can see that our GACapNet quickly achieves good performance within 10 epochs for both of the datasets. After that, the performance gradually stabilizes at a high level of accuracy. Compared to the baseline GAT model, our GACapNet model outperforms the GAT model by large margin along all the training epochs (e.g, more than 20% accuracy score over the GAT model on SARS-CoV-2 infected organoids and more than 6% accuracy score on COVID-19 patients). This experiment clearly shows the superiority of our proposed GACapNet.



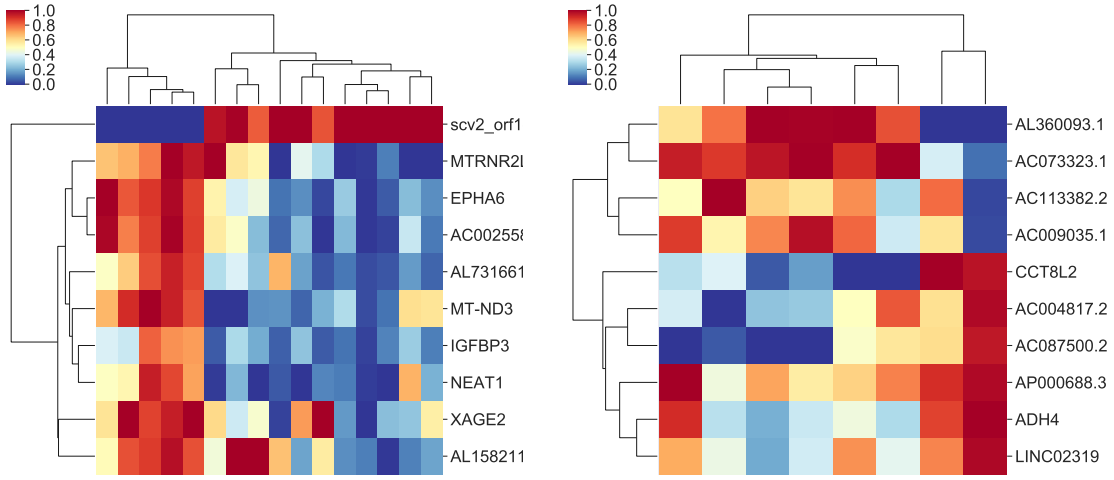
(a) SARS-CoV-2 infected organoids (b) COVID-19 patients

Figure 5.5: Accuracy score versus training epochs.

5.4.7 Analysis of the Gene Features

Figure 5.6 shows top 10 important gene features for each dataset. We extract the weights from the multi-head graph attention layers and average them over the heads as the importance scores. To illustrate the analytical results, we rank all the gene features according the importance scores and select the top 10 genes to present in Figure 5.6.

These 10 genes are ranked as the most important features among the COVID-19 patient profiles from the two selected datasets. Given that the sizes of the datasets are relatively small and the virus keeps evolving, clinical implications of these top ten gene features require further studies and validation. However, this ranked gene feature list can potentially deliver great contributions if integrated well with two other studies: 1) Virology studies: our approach of studying the single-cell RNA sequencing is closest to the study of virology in general. The differences in genetic features of the healthy cohorts and the sick patients can strengthen the understanding of COVID-19 virology and epidemiology from a micro molecular and cellular scope. 2) Clinical studies: together with PCR test reports, blood test results, immunity function report and CT scans, our study is able to assign COVID-19 patients with severity scores and to provide comprehensive guidelines to clinical diagnoses and treatments.



(a) SARS-CoV-2 infected organoids

(b) COVID-19 patients

Figure 5.6: Top 10 important gene features for each dataset, colored by learned weights.

5.5 Discussions and Potential Implications

We hope our study could serve to help contain the deadly spread of the disease, to help improve the efficacy rate of the COVID-19 treatment, and to control the tragic economic and broader societal impacts on the global population to some extent.

Here, we seek to predict SARS-CoV-2 severity level from a machine learning perspective. Our proposed model learns the cellular node features, distinguishes transcriptional signatures of the evolving SARS-CoV-2, and extracts transcriptomic patterns as predictive features. Specifically, we propose a GAT and capsule network architecture to identify the most relevant cells and genes that can determine the infectiousness of SARS-CoV-2. From an economic perspective, effective genes and cells classification could potentially support many downstream applications, such as fast diagnosis of COVID-19 disease, fast track of the SARS-CoV-2 infections mutations, precision treatment of the COVID-19 patients, and robust endorsement of vaccines effectiveness. From a data integration perspective, a well classified collection of genes and cells could facilitate the process of merging, integrating or managing the genetic sequencing data in the near future. We hope the experimental results derived from this work can facilitate future research, clinical validations and investigations in COVID-19 therapeutic properties.

However, we acknowledge that there are still some caveats to our results in the

paper. Even though the experiments have shown significantly better results than the SOTA models in the node classification task, our study still lacks of robust clinical proven theories, large heterogeneous patient samples and strong interpretability to support clinical use. Instead, we could engage these output features as hypotheses into further studies and aim to derive and prove more clinical assumptions. Therefore, we hope researchers and scholars from various domains could work together to conduct more interdisciplinary studies to learn what other factors can possibly cause severe SARS-CoV-2 infection, such as aging or changes in immune systems.

5.6 Conclusion and Future Work

In this chapter, we extend the GAT-based techniques to an ongoing real-life medical problem of COVID-19 disease prediction, and propose the GACapNet model accordingly. Compared to the SOTA methods, our GACapNet, which comprises of a multi-head attention layer and a capsule network layer, has delivered significantly better experimental results on two scRNA-seq datasets. Specifically, the GACapNet extracts more representative and insightful features from nodes and genes and generates predictive SARS-CoV-2 transcriptomic patterns. For the computer science research community, our work demonstrates the capability of GAT-based methods to practise graph learning in medical domain. For the medical background researchers,

our experimental results also indicate potential roles for advanced machine learning techniques to play in closing existing knowledge gaps of the pathogenesis and recovery of COVID-19 pneumonia.

To go one step further, we pose several future directions of research, listed as follows:

- First, we look forward to seeing more high quality COVID-19 single-cell RNA sequencing data to become available to support the SARS-CoV-2 infection prediction research in the near future. Because of the novelty of this COVID-19 pandemic topic and the evolution of the virus, the access to high quality and large quantity of scRNA-seq data is not available yet. Therefore, we only apply two datasets to test our proposed method in this work. With more scRNA-seq datasets become available, we hope to lift our proposed model performance up onto the next level.

- Second, our proposed model is considered just as a starting point of solving the COVID-19 disease prediction problem due to the time restrictions. We have planned further advancement of our GACapNet, which involves incorporating more features, such as cell type and edge features, into our analysis. This improvement of our model also requires more high quality datasets to be available, as mentioned in the previous point.

- Third, we hope to conduct more interdisciplinary research with various COVID-19 related fields to learn how to improve, utilize, employ and deploy our advanced machine learning models more safely and appropriately.

6 Conclusions and Future Work

6.1 Conclusions

In this thesis, we explore various advanced GAT-based solutions to tackle issues in single graph learning, multiple graph learning and ongoing real-world graph learning problems. Specifically, we propose four models to tackle the KGE, CEA and COVID-19 disease predictions respectively. These models include, 1) a GAT-based solution of bidirectional graph attention network (BiGAT), which learns embeddings in a hierarchical neighboring propagation manner, to capture better KGE; 2) a GAT-based solution of Contextual Alignment Enhanced Cross Graph Attention Network (CAECGAT), which jointly learns the embeddings in different KGs by propagating cross-KG information through pre-aligned seed alignments, to obtain more CEA; 3) a GAT-based solution of Dual Gated Graph Attention Network with Dynamic Iterative Training (DuGa-DIT), which learns cross-KG embeddings to bridge the semantic gaps between different KGs by leveraging both neighborhood features and cross-KG

alignment information with dynamic iterative training, to further enhance CEA; and 4) a GAT-based solution of Graph attention capsule network (GACapNet), which extends GATs and Capsule networks to scRNA-seq datasets to predict COVID-19 disease severity.

Through extensive experiments, we demonstrate that these proposed models significantly outperform the baselines across various evaluation metrics. By comparison, we conclude that for the KGE task, the proposed BiGAT achieves the best performance compared to the SOTA approaches on four KGE benchmark datasets. Its superiority lies not only in the model architecture but also its effectiveness and efficiency. For the CEA task, the proposed CAECGAT model is at least comparable to, if not better than, the SOTA baseline models with optimal parameter settings. The DuGa-DIT model is more promising and robust among two thirds of the evaluation metrics than all baseline models. Moreover, it addresses performance restraints of the SOTA models using a unified model framework. Besides, its capability of capturing both neighboring and cross-KG alignment features send this model to the top of the performance ranking list. For the COVID-19 disease predictions, the proposed GACapNet model has delivered significantly better experimental results on two scRNA-seq datasets in the cell node classification task. We also prove that the model architecture indicates that both the proposed approaches and the parameters

are robust and optimal.

Based on the analyses above, the implications of this thesis can thus be ascribed to three folds, stated as follows:

- BiGAT serves as an advanced model to apply GAT (170) architecture to single graph learning on KGE tasks. It not only tackles the limitations of existing methods that learn embeddings from unilateral direction of the neighborhoods, but also leverages the GAT mechanism to capture multi-hop features in KGs and to avoid early-stage information loss. The respective section of the thesis provides a viable solution to address existing problems in the KGE research field.

- Both the CAECGAT and the DuGa-DIT models serve as advanced GAT-based models to address multiple graph learning on CEA tasks. Both proposed methods aim to make full use of the seed alignments among two input graphs to alleviate the semantic gaps between different KGs. As a preliminary study, CAECGAT solves a prevalent problem in conventional GNN-based methods, where they tend to regard all pre-aligned entity pairs as objective seed alignments for training while underestimating the contextual aligned information for cross KG aggregation. Contrarily, our CAECGAT chooses one batch as objective seed alignments and the rest as contextual seed alignments to train the

model iteratively. With this setting, the model learns better cross-KG embeddings by propagating contextual seed alignments across different KGs. Based on the prior work and our preliminary study, DuGa-DIT builds an enhanced GAT-based model, which optimizes CAECGAT in an unified framework. It manages to make full use of the pre-aligned entity pairs in both training and prediction stages. Furthermore, with the dynamic iterative training process, we can dynamically update the cross-KG attention score matrices. This allows more cross-KG information to transfer across different KGs and to achieve better overall performances. As a result, these two proposed models unfold more promising solutions to tackle the CEA task.

- The GACapNet model demonstrates the applicability of GAT-based models in the medical domain. It is able to extract more representative and insightful gene features and to generate predictive SARS-CoV-2 transcriptomic patterns that are linked to the infectiousness and severity predictions of the COVID-19.

6.2 Future Work

Last but not least, we would like to provide some possible directions of future work to conclude this thesis. In this study, we mainly focus on deep learning approaches to address all relevant challenges and problems on single graph learning, multiple graph

learning and real-world graph learning. One simple and general direction of future work could be to involve some non-deep learning methods in experiments for model comparisons. This would allow us to discuss the advantages and disadvantages of various models, and to address significant topics in real-world application, such as model explanability, the number of available training examples and the respective computation costs. The proposed methods will also be applied and evaluated on more data sets for more applications, such as biomedicine, Clinical and Chinese IR, in the near future (53, 54, 55, 90, 91, 223, 237, 238, 239).

For GAT-based models on single graph learning via KGE task, we suggest the future work could be done in mainly two aspects. First, on the micro level, we could try to incorporate more text information that are embedded in the KGs to further enhance the model performance. We could also expand our KGE solutions towards bringing in new advanced machine learning concepts and building more hybrid architectures with attention mechanisms. Second, on the macro level, we believe researchers could apply these existing models to more NLP downstream applications, such as question answering over incomplete KGs, personalised recommendations, etc.. We could also validate our models on a number of real-life KGs and use cases, such as predicting missing links among biomedical protein network.

For GAT-based models on multiple graph learning via CEA task, we suggest the

future work could be done in mainly six aspects. First, we can extend our CEA to three or more KGs. Even though it seems straightforward to increase the number of KGs, the difficulty of aligning precise entities across various KGs could grow exponentially. For initial trials, we should try to apply our existing models to three KGs for the entity alignments task and to optimize them accordingly. However, if the experimental results appear to be sub-optimal, we should consider tailor-made models for different number of KGs and solve respective issues as the number of KGs increases. Second, we should consider involving more entity information, such as entity properties and descriptions, in the learning process. On one hand, as more information being incorporated in the algorithms, the accuracy should improve significantly. On the other hand, we should also consider the re-applicability of these models to other KGs, including those open-sourced ones and domain specific ones. Third, we realize that practising CEA task with our current solutions can be costly and not sustainable in long term. In many real life scenarios, the size of KGs expands exponentially everyday. Thus, it would eventually become too large to train, especially for tasks in personalised search and recommendations. Given the structural differences and inherited variances, KGs can be even bigger. Therefore, designing and developing distributed auto alignment methods and auto learning approaches would become much more important for future research. We hope to see our future

solutions to become more applicable, effective and cost-efficient. Fourth, as our research domain continues expanding, our existed models need to go beyond dominant languages of English, Chinese, French and Japanese. This would require more high quality benchmark datasets to be constructed and made available to the public. For example, as the e-commerce platforms become more trendy and lively in South East Asia (SEA), many tech giants have been eyeing the SEA market to expand their business. However, datasets on key regional languages such as Indonesian, Thai, Malay and Vietnamese are still largely under-developed compared to dominant languages used in our thesis. It could be challenging for them to support effective downstream tasks such as product recommendations. Fifth, the input graphs involved in this thesis contain quite a number of equivalent surface form entities across KGs of different languages. It would be worth running experiments and designing algorithms to test model performances when all these information given are removed. Lastly, while developing new solutions, we should not stop keeping an eye on our published solutions and keeping them up to dated. We should continuously apply and test them to different datasets, industry domains and real life scenarios, which might derive unexpected insights and results.

For GAT-based models on the COVID-19 disease severity predictions, it is still a relatively young and underdeveloped research domain due to its novelty, interdisci-

plinary characteristics and evolving situations. As a fundamental building block, we look forward to seeing more high quality COVID-19 single-cell RNA sequencing data and other COVID-19 related datasets to become available in the near future. As different variants of COVID-19 evolve and spread around the world, more COVID-19 related questions have been and will be continuously generated. We hope to conduct more interdisciplinary research with various COVID-19 related fields to learn how to improve and to utilize our advanced machine learning models more safely and appropriately.

Bibliography

[1] Abbas, A., Abdelsamea, M. M., and Gaber, M. M. (2020). Classification of COVID-19 in chest x-ray images using detrac deep convolutional neural network. *CoRR*, abs/2003.13815.

[2] Aizawa, A., Bergeron, F., Chen, J., Cheng, F., Hayashi, K., Inui, K., Ito, H., Kawahara, D., Kitsuregawa, M., Kiyomaru, H., Kobayashi, M., Kodama, T., Kurohashi, S., Liu, Q., Matsubara, M., Miyao, Y., Morishima, A., Murawaki, Y., Omura, K., Song, H., Sumita, E., Suzuki, S., Tanaka, R., Tanaka, Y., Toyoda, M., Ueda, N., Ueoka, H., Utiyama, M., and Zhong, Y. (2020). A system for worldwide covid-19 information aggregation. In *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*.

[3] Akrami, F., Saeef, M. S., Zhang, Q., Hu, W., and Li, C. (2020). Realistic re-evaluation of knowledge graph completion methods: An experimental study. In *Proceedings of the 2020 International Conference on Management of Data, SIG-*

MOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020, pages 1995–2010.

[4] Anastasopoulos, A., Cattelan, A., Dou, Z.-Y., Federico, M., Federmann, C., Genzel, D., Guzmán, F., Hu, J., Hughes, M., Koehn, P., Lazar, R., Lewis, W., Neubig, G., Niu, M., Öktem, A., Paquin, E., Tang, G., and Tur, S. (2020). Tico-19: the translation initiative for covid-19. In *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*.

[5] Atwood, J. and Towsley, D. (2016). Diffusion-convolutional neural networks. In *30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.*

[6] Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. G. (2007). Dbpedia: A nucleus for a web of open data. In *The Semantic Web, pages 722–735*.

[7] Balazevic, I., Allen, C., and Hospedales, T. M. (2019a). Multi-relational poincaré graph embeddings. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 4465–4475.

[8] Balazevic, I., Allen, C., and Hospedales, T. M. (2019b). Tucker: Tensor factor-

ization for knowledge graph completion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5184–5193.

[9] Bansal, T., Juan, D., Ravi, S., and McCallum, A. (2019). A2N: attending to neighbors for knowledge graph inference. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4387–4392.

[10] Bellomarini, L., Benedetti, M., Gentili, A., Laurendi, R., Magnanimi, D., Muci, A., and Sallinger, E. (2020). COVID-19 and company knowledge graphs: Assessing golden powers and economic impact of selective lockdown via AI reasoning. *CoRR*, abs/2004.10119.

[11] Bianchi, F. M., Grattarola, D., Livi, L., and Alippi, C. (2020). Hierarchical representation learning in graph neural networks with node decimation pooling. *IEEE Transactions on Neural Networks and Learning Systems (Early Access)*, pages 1–13.

[12] Bojchevski, A., Shchur, O., Zugner, D., and Gunnemann, S. (2018). Netgan:

Generating graphs via random walks. In *ICML '18: Proceedings of the 35th International Conference on Machine Learning*.

[13] Bollacker, K. D., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the SIGMOD, pages 1247–1250*.

[14] Bordes, A., Glorot, X., Weston, J., and Bengio, Y. (2014). A semantic matching energy function for learning with multi-relational data - application to word-sense disambiguation. *Mach. Learn.*, 94(2):233–259.

[15] Bordes, A., Usunier, N., García-Durán, A., Weston, J., and Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795.

[16] Cai, L. and Wang, W. Y. (2018). KBGAN: adversarial learning for knowledge graph embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1470–1480.

[17] Cao, Y., Liu, Z., Li, C., Liu, Z., Li, J., and Chua, T. (2019). Multi-channel graph neural network for entity alignment. In *Proceedings of the ACL, pages 1452–1461.*

[18] Chami, I., Wolf, A., Juan, D., Sala, F., Ravi, S., and Ré, C. (2020). Low-dimensional hyperbolic knowledge graph embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6901–6914.

[19] Chen, B., Zhang, J., Tang, X., Chen, H., and Li, C. (2020). Jarka: Modeling attribute interactions for cross-lingual knowledge alignment. In *Proceedings of the PAKDD, pages 845–856.*

[20] Chen, D., Socher, R., Manning, C. D., and Ng, A. Y. (2013). Learning new facts from knowledge bases with neural tensor networks and semantic word vectors. In *arXiv '13: arXiv.*

[21] Chen, M., Tian, Y., Chang, K., Skiena, S., and Zaniolo, C. (2018). Co-training embeddings of knowledge graphs and entity descriptions for cross-lingual entity alignment. In *Proceedings of the IJCAI, pages 3998–4004.*

[22] Chen, M., Tian, Y., Yang, M., and Zaniolo, C. (2017). Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In *Proceedings of IJCAI, pages 1511–1517.*

[23] Chen, Y., Wu, L., and Zaki, M. J. (2019). Bidirectional attentive memory networks for question answering over knowledge bases. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2913–2923.

[24] Chiang, W., Liu, X., Si, S., Li, Y., Bengio, S., and Hsieh, C. (2019). Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 257–266.

[25] Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Deep Learning and Representation Learning Workshop*.

[26] Clouâtre, L., Trempe, P., Zouaq, A., and Chandar, S. (2020). MLMLM: link prediction with mean likelihood masked language model. *CoRR*, abs/2009.07058.

[27] Dettmers, T., Minervini, P., Stenetorp, P., and Riedel, S. (2018). Convolutional 2d knowledge graph embeddings. In *Proceedings of the AAAI, pages 1811–1818*.

[28] Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2019). BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT, pages 4171–4186*.

[29] Du, X., Wang, X., He, X., Li, Z., Tang, J., and Chua, T. (2020). How to learn item representation for cold-start multimedia recommendation? In *Proceedings of the MM, pages 3469–3477*.

[30] Ebisu, T. and Ichise, R. (2018). Toruse: Knowledge graph embedding on a lie group. In McIlraith, S. A. and Weinberger, K. Q., editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1819–1826. AAAI Press.

[31] Ebisu, T. and Ichise, R. (2019). Graph pattern entity ranking model for knowledge graph completion. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 988–997.

[32] Feng, J., Huang, M., Wang, M., Zhou, M., Hao, Y., and Zhu, X. (2016). Knowl-

edge graph embedding by flexible translation. In Baral, C., Delgrande, J. P., and Wolter, F., editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR 2016, Cape Town, South Africa, April 25-29, 2016*, pages 557–560. AAAI Press.

[33] Fout, A., Byrd, J., Shariat, B., and Ben-Hur, A. (2017). Protein interface prediction using graph convolutional networks. In *31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA*.

[34] Färber, M. (2019). The microsoft academic knowledge graph: A linked data source with 8 billion triples of scholarly data. *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference*, pages 113 – 129.

[35] Galassi, A., Lippi, M., and Torroni, P. (2020). Attention in natural language processing. *IEEE Transactions on Neural Networks and Learning Systems (Early Access)*, pages 1–18.

[36] Garcia, V. and Bruna, J. (2017). Few-shot learning with graph neural networks. *arXiv*.

[37] García-Durán, A., Bordes, A., and Usunier, N. (2015). Composing relationships with translations. In *Proceedings of EMNLP, pages 286–290*. The Association for Computational Linguistics.

[38] Garc´ıa-Duran, A. and Niepert, M. (2018). Kblrn: End-to-end learning of knowledge base representations with latent, relational, and numerical features. In *arXiv*.

[39] Gesese, G. A., Biswas, R., and Sack, H. (2019). A comprehensive survey of knowledge graph embeddings with literals: Techniques and applications. In Alam, M., Buscaldi, D., Cochez, M., Osborne, F., Recupero, D. R., and Sack, H., editors, *Proceedings of the Workshop on Deep Learning for Knowledge Graphs (DL4KG2019) Co-located with the 16th Extended Semantic Web Conference 2019 (ESWC 2019), Portoroz, Slovenia, June 2, 2019*, volume 2377 of *CEUR Workshop Proceedings*, pages 31–40. CEUR-WS.org.

[40] Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. In *Proceedings of the AISTATS, pages 315–323*.

[41] Gonçalves, R. S., Horridge, M., Li, R., Liu, Y., Musen, M. A., Nyulas, C. I., Obamos, E., Shrouty, D., and Temple, D. (2019). Use of owl and semantic web technologies at pinterest. *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference*, pages 418–435.

[42] Gozes, O., Frid-Adar, M., Greenspan, H., Browning, P. D., Zhang, H., Ji, W., Bernheim, A., and Siegel, E. (2020). Rapid AI development cycle for the coron-

avirus (COVID-19) pandemic: Initial results for automated detection & patient monitoring using deep learning CT image analysis. *CoRR*, abs/2003.05037.

[43] Grotheer, R., Huang, L., Huang, Y., Kryshchenko, A., Kryshchenko, O., Li, P., Li, X., Rebrova, E., Ha, K., and Needell, D. (2020). Covid-19 literature topic-based search via hierarchical nmf. In *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*.

[44] Guo, L., Sun, Z., and Hu, W. (2019a). Learning to exploit long-term relational dependencies in knowledge graphs. In *Proceedings of ICML, pages 2505–2514*.

[45] Guo, Z., Zhang, Y., Teng, Z., and Lu, W. (2019b). Densely connected graph convolutional networks for graph-to-sequence learning. *Trans. Assoc. Comput. Linguistics*.

[46] Hamaguchi, T., Oiwa, H., Shimbo, M., and Matsumoto, Y. (2017). Knowledge transfer for out-of-knowledge-base entities: A graph neural network approach. In *IJCAI '17: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 1802–1808.

[47] Hamilton, W. L., Ying, Z., and Leskovec, J. (2017). Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*

30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pages 1024–1034.

[48] Hao, Y., Zhang, Y., He, S., Liu, K., and Zhao, J. (2016). A joint embedding method for entity alignment of knowledge bases. In *Proceedings of CCKS*, volume 650, pages 3–14.

[49] He, Q., Chen, B.-C., and Agarwal, D. (2016). Building the linkedin knowledge graph. *LinkedIn Blog*.

[50] He, S., Liu, K., Ji, G., and Zhao, J. (2015). Learning to represent knowledge graphs with gaussian embedding. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, pages 623–632.

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

[52] Hu, W., Liu, B., Gomes, J., Zitnik, M., Liang, P., Pande, V. S., and Leskovec, J. (2020). Strategies for pre-training graph neural networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

[53] Huang, X. and Hu, Q. (2009). A bayesian learning approach to promoting diversity in ranking for biomedical information retrieval. In Allan, J., Aslam, J. A., Sanderson, M., Zhai, C., and Zobel, J., editors, *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, Boston, MA, USA, July 19-23, 2009*, pages 307–314. ACM.

[54] Huang, X., Peng, F., Schuurmans, D., Cercone, N., and Robertson, S. E. (2003). Applying machine learning to text segmentation for information retrieval. *Inf. Retr.*, 6(3-4):333–362.

[55] Huang, X., Zhong, M., and Si, L. (2005). York university at TREC 2005: Genomics track. In Voorhees, E. M. and Buckland, L. P., editors, *Proceedings of the Fourteenth Text REtrieval Conference, TREC 2005, Gaithersburg, Maryland, USA, November 15-18, 2005*, volume 500-266 of *NIST Special Publication*. National Institute of Standards and Technology (NIST).

[56] Hwang, B., Lee, J. H., and Bang, D. (2018). Single-cell rna sequencing technologies and bioinformatics pipelines. *Experimental Molecular Medicine*, 50, 96.

[57] Iana, A., Jung, S., Naeser, P., Birukou, A., Hertling, S., and Paulheim, H. (2019). Building a conference recommender system based on scigraph and wikicfp.

Semantic Systems. The Power of AI and Knowledge Graphs - 15th International Conference, SEMANTiCS 2019, pages 117 – 123.

[58] Jenatton, R., Roux, N. L., Bordes, A., and Obozinski, G. R. (2012). A latent factor model for highly multi-relational data. In *Proceedings of NIPS, pages 3176–3184*.

[59] Ji, G., He, S., Xu, L., Liu, K., and Zhao, J. (2015). Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 687–696.

[60] Ji, G., Liu, K., He, S., and Zhao, J. (2016). Knowledge graph completion with adaptive sparse transfer matrix. In *Proceedings of AAAI, pages 985–991*.

[61] Ji, S., Pan, S., Cambria, E., Marttinen, P., and Yu, P. S. (2020). A survey on knowledge graphs: Representation, acquisition and applications. *CoRR*, abs/2002.00388.

[62] Jin, H., Li, C., Zhang, J., Hou, L., Li, J., and Zhang, P. (2019). XLORE2: large-

scale cross-lingual knowledge graph construction and application. *Data Intell.*, 1(1):77–98.

[63] Kazemi, S. M. and Poole, D. (2018). Simple embedding for link prediction in knowledge graphs. In *Proceedings of NeurIPS, pages 4289–4300.*

[64] Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In *Proceedings of the ICLR.*

[65] Kipf, T. N. and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *Proceedings of ICLR.*

[66] Klein, P., Ponzetto, S. P., and Glavaš, G. (2017). Improving neural knowledge base completion with cross-lingual projections. In *Proceedings of EACL, pages 516–522.*

[67] Krishnan, A. (2018). Making search easier: How amazon's product graph is helping customers find products more easily. *Amazon Blog.*

[68] Kristiadi, A., Khan, M. A., Lukovnikov, D., Lehmann, J., and Fischer, A. (2019). Incorporating literals into knowledge graph embeddings. In *arXiv.*

[69] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classifica-

tion with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25 (NIPS 2012)*.

[70] Krompab, D., Baier, S., and Tresp, V. (2015). Type-constrained representation learning in knowledge graphs. In *Proceedings of ISWC, pages 640–655*.

[71] Lacroix, T., Usunier, N., and Obozinski, G. (2018). Canonical tensor decomposition for knowledge base completion. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 2869–2878.

[Lally et al.] Lally, A., Bagchi, S., Barborak, M. A., Buchanan, D. W., Chu-Carroll, J., Ferrucci, D. A., Glass, M. R., Kalyanpur, A., Mueller, E. T., Murdock, J. W., Patwardhan, S., and M.Prager, J. Watsonpaths: Scenario-based question answering and inference over unstructured information. *AI Magazine*, 38.

[73] Lan, Y., Wang, S., and Jiang, J. (2019). Knowledge base question answering with a matching-aggregation model and question-specific contextual relations. *IEEE ACM Trans. Audio Speech Lang. Process.*, 27(10):1629–1638.

[74] Laurens, V. D. M. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2605):2579–2605.

[75] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. In *IEEE*.

[76] Lee, C.-W., Fang, W., Yeh, C.-K., and Wang, Y.-C. F. (2018). Multi-label zero-shot learning with structured knowledge graphs. In *Proceedings of the 2018 Conference on Computer Vision and Pattern Recognition*.

[77] Lee, J., Lee, Y., Kim, J., Kosiorek, A. R., Choi, S., and Teh, Y. W. (2019). Set transformer: A framework for attention-based permutation-invariant neural networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pages 3744–3753.

[78] Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., and Bizer, C. (2015). Dbpedia - A large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

[79] Li, C., Cao, Y., Hou, L., Shi, J., Li, J., and Chua, T. (2019). Semi-supervised entity alignment via joint knowledge embedding model and cross-graph model. In *Proceedings of EMNLP-IJCNLP, pages 2723–2732*.

[80] Li, J., Tang, J., Li, Y., and Luo, Q. (2009). Rimom: A dynamic multistrat-

egy ontology alignment framework. *IEEE Transactions on Knowledge and Data Engineering, 21:1218–1232, 2009*, 21(8):1218–1232.

[81] Liao, M., Liu, Y., Yuan, J., Wen, Y., and Zhang, Z. (2020). Single-cell landscape of bronchoalveolar immune cells in patients with covid-19. *Nature medicine*, 26(6).

[82] Lin, X., Yang, H., Wu, J., Zhou, C., and Wang, B. (2019). Guiding cross-lingual entity alignment via adversarial knowledge embedding. In *Proceedings of ICDM, pages 429–438*.

[83] Lin, X. V., Socher, R., and Xiong, C. (2018a). Multi-hop knowledge graph reasoning with reward shaping. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 3243–3253.

[84] Lin, Y., Han, X., Xie, R., Liu, Z., and Sun, M. (2018b). Knowledge representation learning: A quantitative review. *CoRR*, abs/1812.10901.

[85] Lin, Y., Liu, Z., Luan, H., Sun, M., Rao, S., and Liu, S. (2015a). Modeling relation paths for representation learning of knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 705–714.

[86] Lin, Y., Liu, Z., Luan, H.-B., Sun, M., Rao, S., and Liu, S. (2015b). Modeling relation paths for representation learning of knowledge bases. In *Proceedings of EMNLP, pages 705–714*.

[87] Lin, Y., Liu, Z., and Sun, M. (2016). Knowledge representation learning with entities, attributes and relations. In *Proceedings of IJCAI, pages 2866–2872*.

[88] Lin, Y., Liu, Z., Sun, M., Liu, Y., and Zhu, X. (2015c). Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 2181–2187.

[89] Liu, H., Wu, Y., and Yang, Y. (2017). Analogical inference for multi-relational embeddings. In Precup, D. and Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 2168–2178. PMLR.

[90] Liu, Y., An, A., and Huang, X. (2006). Boosting prediction accuracy on imbalanced datasets with SVM ensembles. In Ng, W. K., Kitsuregawa, M., Li, J., and Chang, K., editors, *Advances in Knowledge Discovery and Data Mining, 10th*

Pacific-Asia Conference, PAKDD 2006, Singapore, April 9-12, 2006, Proceedings, volume 3918 of *Lecture Notes in Computer Science*, pages 107–118. Springer.

[91] Liu, Y., Huang, X., An, A., and Yu, X. (2007). ARSA: a sentiment-aware model for predicting sales performance using blogs. In Kraaij, W., de Vries, A. P., Clarke, C. L. A., Fuhr, N., and Kando, N., editors, *SIGIR 2007: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Amsterdam, The Netherlands, July 23-27, 2007*, pages 607–614. ACM.

[92] Liu, Z., Cao, Y., Pan, L., Li, J., and Chua, T. (2020). Exploring and evaluating attributes, values, and structures for entity alignment. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6355–6364.

[93] Maas, L, A., Hannun, A. Y., and Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. In *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*.

[94] Madotto, A., Wu, C., and Fung, P. (2018). Mem2seq: Effectively incorporating knowledge bases into end-to-end task-oriented dialog systems. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL*

2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers, pages 1468–1478.

[95] Mahdisoltani, F., Biega, J., and Suchanek, F. (2015). Yago3: A knowledge base from multilingual wikipedias. In *Proceedings of CIDR*.

[96] Mao, X., Wang, W., Xu, H., Lan, M., and Wu, Y. (2020). MRAEA: an efficient and robust entity alignment approach for cross-lingual knowledge graph. In *Proceedings of WSDM, pages 420–428*.

[97] McCallum, A. and Wellner, B. (2004). Conditional models of identity uncertainty with application to noun coreference. In *Proceedings of NIPS, pages 905–912*.

[98] Meng, L., Feng, F., He, X., Gao, X., and Chua, T. (2020). Heterogeneous fusion of semantic and collaborative information for visually-aware food recommendation. In *Proceedings of the MM, pages 3460–3468*.

[99] Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26 (NIPS 2013)*.

[100] Miller, G. A. (1995). Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41.

[101] Mitchell, T. M., Cohen, W. W., Jr., E. R. H., Talukdar, P. P., Yang, B., Betteridge, J., Carlson, A., Mishra, B. D., Gardner, M., Kisiel, B., Krishnamurthy, J., Lao, N., Mazaitis, K., Mohamed, T., Nakashole, N., Platanios, E. A., Ritter, A., Samadi, M., Settles, B., Wang, R. C., Wijaya, D., Gupta, A., Chen, X., Saparov, A., Greaves, M., and Welling, J. (2018). Never-ending learning. *Communications of the ACM*, 61:103–115.

[102] Miwa, M. and Bansal, M. (2016). End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

[103] Narin, A., Kaya, C., and Pamuk, Z. (2020). Automatic detection of coronavirus disease (COVID-19) using x-ray images and deep convolutional neural networks. *CoRR*, abs/2003.10849.

[104] Nathani, D., Chauhan, J., Sharma, C., and Kaul, M. (2019). Learning attention-based embeddings for relation prediction in knowledge graphs. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4710–4723.

[105] Navigli, R. and Ponzetto, S. P. (2010). Babelnet: building a very large mul-

tilingual semantic network. *ACL '10: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 216 – 225.

[106] Nguyen, D. Q., Nguyen, T. D., Nguyen, D. Q., and Phung, D. (2018). A novel embedding model for knowledge base completion based on convolutional neural network. In *Proceedings of NAACL, pages 327–333*.

[107] Nguyen, D. Q., Sirts, K., Qu, L., and Johnson, M. (2016). Neighborhood mixture model for knowledge base completion. In *Proceedings of SIGNLL, pages 40–50*.

[108] Nguyen, D. Q., Vu, T., Nguyen, T. D., Nguyen, D. Q., and Phung, D. Q. (2019). A capsule network-based embedding model for knowledge graph completion and search personalization. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2180–2189.

[109] Nickel, M., Rosasco, L., and Poggio, T. (2016a). Holographic embeddings of knowledge graphs. In *Proceedings of AAAI, pages 1955–1961*.

[110] Nickel, M., Rosasco, L., and Poggio, T. A. (2016b). Holographic embeddings of knowledge graphs. In Schuurmans, D. and Wellman, M. P., editors, *Proceedings*

of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA, pages 1955–1961. AAAI Press.

[111] Nickel, M., Tresp, V., and Kriegel, H. (2011). A three-way model for collective learning on multi-relational data. In Getoor, L. and Scheffer, T., editors, *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 809–816. Omnipress.

[112] Nie, H., Han, X., Sun, L., Wong, C. M., Chen, Q., Wu, S., and Zhang, W. (2020). Global structure and local semantics-preserved embeddings for entity alignment. In *Proceedings of the IJCAI, pages 3658–3664*.

[113] Niu, G., Li, B., Zhang, Y., Pu, S., and Li, J. (2020). Autoeter: Automated entity type representation for knowledge graph embedding. *CoRR*, abs/2009.12030.

[114] Nowak, A., Villar, S., Bandeira, A. S., and Bruna, J. (2018). Revised note on learning quadratic assignment with graph neural networks. In *2018 IEEE Data Science Workshop, DSW 2018 - Proceedings*.

[115] Noy, N. F., Gao, Y., Jain, A., Narayanan, A., Patterson, A., and Taylor, J. (2019). Industry-scale knowledge graphs: Lessons and challenges. *ACM Queue 17*, page 20.

[116] Oh, B., Seo, S., and Lee, K. (2018). Knowledge graph completion by context-aware convolutional learning with multi-hop neighborhoods. In *Proceedings of CIKM, pages 257–266*.

[117] Pei, S., Yu, L., Hoehndorf, R., and Zhang, X. (2019a). Semi-supervised entity alignment via knowledge graph embedding with awareness of degree difference. In *Proceedings of WWW, pages 3130–3136*.

[118] Pei, S., Yu, L., Yu, G., and Zhang, X. (2020). REA: robust cross-lingual entity alignment between knowledge graphs. In *Proceedings of the KDD, pages 2175–2184*.

[119] Pei, S., Yu, L., and Zhang, X. (2019b). Improving cross-lingual entity alignment via optimal transport. In *Proceedings of the IJCAI, pages 3231–3237*.

[120] Peng, H., Li, J., He, Y., Liu, Y., Bao, M., Wang, L., Song, Y., and Yang, Q. (2018). Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In *WWW '18: Proceedings of the 2018 World Wide Web ConferenceApril 2018*.

[121] Peroni, S., Shotton, D. M., and Vitali, F. (2017). One year of the opencitations corpus – releasing rdf-based scholarly citation data into the public domain. *The*

Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, pages 184–192.

[122] Pinchin, V. (2016). I'm feeling yucky :(searching for symptoms on google. *Google Blogs*.

[123] Pittman, R. J., Srivastava, A., Hewavitharana, S., Kale, A., and Mansour, S. (2017). Cracking the code on conversational commerce. *eBay Blog*.

[124] Polanski, K., Young, M. D., Miao, Z., Meyer, K. B., Teichmann, S. A., and Park, J.-E. (2019). Bbknn: fast batch alignment of single cell transcriptomes. *Bioinformatics*, 36(3):964–965.

[125] Qian, W., Fu, C., Zhu, Y., Cai, D., and He, X. (2018). Translating embeddings for knowledge graph completion with relation attention mechanism. In Lang, J., editor, *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4286–4292. ijcai.org.

[126] Qu, M. and Tang, J. (2019). Probabilistic logic neural networks for reasoning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 7710–7720.

[127] Ramaswami, P. (2015). A remedy for your health-related questions: health info in the knowledge graph. *Google Blogs*.

[128] Ravindra, N., Sehanobish, A., Pappalardo, J. L., Hafler, D. A., and Dijk, D. V. (2020a). Disease state prediction from single-cell data using graph attention networks. In *CHIL '20: Proceedings of the ACM Conference on Health, Inference, and Learning*, pages 121–130.

[129] Ravindra, N. G., Alfajaro, M. M., Gasque, V., Wei, J., Filler, R. B., Huston, N. C., Wan, H., Szigeti-Buck, K., Wang, B., Montgomery, R. R., et al. (2020b). Single-cell longitudinal analysis of sars-cov-2 infection in human bronchial epithelial cells. *bioRxiv*.

[130] Rebele, T., Suchanek, F., Hoffart, J., Biega, J., Kuzey, E., and Weikum, G. (2016a). Yago: A multilingual knowledge base from wikipedia, wordnet, and geonames. *The Semantic Web - ISWC 2016*, pages 177–185.

[131] Rebele, T., Suchanek, F. M., Hoffart, J., Biega, J., Kuzey, E., and Weikum, G. (2016b). YAGO: A multilingual knowledge base from wikipedia, wordnet, and geonames. In *Proceedings of ISWC, pages 177–185*.

[132] Rhee, S., Seo, S., and Kim, S. (2018). Hybrid approach of relation network and localized graph convolutional filtering for breast cancer subtype classification.

In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)*.

[133] Roberts, K., Alam, T., Bedrick, S., Demner-Fushman, D., Lo, K., Soboroff, I., Voorhees, E., Wang, L. L., and Hersh, W. R. (2020). Trec-covid: Rationale and structure of an information retrieval shared task for covid-19. *Journal of the American Medical Informatics Association*, ocaa091.

[134] Ruffinelli, D., Broscheit, S., and Gemulla, R. (2020). You CAN teach an old dog new tricks! on training knowledge graph embeddings. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

[135] Sabour, S., Frosst, N., and Hinton, G. E. (2017). Dynamic routing between capsules. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3856–3866.

[136] Satija, R., Farrell, J. A., Gennert, D., Schier, A. F., and Regev, A. (2015). Spatial reconstruction of single-cell gene expression data. *Nature Biotechnology*, 33(5).

[137] Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G.

(2009). The graph neural network model. *IEEE Transactions on Neural Networks*, 20:61–80.

[138] Schlichtkrull, M. S., Kipf, T. N., Bloem, P., van den Berg, R., Titov, I., and Welling, M. (2018). Modeling relational data with graph convolutional networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, pages 593–607.

[139] Sehanobish, A., Ravindra, N. G., and van Dijk, D. (2020). Gaining insight into sars-cov-2 infection and COVID-19 severity using self-supervised edge features and graph neural networks. *CoRR*, abs/2006.12971.

[140] Shang, C., Tang, Y., Huang, J., Bi, J., He, X., and Zhou, B. (2019). End-to-end structure-aware convolutional networks for knowledge base completion. In *Proceedings of AAAI, pages 3060–3067*.

[141] Shen, I., Zhang, L., Lian, J., Wu, C., González-Fierro, M., Argyriou, A., and Wu, T. (2020). In search for a cure: Recommendation with knowledge graph on CORD-19. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 3519–3520.

[142] Sheng, M., Wang, J., Zhang, Y., Li, X., Li, C., Xing, C., Li, Q., Shao, Y., and

Zhang, H. (2019). Dockg: A knowledge graph framework for health with doctor-in-the-loop. *International Conference on Health Information Science 2019*, pages 3–14.

[143] Shervashidze, N., Schweitzer, P., van Leeuwen, E. J., Mehlhorn, K., and Borgwardt, K. M. (2011). Weisfeiler-lehman graph kernels. In *Journal of Machine Learning Research 12 (2011) 2539-2561.*

[144] Shervashidze, N., Vishwanathan, S., Petri, T. H., Mehlhorn, K., and Borgwardt, K. M. (2009). Efficient graphlet kernels for large graph comparison. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS) 2009, Clearwater Beach, Florida, USA. Volume 5 of JMLR: WCP 5. Copyright 2009.*

[145] Shi, B. and Weninger, T. (2017). Proje: Embedding projection for knowledge graph completion. In *Proceedings of AAAI, pages 1236–1242.*

[146] Shi, X. and Xiao, Y. (2019). Modeling multi-mapping relations for precise cross-lingual entity alignment. In *Proceedings of the EMNLP, pages 813–822.*

[147] Simonovsky, M. and Komodakis, N. (2017). Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *2017 IEEE Conference on Computer*

Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, pages 29–38.

[148] Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *ICLR 2015*.

[149] Singhal, A. (2012). Introducing the knowledge graph: things, not strings. *Google Blog*.

[150] Socher, R., Chen, D., Manning, C. D., and Ng, A. Y. (2013). Reasoning with neural tensor networks for knowledge base completion. In *Proceedings of NIPS, pages 926–934*.

[151] Sondhi, P., Sun, J., Tong, H., and Zhai, C. (2012). Sympgraph: A framework for mining clinical notes through symptom relation graphs. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1167–1175.

[152] Song, L., Zhang, Y., Wang, Z., and Gildea, D. (2018a). A graph-to-sequence model for amr-to-text generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 1616–1626.

[153] Song, L., Zhang, Y., Wang, Z., and Gildea, D. (2018b). N-ary relation extraction using graph state lstm. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

[154] Speer, R., Chin, J., and Havasi, C. (2017). Conceptnet 5.5: An open multilingual graph of general knowledge. *AAAI 31*, pages 4444–4451.

[155] Stuart, T. and Satija, R. (2019). Integrative single-cell analysis. *Nature Reviews Genetics*, 20(5).

[156] Su, D., Xu, Y., Yu, T., Siddique, F. B., Barezi, E. J., and Fung, P. (2020). Caire-covid: A question answering and query-focused multi-document summarization system for covid-19 scholarly information management. In *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*.

[157] Suchanek, F. M., Kasneci, G., and Weikum, G. (2008). YAGO: A large ontology from wikipedia and wordnet. *J. Web Semant.*, 6(3):203–217.

[158] Sun, Z., Chen, M., Hu, W., Wang, C., Dai, J., and Zhang, W. (2020a). Knowledge association with hyperbolic knowledge graph embeddings. *CoRR*, abs/2010.02162.

[159] Sun, Z., Deng, Z., Nie, J., and Tang, J. (2019). Rotate: Knowledge graph embedding by relational rotation in complex space. In *Proceedings of the ICLR*.

[160] Sun, Z., Hu, W., and Li, C. (2017). Cross-lingual entity alignment via joint attribute-preserving embedding. In *Proceedings of ISWC, pages 628–644*.

[161] Sun, Z., Hu, W., Zhang, Q., and Qu, Y. (2018). Bootstrapping entity alignment with knowledge graph embedding. In *Proceedings of IJCAI, pages 4396–4402*.

[162] Sun, Z., Vashishth, S., Sanyal, S., Talukdar, P. P., and Yang, Y. (2020b). A re-evaluation of knowledge graph completion methods. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5516–5522.

[163] Sun, Z., Wang, C., Hu, W., Chen, M., Dai, J., Zhang, W., and Qu, Y. (2020c). Knowledge graph alignment network with gated multi-hop neighborhood aggregation. In *Proceedings of the AAAI, pages 222–229*.

[164] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2014). Going deeper with convolutions. In *ImageNet Large-Scale Visual Recognition Challenge 2014 (ILSVRC 2014)*.

[165] Tang, Y., Huang, J., Wang, G., He, X., and Zhou, B. (2020). Orthogonal relation transforms with graph context modeling for knowledge graph embedding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2713–2722.

[166] Toutanova, K., Chen, D., Pantel, P., Poon, H., Choudhury, P., and Gamon, M. (2015). Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1499–1509.

[167] Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., and Bouchard, G. (2016). Complex embeddings for simple link prediction. In *Proceedings of the ICML, pages 2071–2080.*

[168] Vashishth, S., Sanyal, S., Nitin, V., Agrawal, N., and Talukdar, P. P. (2020). Interacte: Improving convolution-based knowledge graph embeddings by increasing feature interactions. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 3009–3016.

[169] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural In-*

formation Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA, pages 5998–6008.

[170] Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. (2017). Graph attention networks. *ArXiv*, abs/1710.10903.

[171] Vinyals, O., Bengio, S., and Kudlur, M. (2016). Order matters: Sequence to sequence for sets. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

[172] Vrandecic, D. (2012). Wikidata: a new platform for collaborative data collection. In *Proceedings of WWW, pages 1063–1064*.

[173] Vrandečić, D. and Krötzsch, M. (2014). Wikidata: A free collaborative knowledgebase. *Communications of the ACM*, 57:78 – 85.

[174] Wang, H., Zhang, F., Zhang, M., Leskovec, J., Zhao, M., Li, W., and Wang, Z. (2019a). Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In *KDD '19: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, page 968–977.

[175] Wang, J., Huang, P., Zhao, H., Zhang, Z., Zhao, B., and Lee, D. L. (2018a). Billion-scale commodity embedding for e-commerce recommendation in alibaba.

In *KDD '18: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, pages 839–848. the Association for Computing Machinery.

[176] Wang, K., Liu, Y., Xu, X., and Lin, D. (2018b). Knowledge graph embedding with entity neighbors and deep memory network. In *CoRR '18: CoRR, abs/1808.03752.*

[177] Wang, L., Lin, Z. Q., and Wong, A. (2020a). Covid-net: A tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images. *arXiv*.

[178] Wang, L. L., Lo, K., Chandrasekhar, Y., Reas, R., Yang, J., Eide, D., Funk, K., Kinney, R., Liu, Z., Merrill, W., Mooney, P., Murdick, D., Rishi, D., Sheehan, J., Shen, Z., Stilson, B., Wade, A. D., Wang, K., Wilhelm, C., Xie, B., Raymond, D., Weld, D. S., Etzioni, O., and Kohlmeier, S. (2020b). CORD-19: the covid-19 open research dataset. *CoRR*, abs/2004.10706.

[179] Wang, Q., Huang, P., Wang, H., Dai, S., Jiang, W., Liu, J., Lyu, Y., Zhu, Y., and Wu, H. (2019b). Coke: Contextualized knowledge graph embedding. *CoRR*, abs/1911.02168.

[180] Wang, Q., Li, M., Wang, X., Parulian, N., Han, G., Ma, J., Tu, J., Lin, Y.,

Zhang, H., Liu, W., Chauhan, A., Guan, Y., Li, B., Li, R., Song, X., Ji, H., Han, J., Chang, S., Pustejovsky, J., Rah, J., Liem, D., Elsayed, A., Palmer, M., Voss, C. R., Schneider, C., and Onyshkevych, B. A. (2020c). COVID-19 literature knowledge graph construction and drug repurposing report generation. *CoRR*, abs/2007.00576.

[181] Wang, Q., Mao, Z., Wang, B., and Guo, L. (2017). Knowledge graph embedding: A survey of approaches and applications. *IEEE Trans. Knowl. Data Eng.*, 29(12):2724–2743.

[182] Wang, R., Li, B., Hu, S., Du, W., and Zhang, M. (2020d). Knowledge graph embedding via graph attenuated attention networks. *IEEE Access*, 8:5212–5224.

[183] Wang, S., Kang, B., Ma, J., Zeng, X., Xiao, M., Guo, J., Cai, M., Yang, J., Li, Y., Meng, X., and Xu, B. (2020e). A deep learning algorithm using ct images to screen for corona virus disease (covid-19). *medrXiv*.

[184] Wang, X., Liu, W., Chauhan, A., Guan, Y., and Han, J. (2020f). Automatic textual evidence mining in COVID-19 literature. *CoRR*, abs/2004.12563.

[185] Wang, X., Wang, D., Xu, C., He, X., Cao, Y., and Chua, T. (2019c). Explainable reasoning over knowledge graphs for recommendation. In *Proceedings of the AAAI, pages 5329–5336*.

[186] Wang, X., Ye, Y., and Gupta, A. (2018c). Zero-shot recognition via semantic embeddings and knowledge graphs. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[187] Wang, Z., Li, J., Wang, Z., Li, S., Li, M., Zhang, D., Shi, Y., Liu, Y., Zhang, P., and Tang, J. (2013). Xlore: A large-scale english-chinese bilingual knowledge graph. *ISWC-PD '13: Proceedings of the 12th International Semantic Web Conference (Posters Demonstrations Track)*, 1035:121–124.

[188] Wang, Z., Lv, Q., Lan, X., and Zhang, Y. (2018d). Cross-lingual knowledge graph alignment via graph convolutional networks. In *Proceedings of the EMNLP, pages 349–357*.

[189] Wang, Z., Lv, Q., Lan, X., and Zhang, Y. (2018e). Cross-lingual knowledge graph alignment via graph convolutional networks. In *EMNLP '18: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, page 349–357.

[190] Wang, Z., Zhang, J., Feng, J., and Chen, Z. (2014). Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*, pages 1112–1119.

[191] Wei, Y., Wang, X., Nie, L., He, X., and Chua, T. (2020). Graph-refined convolutional network for multimedia recommendation with implicit feedback. In *Proceedings of the MM, pages 3541–3549*.

[192] West, R., Gabrilovich, E., Murphy, K., Sun, S., Gupta, R., and Lin, D. (2014). Knowledge base completion via search-based question answering. In *23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014*, pages 515–526.

[193] Wise, C., Ioannidis, V. N., Calvo, M. R., Song, X., Price, G., Kulkarni, N., Brand, R., Bhatia, P., and Karypis, G. (2020). COVID-19 knowledge graph: Accelerating information retrieval and discovery for scientific literature. *CoRR*, abs/2007.12731.

[194] Wolf, F. A., Angerer, P., and J., T. F. (2018). Scanpy: largescale single-cell gene expression data analysis. *Genome Biology*, 19(15).

[195] Wu, P., Huang, S., Weng, R., Zheng, Z., Zhang, J., Yan, X., and Chen, J. (2019a). Learning representation mapping for relation detection in knowledge base question answering. In *Proceedings of the ACL, pages 6130–6139*.

[196] Wu, Y., Liu, X., Feng, Y., Wang, Z., Yan, R., and Zhao, D. (2019b). Relation-

aware entity alignment for heterogeneous knowledge graphs. In *Proceedings of IJCAI, pages 5278–5284*.

[197] Wu, Y., Liu, X., Feng, Y., Wang, Z., and Zhao, D. (2019c). Jointly learning entity and relation representations for entity alignment. In *Proceedings of EMNLP-IJCNLP, pages 240–249*.

[198] Wu, Y., Liu, X., Feng, Y., Wang, Z., and Zhao, D. (2020). Neighborhood matching network for entity alignment. In *Proceedings of the ACL, pages 6477–6487*.

[199] Wu, Y. and Wang, Z. (2018). Knowledge graph embedding with numeric attributes of entities. In *Proceedings of the 3rd Workshop on Representation Learning for NLP, pages 132–136*.

[200] Xiao, H., Huang, M., and Zhu, X. (2016). Transg : A generative model for knowledge graph embedding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

[201] Xiaoqi, W., Xin, B., Xu, Z., Li, K., Li, F., Zhong, W., Tan, W., and Peng, S. (2020). Network representation learning-based drug mechanism discovery and anti-inflammatory response against covid-19. *ChemRxiv*.

[202] Xie, Z., Zhou, G., Liu, J., and Huang, J. X. (2020a). Reinceptione: Relation-aware inception network with joint local-global structural information for knowledge graph embedding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5929–5939.

[203] Xie, Z., Zhu, R., Zhao, K., Liu, J., Zhou, G., and Huang, J. X. (2020b). A contextual alignment enhanced cross graph attention network for cross lingual entity alignment.

[204] Xie, Z., Zhu, R., Zhao, K., Liu, J., Zhou, G., and Huang, J. X. (2020c). A contextual alignment enhanced cross graph attention network for cross-lingual entity alignment. In *Proceedings of the 28th International Conference on Computational Linguistics*, page 5918–5928.

[205] Xinyi, Z. and Chen, L. (2019). Capsule graph neural network. In *ICLR2019*.

[206] Xiong, C., Power, R., and Callan, J. (2017a). Explicit semantic ranking for academic search via knowledge graph embedding. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*, pages 1271–1279.

[207] Xiong, W., Hoang, T., and Wang, W. Y. (2017b). Deeppath: A reinforcement

learning method for knowledge graph reasoning. In *Proceedings of EMNLP, pages 564–573.*

[208] Xiong, W., Yu, M., Chang, S., Guo, X., and Wang, W. Y. (2019). Improving question answering over incomplete kbs with knowledge-aware reader. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4258–4264.

[209] Xu, C. and Li, R. (2019). Relation embedding with dihedral group in knowledge graph. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 263–272.

[210] Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2019a). How powerful are graph neural networks? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019.*

[211] Xu, K., Song, L., Feng, Y., Song, Y., and Yu, D. (2020). Coordinated reasoning for cross-lingual knowledge graph alignment. In *Proceedings of the AAAI, pages 9354–9361.*

[212] Xu, K., Wang, L., Yu, M., Feng, Y., Song, Y., Wang, Z., and Yu, D. (2019b).

Cross-lingual knowledge graph alignment via graph matching neural network. In *Proceedings of the ACL, pages 3156–3161.*

[213] Xue, Y., Yuan, Y., Xu, Z., and Sabharwal, A. (2018). Expanding holographic embeddings for knowledge completion. In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, pages 4496–4506.

[214] Yan, Z., Peng, R., Wang, Y., and Li, W. (2020). Ctea: Context and topic enhanced entity alignment for knowledge graphs. *Neurocomputing*, 410:419–431.

[215] Yang, B. and Mitchell, T. M. (2017). Leveraging knowledge bases in lstms for improving machine reading. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1436–1446.

[216] Yang, B., Yih, W., He, X., Gao, J., and Deng, L. (2015). Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the ICLR.*

[217] Yang, H., Zou, Y., Shi, P., Lu, W., Lin, J., and Sun, X. (2019a). Aligning cross-

lingual entities with multi-aspect information. In *Proceedings of the EMNLP, pages 4430–4440*.

[218] Yang, S., Tian, J., Zhang, H., Yan, J., He, H., and Jin, Y. (2019b). Transms: Knowledge graph embedding for complex relations by multidirectional semantics. In Kraus, S., editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 1935–1942. ijcai.org.

[219] Yao, L., Mao, C., and Luo, Y. (2019a). Graph convolutional networks for text classification. In *33rd AAAI Conference on Artificial Intelligence (AAAI 2019)*.

[220] Yao, L., Mao, C., and Luo, Y. (2019b). KG-BERT: BERT for knowledge graph completion. *CoRR*, abs/1909.03193.

[221] Ying, R., R. He, K. C., Eksombatchai, P., Hamilton, W. L., and Leskovec, J. (2018). Graph convolutional neural networks for web-scale recommender systems. *KDD '18: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery Data MiningJuly 2018*, page 974–983.

[222] You, J., Liu, B., Ying, R., Pande, V., and Leskovec, J. (2018). Graph convolutional policy network for goal-directed molecular graph generation. In *32nd*

Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

[223] Yu, X., Liu, Y., Huang, X., and An, A. (2012). Mining online reviews for predicting sales performance: A case study in the movie domain. *IEEE Trans. Knowl. Data Eng.*, 24(4):720–734.

[224] Zaheer, M., Kottur, S., Ravanbakhsh, S., Póczos, B., Salakhutdinov, R., and Smola, A. J. (2017). Deep sets. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3391–3401.

[225] Zeng, X., Song, X., Ma, T., Pan, X., Zhou, Y., Hou, Y., Zhang, Z., Karypis, G., and Cheng, F. (2020). Repurpose open data to discover therapeutics for COVID-19 using deep learning. *Journal of Proteome Research*, acs.jproteome.0c00316.

[226] Zhang, S., Tay, Y., Yao, L., and Liu, Q. (2019a). Quaternion knowledge graph embeddings. In *Proceedings of the NeurIPS, pages 2731–2741.*

[227] Zhang, Y., Liu, L., Fu, S., and Zhong, F. (2018a). Entity alignment across knowledge graphs based on representative relations selection. pages 1056–1061.

[228] Zhang, Y., Liu, Q., and Song, L. (2018b). Sentence-state lstm for text rep-

resentation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

[229] Zhang, Y., Qi, P., and Manning, C. D. (2018c). Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

[230] Zhang, Z., Cai, J., Zhang, Y., and Wang, J. (2020). Learning hierarchy-aware knowledge graph embeddings for link prediction. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 3065–3072.

[231] Zhang, Z., Han, X., Liu, Z., Jiang, X., Sun, M., and Liu, Q. (2019b). Enhanced language representation with informative entities. In *Proceedings of the ACL, pages 1441–1451*.

[232] Zhao, S., Qin, B., Liu, T., and Wang, F. (2020). Biomedical knowledge graph refinement with embedding and logic rules. *arXiv*.

[233] Zheng, G. X. Y., Terry, J. M., Belgrader, P., Ryvkin, P., Bent, Z. W., Wilson, R., Ziraldo, S. B., Wheeler, T. D., McDermott, G. P., Zhu, J., Gregory, M. T.,

Shuga, J., Montesclaros, L., Underwood, J. G., Masquelier, D. A., Y.Nishimura, S., Schnall-Levin, M., Wyatt, P. W., Hindson, C. M., Bharadwaj, R., Wong, A., Ness, K. D., Beppu, L. W., Deeg, H. J., McFarland, C., Loeb, K. R., Valente, W. J., Ericson, N. G., Stevens, E. A., Radich, J. P., Mikkelsen, T. S., Hindson, B. J., and Bielas, J. H. (2017). Massively parallel digital transcriptional profiling of single cells. *Nature Communications*, 8:14049.

[234] Zheng, N., Du, S., Wang, J., Zhang, H., Cui, W., Kang, Z., Yang, T., Lou, B., Chi, Y., Long, H., Ma, M., Yuan, Q., Zhang, S., Zhang, D., Ye, F., and Xin, J. (2020). Predicting covid-19 in china using hybrid ai model. In *IEEE Transactions on Cybernetics*, volume 50.

[235] Zhu, H., Xie, R., Liu, Z., and Sun, M. (2017). Iterative entity alignment via joint knowledge embeddings. In *Proceedings of the IJCAI, pages 4258–4264*.

[236] Zhu, Q., Zhou, X., Wu, J., Tan, J., and Guo, L. (2019). Neighborhood-aware attentional representation for multilingual knowledge graphs. In *Proceedings of IJCAI, pages 1943–1949*.

[237] Zhu, R. (2020). Bridging east and west: An integration of tcm and western medicine in medical text mining. In *SIGIR '20: Proceedings of the 43rd Inter-*

national ACM SIGIR Conference on Research and Development in Information Retrieval, page 2488.

[238] Zhu, R., Tu, X., and Huang, X. (2020). Using deep learning based natural language processing techniques for clinical decision-making with ehrs. *Deep Learning Techniques for Biomedical and Health Informatics*, pages 257–295.

[239] Zhu, R., Tu, X., and Huang, X. (2021). Utilizing bert for biomedical and clinical text mining. *Data Analytics in Biomedical Engineering and Healthcare*, pages 73–103.

[240] Zitnik, M., Agrawal, M., and Leskovec, J. (2018). Modeling polypharmacy side effects with graph convolutional networks. *ISMB Bioinformatics*, 34:13:457–466.

A Appendices

Our experiments from Chapter 3 and 4 are conducted on the deep learning framework Tensorflow [8] for our model implementation. The version is Tensorflow 1.14. All the experiments in this thesis are conducted on RTX 2080Ti and NVIDIA GTX 1080Ti GPUs respectively. Our experiments for Chapter 5 are conducted on a server with a single NVIDIA GTX 1080Ti GPU with PyTorch. All the datasets and codes in this thesis are available at https://github.com/sherrybbalonsozhu.

A.1 TASLP

A.1.1 Data Explanations

The datasets used in this study are listed below:

[8] https://tensorflow.org

Table A.1: Description of the benchmark datasets.

	#Entity	#Relation	#Train	#Valid	#Test
WN18RR	40,943	11	86,835	3034	3134
FB15k-237	14,541	237	272,115	17,535	20,466
NELL-995	75,492	200	149,678	543	3992
Kinship	104	25	8544	1068	1074

A.1.2 Code Explanations

Details of the codes are included in the documentation in files on Github link.

A.2 CAECGAT

A.2.1 Data Explanations

DBpedia is a large-scale multi-lingual KB including inter-language links (ILLs) from entities of English version to those in other languages. In our experiments, we extracted 15 thousand ILLs with popular entities from English to Chinese, Japanese and French respectively, and considered them as our reference alignment (i.e., gold standards). Our strategy to extract datasets is that we randomly selected an ILL pair s.t. the involved entities have at least 4 relationship triples and then extracted relationship and attribute infobox triples for selected entities. The statistics of the

Algorithm 3 CAECGAT Model

Input: Given KGs G_1 and G_2, as well as the initial entity embeddings $\mathbf{E}_1$, $\mathbf{E}_2$ and pre-aligned seed alignments A.

Output: The final entity embeddings $\bar{\mathbf{E}}_1$, $\bar{\mathbf{E}}_2$ and parameters ϕ.

Let $\mathbf{E}_1^0 = \mathbf{E}_1$ and $\mathbf{E}_2^0 = \mathbf{E}_2$.
repeat
    Select a batch seed alignments as objective seed alignments A_{obj} and the rest are used as context seed alignments A_{ctx}.
    for $l = 1$ to L **do**
        $\mathbf{E}_1^{l+1} = crossAtt(crossAggr(\mathbf{E}_1^l, \mathbf{E}_2^l, A_{ctx}), G_1)$;
        $\mathbf{E}_2^{l+1} = crossAtt(crossAggr(\mathbf{E}_2^l, \mathbf{E}_1^l, A_{ctx}), G_2)$;
    end for
    Set $\bar{\mathbf{E}}_1 = \mathbf{E}_1^L$ and $\bar{\mathbf{E}}_2 = \mathbf{E}_2^L$
    Compute loss function $\mathcal{L}(\phi; A_{obj})$.
    Optimize parameters ϕ in CAECGAT model.
until reach the maximum number of training epochs
Set $A_{ctx} = A$.
Use the learned CAECGAT model to predict the final entity embeddings $\bar{\mathbf{E}}_1$ and $\bar{\mathbf{E}}_2$. =0

three datasets are listed in Table 1 in the chapter, which indicate that the number of involved entities in each language is much larger than 15 thousand, and attribute triples contribute to a significant portion of the datasets.

A.2.2 Code Explanations

The training process of our CAECGAT model is illustrated as follows: Other details of the codes are included in the documentation in files on Github link.

Table A.2: Statistics of the datasets.

Dataset		Entities	Relations	Rel.Triples	Training/Dev/Test
DBP15K_{ZH-EN}	Chinese	66,469	2,830	153,929	4,050/450/10,500
	English	98,125	2,317	237,674	
DBP15K_{JA-EN}	Japanese	65,744	2,043	164,373	4,050/450/10,500
	English	95,680	2,096	233,319	
DBP15K_{FR-EN}	French	66,858	1,379	192,191	4,050/450/10,500
	English	105,889	2,209	278,590	
WK31-60K_{FR-EN}	French	45,255	277	258,337	14,095/1,566/36,544
	English	64,539	458	569,393	
WK31-60K_{FR-DE}	German	43,503	172	244,647	14,451/1605/37,467
	English	64,539	458	569,393	

A.3 TOIS

A.3.1 Data Explanations

The datasets used in this study are listed below:

A.3.2 Code Explanations

The training process of our DuGa-DIT model is illustrated as follows: Other details of the codes are included in the documentation in files on Github link.

Algorithm 4 DuGa-DIT Model

Input: Given two KGs G_1 and G_2, the initial entity embeddings $\mathbf{E}_1$, $\mathbf{E}_2$ and pre-aligned seed alignments A.

Output: The final entity embeddings $\mathbf{E}_1^L$ and $\mathbf{E}_2^L$.

Initialize A^{new} as an empty set
repeat
  Let $\mathbf{E}_1^0 = \mathbf{E}_1$ and $\mathbf{E}_2^0 = \mathbf{E}_2$.
  repeat
    Randomly choose some seed alignments from A as A_{obj} and the rest as A_{ctx}.
    Randomly choose some seed alignments from A^{new} as A_{obj}^{new} and the rest as A_{ctx}^{new}.
    for $l = 1$ to L **do**
      Perform intra-KG attention and cross-KG attention and obtain H_1^l, H_2^l, C_1^l and C_2^l
      Update entity embeddings:
      $\mathbf{E}_1^l = \mathbf{H}_1^l \cdot g_1 + \mathbf{C}_1^l \cdot (1 - g_1)$;
      $\mathbf{E}_2^l = \mathbf{H}_2^l \cdot g_2 + \mathbf{C}_2^l \cdot (1 - g_2)$;
    end for
    Compute loss function $\mathcal{L}(\phi; A_{obj} \bigcup A_{obj}^{new})$.
    Optimize parameters ϕ in DuGa-DIT model.
  until reach the maximum number of training epochs
  Predict new seed alignments and update A^{new}
until reach the maximum number of iterations
Set $A_{ctx} = A \bigcup A^{new}$.
Use the learned DuGa-DIT model to predict the final entity embeddings $\mathbf{E}_1^L$ and $\mathbf{E}_2^L$.
=0

Table A.3: Statistic of the datasets.

Dataset	Nodes(train/val/test)	Node features	Edges(train/val/test)	Classes
SARS-CoV-2 infected organoids	54353/11646/11648	24714	1041226/230429/228630	7
COVID-19 patients	63486/13604/13605	25626	2746280/703217/707529	3

A.4 GACapNet

A.4.1 Data Explanations

The datasets used in this study are listed below:

A.4.2 Code Explanations

Details of the codes are included in the documentation in files on Github link.