Deep Learning on Graph Structured Data: Algorithms and Applications

Submitted by

Muthuthanthrige Thilini Thushari COORAY

Thesis Advisor

Prof. Ngai-Man CHEUNG and Prof. Wei LU

Information Systems Technology and Design

A thesis submitted to the Singapore University of Technology and Design in fulfillment of the requirement for the degree of Doctor of Philosophy

2021

PhD Thesis Examination Committee

TEC Chair:	Prof. Zhou Jianying
Advisor:	Prof. Ngai-Man Cheung
Advisor:	Prof. Wei Lu
Internal TEC member 1:	Prof. Lim Kwan Hui
Internal TEC member 2:	Prof. Liu Jun

Abstract

Information Systems Technology and Design

Doctor of Philosophy

Deep Learning on Graph Structured Data: Algorithms and Applications

by Muthuthanthrige Thilini Thushari COORAY

Graph deep learning aims at expanding deep learning techniques to non-euclidean data structures, improving the generalizability of neural models for data with arbitrary structures. Graphs are highly generic data structures that can be used to represent both the data which are naturally formed in a network structure such as social networks or chemical compounds, as well as any scenario which could be modelled as a network such as a scene graph from an image or a knowledge graph from a natural language text. Therefore, it is important to improve algorithm designs not only to focus on catering natural graph structured data, but also to be flexible enough to incorporate application specific requirements.

When designing algorithms for natural graph structured data, unsupervised graph representation learning plays a crucial role, specially when data annotation is expensive. Inspired by the real world graph generation processes where the graphs are formed based on one or more global factors which are common to all elements of the graph (e.g., topic of a discussion thread, solubility level of a molecule), we propose to extract graph-wise common latent factors filtering node-level specific factors as graph embeddings. We empirically demonstrate that, while extracting common latent factors is beneficial for graph level tasks to alleviate distractions caused by local variations of individual nodes or local neighbourhoods, it also benefits node level tasks by enabling long-range node dependencies specially for disassortative graphs.

Then, we move to an application of deep graph learning; Situation Recognition, a Vision and Language structured prediction task. Graph-encoding neural models are designed to address structure related requirements like neighbourhood information propagation. When those are applied directly to advanced reasoning tasks such as Situation Recognition without properly adapting to the domain of the application, they are unable to perform well due to their inability in complex multi-modal reasoning. We address this by proposing two novel approaches; a transfer learning based iterative mechanism and an inter-dependent latent query based reasoning mechanism. Unlike graph neural networks, our query based information propagation methods do not over emphasize on inter-node similarity in neighbourhoods, which causes bias towards frequent neighbour co-occurrence patterns ignoring rarely occurred but plausible scenarios.

Publications

- **Thilini Cooray** and Ngai-Man Cheung. 2021. Graph-wise Common Latent Factor Extraction for Unsupervised Graph Representation Learning. Accepted to be in *Proceedings of The Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI-22).*
- **Thilini Cooray** and Ngai-Man Cheung. 2021. Efficient graph learning via extraction of graph-wise common latent factors. *Under submission.*
- **Thilini Cooray**, Ngai-Man Cheung and Wei Lu. 2020. Attention-Based Context Aware Reasoning for Situation Recognition. In *Proceedings of The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).*
- **Thilini Cooray**, Ngai-Man Cheung and Wei Lu. 2019. Sometime you just need to ask: Situation Recognition via VQA. Presented at *Workshop on Language and Vision at The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).*

Acknowledgements

First and foremost, I would like to express gratitude for my thesis advisors Prof. Ngai-Man Cheung and Prof. Wei Lu. I am very grateful for my advisors for accepting me to SUTD and providing me with all the guidance and resources to conduct my research. I am specially thankful for Prof. Ngai-Man Cheung for sharing his expert knowledge in Computer Vision and Geometric Machine Learning domains and providing me very valuable feedback for my work in those areas. I would also like to thank my thesis committee consisting of Prof. Zhou Jianying, Prof. Lim Kwan Hui and Prof. Liu Jun for their valuable feedback.

I would like to give a special thanks to Chen Sin Chee, Associate Program Director - Graduate Studies for ISTD. Thank you for always looking out for us and trying to help us in difficult situations. Personally, I am very grateful for your support and encouragement. I also take this opportunity to thank Office of Graduate Studies for their support through out my studies.

I am very grateful for SUTD, SUTD hostel and Singapore for hosting me and providing me with all the facilities to carry out my studies for the last five years.

In addition, I would like to sincerely thank my batch mates Penny Chong and Marie Siew, for always being there for me. Thank you for never abandoning me and always being supportive. I would also like to thank my friends Himan Gamage and Supuni Weliwita for always welcoming me to their home and making me feel like family. My gratitude also goes to Girisha De Silva for his enormous help when I first came to Singapore. I also like to thank Allan Jie and Tuan Hoang for their support.

I also add my utmost gratitude to my homeland, Sri Lanka. You provided me almost eighteen years of free education from primary to tertiary, that made me who I am today. You not only nurtured me with education, but also built my self confidence and showed me by example the value of humanity. All these things helped me to survive in the outside world.

I kept the most important people to whom am in debt, to last. I am dedicating this thesis to my parents and my husband. I am so lucky to have strong and loving parents who supported me throughout my life and encouraged me in every step I took. I am also very fortunate to have my husband, the best partner one can ask for. Although we live oceans apart, thank you for providing me constant company and being my rock. If I have achieved any success, it is all because the three of you believed in me.

Contents

List of Figures

List of Tables

To my parents & my husband, for always believing in me.

Chapter 1

Introduction

Machine learning has caused massive advancements in the field of Artificial Intelligence with its ability to conduct tasks with less human intervention Artificial Neural Networks have taken the centre stage in it. Deep learning has taken a further step on machine learning with more sophisticated models following human brain, where it thrives in a plethora of data modalities including images (LeCun, Y. Bengio, and Hinton, 2015), videos (W. Zhang et al., 2016), text (Schuster and Paliwal, 1997; Vaswani et al., 2017), speech (Graves et al., 2006) and networks (Thomas N Kipf and Welling, 2016a) from domains such as Computer Vision, Natural Language Processing and Graph Machine Learning. These advanced systems have paved the way for variety of real life applications such as robotics (Malekzadeh et al., 2017), self-driving cars (A. Gupta et al., 2021), language translation (Klein et al., 2017), drug discovery (H. Chen et al., 2018) and recommendation systems (Shuai Zhang et al., 2019), which are widely used for improving the quality of life for humans (Robertson et al., 2019), conserve the environment (Ridge et al., 2020) as well as to foster advances in other scientific domains such as space exploration (Bird et al., 2021).

Graph deep learning (Michael M. Bronstein et al., 2021b) aims at expanding the deep learning techniques to non-euclidean data structures improving the generalizability of neural models for data with arbitrary structure. A graph is a representation of multiple inter-related objects where the objects or the entities are represented as nodes and their relationships are indicated by connections among nodes. Graphs are a generic data structure that can be used to represent variety of information from different domains including, but not limited to social networks (Newman and Girvan, 2004), sensor networks (Suhonen et al., 2012), knowledge graphs (Yeh and Ratnaparkhi, 2014), protein-protein interactions (Krogan et al., 2006), citation networks (Z. Yang, Cohen, and Salakhutdinov, 2016) and customer purchasing patterns (Bhatia et al., 2016). Extracting knowledge from these are vital for applications such as community detection (Yanardag and Vishwanathan, 2015), material design (Sanchez-Lengeling and Aspuru-Guzik, 2018), protein property detection (Duvenaud et al., 2015) for disease identification and drug design.

Unlike for other data modalities such as images or natural language text, patterns in graphs are not always easily human recognizable. For an example, in a chemical compound graph, certain bonds types among molecules could make the compound more water-soluble increasing its usability in drug design, while other bond types among same molecules could make them toxic for humans. To identify such complex patterns, even humans require expert knowledge. Schneider and Fechner (2005) state that, chemists are required to search through an estimated space of $10^{60} \sim 10^{100}$ synthetically

available molecules to filter potential leads for a useful drug. This is a very exhaustive process (Gómez-Bombarelli et al., 2018) and the filtered properties could be highly biased towards the expertise of the involved personals (Merkwirth and Lengauer, 2005). Prediction of more general molecule properties is very important to find better drug candidates and using computational methods to achieve this can vastly enhance the efficiency of the process, which on average costs around $2.8 billion per a discovery (K. Yang et al., 2019; Wieder et al., 2020).

Problems like these emphasize the requirement of properly designed machine learning algorithms for graph structured data. Graph Neural Networks (GNNs) have recently become the prominent approach for representing graph structured data (Yujia Li, Tarlow, Brockschmidt, and R. S. Zemel, 2016; Gilmer et al., 2017; Thomas N. Kipf and Welling, 2017; Velickovic, Cucurull, et al., 2018; K. Xu et al., 2019). GNNs are capable of representing graphs in a permutation invariant manner, enabling information propagation among neighbours and mapping graphs to low dimensional spaces. However, to train them properly to learn relevant knowledge, annotated datasets are required. Due to huge varieties of domains which require expertise knowledge in graph structured data, data annotation is highly expensive (K. Yang et al., 2019; Wieder et al., 2020; F.-Y. Sun et al., 2020). Hence, unsupervised graph representation learning have drawn the attention of graph deep learning community.

Main intention of unsupervised graph representation learning is to learn and extract useful knowledge from graphs which can later be utilized in a task-specific manner (Gärtner, Flach, and Wrobel, 2003; Grover and Leskovec, 2016; Adhikari et al., 2018; F.-Y. Sun et al., 2020; Hassani and Khasahmadi, 2020). Main challenge of unsupervised graph representation learning is determining the relevant information for each level of the graph, and differentiating the information required for one level of the graph from another level without annotations. For an example, the most widely categorized levels of a graph are node-level and graph-level. While node-level tasks such as node classification require information to differentiate each node from all others, graph-level representation of the same graph requires to capture properties common to the entire graph to represent it as a whole. Incorporating node specific properties to a graph-level representation could add noise and cause performance degradation. Similarly adding graph-level information for node-level representations could potentially hinder node-wise specific properties which are crucial for node-level tasks. Therefore, is it possible to differentiate node-level information from graph-level in an unsupervised manner?

To answer this question, in this thesis, we propose a generative based approach in Chapter 3. Instead of treating graph-level information as the combination of all node-level information, we take a different perspective of the graph to enable a filtering process. Our model is formulated to extract features common for the entire graph, filtering out each node-specific unique features. Hence, the graph-wise common features can be considered to capture high-level features relevant to the entire graph. Although we do not have access to downstream task annotations during training, we show that extracted graph-wise common factors indeed capable of capturing information valuable for graph-level representations. Additionally, this filtering process provides the flexibility to combine common and non-common features optimally, in a task-specific manner without node-level features becoming noise to graph-level embeddings and

vice versa. In this way, our model can incorporate long range node dependency information to node-level representations while also preserving node specific features.

Apart from information naturally structured as graphs or networks, a recent interest has been emerging to model problems of other data modalities such as images and text as graphs and apply graph deep learning algorithms to solve them. We look at this direction as *Applications of graph deep learning*. In Computer Vision, images can be modelled as graphs in multiple different ways. First is, in pixel level, where each pixel is considered as a node and edges are formed if two pixels are next to each other based on its spatial location (Monti et al., 2017). Next is, a scene in an image can be modelled as a graph where objects are nodes and object relations such as book-on-table can be used to form triplets (subject-predicate-object) and these are used to form a scene graph (D. Xu, Y. Zhu, Christopher B. Choy, et al., 2017b). A human's poses can also be modelled as graphs (J. Wang et al., 2020) where specific key joints of human body with their position coordinates are treated as nodes and how the joints are connected forms the edges. Similarly in natural language, text can also be modelled as graphs. A sentence can be modelled as a dependency tree (Kulick, Bies, and Mott, 2012) where words are nodes and grammatical relations among words form edges. A sentence can also be modelled as a directed acyclic graph in the semantic-level using Abstract Meaning Representation (Sheng Zhang et al., 2019), where concepts of the sentence are nodes and edges are formed for semantic relations among concepts. Next, a document can be modelled as a graph by extracting the most important sentences from the document and forming them as nodes and their conceptual relations as edges (Yasunaga et al., 2017). A discussion thread can also be modelled as a graph where the responses from users of the thread are nodes and edges are formed if one response is a reply to another (Hamilton, Ying, and Leskovec, 2017). Inspired by the applied graph learning techniques of Computer Vision and Natural Language Processing, Vision and Language multi-modal domain has also made efforts in modelling their problems as graphs. For tasks like visual question answering (Teney, L. Liu, and Hengel, 2017) and visual commonsense reasoning (W. Yu et al., 2019), the input questions, images and available concepts are modelled as graphs and graph alignment is used to match the text with objects in the scenes to find answers.

The main reason for modelling these problems as graphs while having modal specific architectures like Convolutional Neural Nets (CNN) (Lecun and Y. Bengio, 1995; LeCun, Y. Bengio, and Hinton, 2015) and Long Short Term Memory Nets (LSTM) (Hochreiter and Schmidhuber, 1997) to encode images and text is, the flexibility graphs provide in message passing among arbitrary structured data. While CNNs handle images as fixed size grid graphs and LSTMs handle text as line graphs, GNNs can handle both of these modalities with arbitrary sizes and structures. Therefore, most problems apply graph learning on top of modality specific encodings to enable flexible information propagation which have proven to improve task performance. However, for multi-modal tasks, semantic reasoning fueled by multi-modal fusion is the most important component in solving those tasks (J.-H. Kim, Jun, and B.-T. Zhang, 2018; Z. Yu et al., 2018). But, these fusion models do not have the capability to share information among their related elements like in a graph structure. They reason each element which was input to the model independently. On the other hand, GNNs can enable inter-element

information sharing, but existing GNN algorithms do not have multi-modal fusion capabilities to fulfil visual reasoning requirement of multi-modal tasks. Therefore, we find two directions of deep learning approaches, one is capable of advanced multi-modal reasoning and the other is capable of information propagation among inter-related data. One approach cannot fulfil the requirement of other. We identify the requirement of a model which can join both these to enable information sharing among multi-modal reasoning tasks.

To highlight our point of view, let's take human brain function as an example. Consider how a human brain performs a visual question answering task where an image and a set of related questions are given. When answering each question, it would not only focuses on that particular question, but also considers the agreeability among answers for all the questions in order to make sure that the answers are not contradicting with each other. Current multi-modal reasoning tasks cannot achieve this level of reasoning power as they cannot share knowledge among related elements. Is it possible to enable this capability for deep learning based structured semantic reasoning tasks?

To address this question, we analyze from two aspects. First is, on individual node (single task) performance improvement using semantic reasoning and the second is enhancing individual reasoning and maintaining agreeability among all elements of the structure via inter-dependent knowledge sharing. In Chapter 4, we utilize a transfer learning based approach to achieve the former. We identify that although there are variety of multi-modal tasks available with different datasets, the core knowledge they try to obtain is similar. We demonstrate that by showing how individual elements of a structure from a structured visual reasoning prediction task can improve its individual reasoning abilities by transferring knowledge from another different visual reasoning task. Next, to enable information propagation among nodes, we incorporate the transfer learning mechanism to a graph structure where each node value is iteratively updated directly conditioned on its neighbours. Instead of completely relying on predicted neighbour information for final node accuracy, Chapter 5 proposes a learnable component which has the capability to determine the amount of knowledge from neighbours to be incorporated in the semantic reasoning process without suppressing the findings of the individual node reasoning. Proposed model is specially beneficial for learning rarely occurred structures in the dataset compared to normal GNNs. This is because, normal GNNs have a tendency to be highly biased towards learning frequent object co-occurrence patterns in training dataset, ignoring rarely occurring data.

1.1 Thesis Outline

This thesis is organized as follows:

- Chapter 2 presents the background of deep learning on graph structured data and how other modalities have applied these techniques by converting their problems as graph structures. We discuss current approaches and the challenges need to be addressed on graph processing deep learning algorithms and also the pros and cons of applying these techniques on other modalities to highlight the motivation for the problems addressed in this thesis.

- In Chapter 3, we investigate the relationship between factors common to all the elements of a graph and an optimal graph summary representation. We propose a novel principle and an algorithm to extract graph-wise common latent factors in an unsupervised manner and we analyze the impact of those extracted factors on downstream tasks. Experimental results demonstrate not only the extracted common factors' ability to capture knowledge important for the graph-level tasks, but also its ability to capture long-range node dependencies whose incorporation improves node-level performance.
- We move to applications of graph deep learning in Chapter 4. We selected Situation Recognition (SR) as our application, a structured prediction task from Vision and Language domain. We analyze the impact of semantic reasoning improvements obtained by both advanced fusion mechanisms as well as modelling inter-element dependencies like a graph, for the task performance. In this chapter, we propose a transfer learning based method. We adapt Visual Question Answering (VQA) as our source task, whose advanced multi-modal fusion capabilities that we want to transfer to our task. While fusion-based reasoning abilities transferred from VQA improves the performance of SR, we further extend this model to incorporate inter-semantic role knowledge sharing by introducing context aware questions. Our performance further increases showing that, modelling semantic role structure as a graph and applying inter-node information propagation mechanisms is beneficial for SR.
- Chapter 5 further studies the importance of extending advanced visual reasoning capabilities to graph learning algorithms for adopting graph deep learning to Vision and Language application domain. We propose three novel mechanisms for context aware reasoning on SR, which enable inter-dependent question answering inspired by GNN's information propagation mechanism. While achieving state-of-the-art results for SR, our model also demonstrate interpretability which is crucial for error analysis and future improvements.
- In Chapter 6, we present the conclusions of this thesis and discuss possible directions for future work.

1.2 Thesis Contribution

We summarize overall contributions of this thesis as follows:

- Inspired by the formation of real world graphs, we propose the novel principle for unsupervised graph representation learning based on the notion of Graph-wise Common latent Factor EXtraction (GCFX). To the best of our knowledge, deepGCFX presented in Chapter 3 is the first graph embedding learning method based on this principle.
- We empirically demonstrate the strong advantage of GCFX principle in graph representation learning in four directions. First, filtering graph-wise common

latent factors from node-specific uncommon local latent factors facilitates reducing noise caused by irrelevant local node-level information in a graph embedding. Second, separating these two types of latent factors preserving their unique characteristics provides the flexibility to combine them in a task-specific manner, which proved to improve downstream task performance. Third, we observed the ability of extracted graph-wise common factors in capturing long distance node similarities, which is specially beneficial for disassortative graph representation learning. Finally, GCFX enables novel graph generation with desired characteristics by utilizing extracted common and non-common latent factors.

- Next, we introduce an application of graph deep learning by enabling advanced multi-modal fusion capabilities for the task of Situation Recognition. We present a transfer learning method in Chapter 4 (Cooray, Cheung, and Lu, 2019) to obtain advanced semantic reasoning abilities from VQA. This is the first attempt of transfer learning from a VQA system to another visual reasoning task. This is also the first model which utilizes transferred visual reasoning to address Situation Recognition. We propose an iterative extension to this model and a context aware question generation mechanism to adapt this method for graph structured data.

- We further explore this idea of incorporating query based visual reasoning and graph learning based information propagation abilities to SR in Chapter 5 (Cooray, Cheung, and Lu, 2020). To the best of our knowledge, ours is the first attempt in incorporating inter-dependent query handling capabilities to query-based visual reasoning models. Compared to neighbour information propagation mechanisms of existing GNN algorithms, our proposed inter-dependent query based information propagation method does not over emphasize on inter-node agreement in neighbourhood. Hence, it reduces bias towards learning frequent object co-occurrence patterns ignoring rarely occurred but plausible situations.

Chapter 2

Background

This chapter provides the background of the two directions of deep learning on graph structured data; Deep graph learning algorithms and their applications. We also discuss about existing approaches on these directions and point out the gaps in them which motivated us for the contributions of this thesis.

2.1 Graph data structure

Graphs are generic data structures which could be used to represent any set of entities which have relations to connect them with each other. Instead of treating those inter-related entities as independent data points, graph data structure provides a way to describe them with respect to their connections with other data points (Leskovec, 2021). There are so many examples of graphs around us. In a computer network, each computer is an entity and the physical wiring or the optical connections enable relationships among them which ultimately form a graph structure. If we consider a transportation network such as Mass Rapid Transit (MRT) system, each station is an entity which is connected to other entities via transit routes. Social network is another example of graphs where each user is an entity and users connect with each other forming a graph structure via variety of different relationships such as school mates, colleagues, neighbours etc.

We consider that there are two main categories of data which can be described using graph data structure. First is natural graphs. Domains of these can naturally be described as graphs. Examples are communication networks (Quarterman, 1990), social networks (Nasution, 2016), transportation networks (Bell and Iida, 1997) and brain (Fornito, Zalesky, and Bullmore, 2016). The underlying phenomenon of all these is inter-connected entities. Second category is the data which has some relational information and we can model those relational information as graphs. A scene graph (Yikang Li et al., 2017) is one such example; if we consider an image which depicts a scene such as children playing, this scene can be modelled as a graph based on spatial and semantic relations those objects demonstrate in the scene such as "child1 passes ball to child2", "woman standing next to child1 and child1 wears a cap". Other examples are knowledge graphs (Sarrafzadeh, Vechtomova, and Jokic, 2014), 3-dimensional data (Y. Wang et al., 2019) and semantic graphs of text (Lyu and Titov, 2018; D. Cai and Lam, 2020).

Common characteristics of any of these graphs are that they could have any number of entities and any number of connections, the topology those graphs are formed

can be arbitrary as well, entities of graphs usually do not have any ordering and therefore do not have an exact reference or starting point. Due to these complexities, graphs cannot be processed using deep learning approaches proposed for other data modalities (Michael M Bronstein et al., 2021a; Ziwei Zhang, Cui, and W. Zhu, 2020). Deep learning has achieved leaps and bounds in Computer Vision and Natural Language Processing whose underlying data modalities such as images and text are more simple than graphs. Text can be represented as sequences and each word is present in an order which could be easily referenced. Images are fixed sized grids, where each pixel is connected with a fixed set of other pixels in a fixed locality. Existing deep learning models for image (LeCun, Y. Bengio, and Hinton, 2015; Simonyan and Zisserman, 2015) and text (Hochreiter and Schmidhuber, 1997; Graves et al., 2006) have proposed mechanisms to capture these characteristics and they have achieved very high performance due to this in their respective fields. Therefore, it is important to develop deep learning algorithms which can explicitly capture the characteristics specific to graphs to improve performance of graph related tasks via more accurate models.

2.2 Deep learning algorithms for graphs

2.2.1 Challenges

When applying deep learning algorithms for graph structured data, there are many aspects of graphs that cause variety of challenges (Ziwei Zhang, Cui, and W. Zhu, 2020). While some aspects are common to all graphs, others are specific to certain scenarios. We list some of the main aspects below:

- **Arbitrary structure** - As briefly mentioned earlier, graphs can have any number of entities and links forming arbitrary structures. These arbitrary structured nature of graphs make it a more general data structure. Other data structures can be considered as special types of graphs. For an example, an image can be considered as a grid graph which has fixed number of nodes and grid topology. Text can be viewed as a line graph. In general, nodes or entities in a graph do not have any node order or an exact position in the graph structure. Each node can only be described in relative to its neighbours. Therefore, a deep learning algorithm modelling a graph must be able to handle this arbitrary nature of structure without biasing towards any fixed structure link CNN or RNN.

- **Diversity** - There are different types and different levels of graphs which need to be considered when designing deep learning algorithms to accurately represent them. Nodes in a graph may or may not have features, therefore modelling the structure is the most essential objective of a graph learning algorithm. When modelling the structure, the connections among nodes, commonly known as edges could be directed or undirected (Quiterio and Lorena, 2018), weighted or unweighted (Ganie and Chat, 2019), could have multiple edge types and edges could have features as well. Also when considering entire graphs, they can be homogeneous or heterogeneous (Y. Sun and J. Han, 2012), assortative or disassortative (Noldus and Mieghem, 2015). There are different levels a graph can be

represented and each level has its own tasks. node-level (H. Xu et al., 2013; Donnat et al., 2018) and graph-level (Narayanan et al., 2017; M. Zhang et al., 2018) are the two main levels. Apart from them, edge-level (G. Xu et al., 2019; Lim et al., 2019) and sub-graph-level (Moon et al., 2016; Filippidou and Kotidis, 2015; Acosta-Mendoza et al., 2020) can also be seen in the literature.

- **Scalability** - Scalability is a common challenge any deep learning algorithm faces in the big data era. However, graph data structures introduce additional problems such as how to accurately approximate neighbourhoods of each node when the number of nodes and links in a graph increases arbitrarily, how to properly summarize huge graphs without losing important knowledge and how to partition large graphs to process them within available hardware capacities (Bodra et al., 2018; Ziwei Zhang, Cui, and W. Zhu, 2020).

- **Domain Adaptability** - Deep learning algorithms on graphs should not only focus on modelling natural graphs. There are so many domains as mentioned earlier whose data contain variety of relations which could be modelled as graphs and whose task performance can be improved by applying graph deep learning. However, most domains have their own requirements. Multi-modal domains such as Vision and Language and video require advanced fusion methods in their relationship modelling to extract knowledge from all the modalities (Anderson et al., 2018; J.-H. Kim, Jun, and B.-T. Zhang, 2018). In chemical informatics domain, the objective functions based on chemical constraints are not differentiable (Ziwei Zhang, Cui, and W. Zhu, 2020), therefore special training strategies are required to train models for those domains.

2.2.2 Graph encoders

In the scope of this thesis, we focus only on node-level and graph-level encoding. We start by discussing about node encoding also known as node embedding or node representation learning as it is the main building block for entire graph encoding.

Consider a graph G with a set of V nodes with $v \in V,\ v \in \mathbb{R}^m$ and A is the adjacency matrix. m is the size of input features for each node. An encoding function ENC aims at mapping $v \in V$ to a latent low dimensional embedding space $\mathbf{z}_v = \text{ENC}(v),\ \mathbf{z}_v \in \mathbb{R}^d$ such that, for a given pair of nodes $u, v \in V$,

$$Similarity(u, v) \approx \text{SIM}(\mathbf{z}_u, \mathbf{z}_v) \tag{2.1}$$

SIM is the selected function to calculate similarity between $\mathbf{z}_u$ and $\mathbf{z}_v$ in the latent space such as Cosine Similarity. This means that the encoding function ENC should be able to map the nodes closer together in the embedding space if they are similar in the input space.

Prior to deep learning era, several shallow encoding methods had been proposed such as node2vec (Grover and Leskovec, 2016) and DeepWalk (Perozzi, Al-Rfou, and Skiena, 2014). These methods mainly focus on learning similar embeddings to nodes which lie closer together in the graph structure (E.g., if u and v are frequently visited together in a random walk across the graph, they should have similar embeddings). There are several notable drawbacks of these methods. First is, these methods do not

have any parameter sharing mechanism among nodes. Therefore, unique embeddings of all V nodes are the parameters of the model. This makes the model less scalable for large graphs with billions of nodes. Second is, the structure of the graph is only used for node embedding learning without giving much importance to node features. Most real world graphs have node features such as user profile details in social networks or object labels and coordinates of a scene graph. These are crucial to be included in an embedding to maintain its relation with the input space without getting overridden by structural information. Third is, these encoding methods can only be used for transductive tasks where embeddings for new nodes cannot be learnt after training time. This is because model parameters are unique to each node embedding. Hence it is less generalizable.

To address these weaknesses, deep graph encoders were proposed. Graph Neural Networks is the most prominent class of architectures for this, which we also utilize for our proposed model in Chapter 3.

2.2.3 Graph Neural Networks

Main expectation of having deep graph encoders is to follow deep Computer Vision architectures like CNN (LeCun, Y. Bengio, and Hinton, 2015) and obtain a scalable and a generalizable model which utilizes all available features of the input and optimized to cater the unique characteristics of graphs. Unlike images, graphs do not have a fixed structure and different graphs in a dataset could be different in size and structure. Graphs also do not have any node ordering, it only has the notion of neighbourhood. Neighbours of a given node v, $\mathcal{N}(v)$ is a set which has no order. Hence, permutation invariance is an essential requirement for a deep graph encoder.

Graph Neural Network (GNN) was first proposed by extending the message passing idea of convolution operations in CNN to arbitrary structures (Thomas N. Kipf and Welling, 2017). Core idea of GNN is, for all nodes in a graph to obtain information from all its neighbours as messages, aggregate them together and combine them to already available information in the node and get a new node representation. The neural network structure which processes information of each node in a graph is defined by the structure of the neighbourhood of each given node. Given a node v, it is going to get information from its neighbours $\mathcal{N}(v)$, its neighbours going to get information from their own neighbours $\mathcal{N}(\mathcal{N}(v))$ etc. and the neural network should learn how to propagate these information following the adjacency matrix A and aggregate and combine them to finally obtain the node embedding $\mathbf{z}_v$. Due to the arbitrary structure of graph, we can understand that each node has different neighbourhoods, hence the neighbourhood structure differs, so as the neural network architecture which processes its information. However, the basic components used for aggregating and combining information is shared across the entire graph.

Therefore, we can define the basic operation of a single GNN layer as a two step function. A GNN contains many number of layers to propagate information through neighbours across multiple hops. Mathematically, n^{th} layer of a GNN can be defined in general as

$$\mathbf{a}_v^{(n)} = \text{AGGREGATE}^{(n)}\left(\left\{\left(\mathbf{h}_v^{(n-1)}, \mathbf{h}_u^{(n-1)}, e_{vu}\right) : u \in \mathcal{N}(v)\right\}\right) \tag{2.2}$$

$$\mathbf{h}_v^{(n)} = \text{COMBINE}^{(n)}\left(\mathbf{h}_v^{(n-1)}, \mathbf{a}_v^{(n)}\right) \tag{2.3}$$

where $\mathbf{h}_v^{(n)} \in \mathbb{R}^d$ is the feature vector of node $v \in V$ at the n^{th} layer after propagating information from its neighbours $u \in \mathcal{N}(v)$. $\mathbf{e}_{vu}$ is the feature vector of the edge between u and v where $(v, u) \in A$. $\mathbf{h}_v^{(0)}$ is often initialized with node features. For a N-layer GNN, $\mathbf{z}_v = \mathbf{h}_v^{(N)}$.

Based on the choice of AGGREGATE and COMBINE functions, several variants of GNN were proposed in the literature (Thomas N. Kipf and Welling, 2017; Yujia Li, Tarlow, Brockschmidt, and R. S. Zemel, 2016; Hamilton, Ying, and Leskovec, 2017; Defferrard, Bresson, and Vandergheynst, 2016; Veličković et al., 2018; K. Xu et al., 2019). We discuss four of the most popular and widely used GNN variants below.

1. **Graph Convolutional Network (GCN).** GCN (Thomas N. Kipf and Welling, 2017) does not have two separate steps as AGGREGATE and COMBINE. All the neighbours including the node itself is aggregated together using mean pooling as:

$$\mathbf{h}_v^{(n)} = \text{ReLU}\left(\text{MEAN}\left\{\mathbf{h}_u^{(n-1)} : u \in \mathcal{N}(v) \cup \{v\}\right\} \cdot \mathbf{W}\right), \tag{2.4}$$

   where $\mathbf{W} \in \mathbb{R}^{d \times d}$ are learnable parameters.

2. **GraphSage.** GraphSage (Hamilton, Ying, and Leskovec, 2017) has several variants of aggregating methods. We indicate the max pooling based AGGREGATE here. The COMBINE operation is concatenation ($[\cdot]$).

$$\mathbf{a}_v^{(n)} = \text{MAX}\left\{\text{ReLU}\left(\mathbf{h}_u^{(n-1)} \cdot \mathbf{W}_1\right) : u \in \mathcal{N}(v)\right\}, \tag{2.5}$$

$$\mathbf{h}_v^{(n)} = [\mathbf{h}_v^{(n-1)}, \mathbf{a}_v^{(n)}] \cdot \mathbf{W}_2, \tag{2.6}$$

   where $\mathbf{W}_1 \in \mathbb{R}^{d \times d}$ and $\mathbf{W}_2 \in \mathbb{R}^{2d \times d}$ are learnable parameters.

3. **Graph Attention Network (GAT).** GAT (Veličković et al., 2018) calculates attention weights of all the neighbours w.r.t. the current node and uses weighted sum as the AGGREGATE operation. Attention weights for all the neighbours are calculated as:

$$\alpha_{v,u}^{(n)} = \frac{\exp\left(\text{LeakyReLU}\left([\mathbf{h}_v^{(n-1)} \cdot \mathbf{W}_1, \mathbf{h}_u^{(n-1)} \cdot \mathbf{W}_1]\mathbf{a}^\top\right)\right)}{\sum_{k\in\mathcal{N}(v)\cup\{v\}} \exp\left(\text{LeakyReLU}\left([\mathbf{h}_v^{(n-1)} \cdot \mathbf{W}_1, \mathbf{h}_k^{(n-1)} \cdot \mathbf{W}_1]\mathbf{a}^\top\right)\right)} : u \in \mathcal{N}(v) \cup \{v\}, \tag{2.7}$$

where $\mathbf{a} \in \mathbb{R}^{1\times d}, \mathbf{W}_1 \in \mathbb{R}^{d\times d}$ are learnable parameters and $[\cdot]$ indicates the concatenation. Same as GCN, GAT also combines AGGREGATE and COMBINE functions together as:

$$\mathbf{h}_v^{(n)} = \text{ReLU}\left(\sum_{u\in\mathcal{N}(v)\cup\{v\}} \alpha_{v,u}^{(n)} \mathbf{h}_u^{(n-1)} \cdot \mathbf{W}_2\right), \tag{2.8}$$

where $\mathbf{W}_2 \in \mathbb{R}^{d\times d}$ are learnable parameters.

4. **Graph Isomorphism Network (GIN).** GIN (K. Xu et al., 2019) opted for a simple but more powerful GNN than GCN, which has proven to be as powerful as Weisfeiler-Lehman graph isomorphism test (Weisfeiler and Leman, 1968). AGGREGATE of GIN is summation of neighbour embeddings and the COMBINE is summation followed by a Multi Layer Perceptron (MLP) based projection. ϵ can be either a fixed scalar or a layer-wise learnable parameter.

$$\mathbf{a}_v^{(n)} = \sum_{u\in\mathcal{N}(v)} \mathbf{h}_u^{(n-1)} \tag{2.9}$$

$$\mathbf{h}_v^{(n)} = \text{MLP}^{(n)}\left((1+\epsilon^{(n)}) \cdot \mathbf{h}_v^{(n-1)} + \mathbf{a}_v^{(n)}\right) \tag{2.10}$$

2.2.4 Readout functions for graph embedding

Node embeddings output by GNN can directly be used for node-level tasks such as node classification or clustering. However, for graph-level tasks, we need a mechanism to summarize the entire graph to a single embedding vector. After using a deep graph encoder like GNN to obtain node embedding, most work use those output node embeddings and accumulate them using a readout function to obtain graph embedding $\mathbf{z}_G$. One important design requirement of a graph readout/pooling method is that it should be permutation invariant as the nodes in a graph do not have an order. Some of the available graph readout methods are sum pooling, mean pooling, set2set (Vinyals, S. Bengio, and Kudlur, 2016), DiffPool (Ying et al., 2018) and MultiSet Transformer (Baek, Kang, and Hwang, 2021).

2.2.5 Graph encoder training

A similarity function is required to train these graph encoders so that the nodes and graphs which are similar in the input space would remain similar in the embedding space. For supervised setting, where node and graph labels are available, the similarity is calculated using a loss function such as cross entropy between actual labels and predicted labels of the embeddings using a classifier. For unsupervised learning, when node or graph labels are unavailable, similarity functions calculated against input space such as adjacency matrix reconstruction loss or contrastive loss would be employed.

One challenge deep learning faces when training models with lots of parameters is that, they need huge datasets with clean labels in the supervised setting. This is faced by deep graph encoders as well. However, compared to other modalities such as images and text, it is not trivial even for humans on how to annotate graphs due to complex relationships and advanced knowledge been represented in graphs. For many domains where graph deep learning is actively used to improve the efficiency reducing costs such as bioinformatics, material science, pharmaceuticals and quantum physics, annotating data is very expensive as it needs expert knowledge (K. Yang et al., 2019; Wieder et al., 2020; F.-Y. Sun et al., 2020). Even with experts, there is a tendency that those annotations could be biased towards the experts' preference (Merkwirth and Lengauer, 2005). Hence training embeddings with such labels makes the embeddings specialized only for a particular task and less generalizable.

To address this shortcoming, unsupervised graph representation learning mainly focuses on extracting as much information as possible from the input graph and incorporating them in the embedding without considering any task specific labels. There are many methods proposed for unsupervised graph representation learning which we discuss in the next section.

2.2.6 Unsupervised graph representation learning

Contrastive Learning. The most recent family of graph embedding methods are based on contrastive learning. Main idea is to train an encoder model to make it learn the contrast in between a representation which captures the structural and statistical information provided by original data and a negative sample. InfoGraph by F.-Y. Sun et al. (2020) was the first graph-level embedding model which utilized contrastive learning and this method was inspired by Infomax principle (Linsker, 1988) based Deep Graph Infomax (DGI) (Velickovic, Fedus, et al., 2019) for node-level unsupervised embedding learning. It draws negative samples from other graphs and sum pooling is used as the readout function. Infomax principle based and contrastive learning based methods have produced the best performance for graph embedding models so far.

This basic idea of contrastive learning for unsupervised graph and node embeddings has improved in multiple directions in recent work. First is, using multiple views of each input graph as positive samples. Data augmentation is a very popular approach for this. Multi-view contrastive (CMV) learning method (Hassani and Khasahmadi, 2020) is the first work on this direction. It enhances InfoGraph by introducing multi-view based data augmentation mechanism which uses contrastive learning to maximize mutual information among multiple structural views of the input graph. Next,

GraphCL (You, T. Chen, Sui, et al., 2020) analyzes multiple potential graph augmentation mechanisms and reports that different datasets require different types of augmentations to achieve optimal performance. Unlike images, where common augmentation mechanisms work for any dataset, this is a quite challenging setup to find which augmentation works for which dataset. Zeng and Xie (2021) propose to use a sequence of similar augmentations for obtaining positive samples for constrastive learning. To address the problem of manually selecting dataset specific augmentation methods, You, T. Chen, Shen, et al. (2021) have proposed automating augmentation selection mechanisms and have obtained promising results.

On the other hand, several other work have used contrastive learning to pretrain on huge unlabelled datasets and transfer that knowledge to downstream tasks. Graph Contrastive Coding (GCC) (Qiu et al., 2020) uses this approach.

One drawback of all these contrastive learning methods is that they rely heavily on the selection of negative samples. The performance varies based on the quality of the selected negative samples. To address this on node-level, M. Jin et al. (2021) propose to use a multi-scale siamese network to generate multiple views and use a cross view approach to learn representations for node-level tasks.

Main drawbacks we observe in constrastive learning for graph-level representation learning methods are as follows. First is, their high dependence on finding effective negative samples. The learnt graph representations depends on from which graphs it should differ. If the negative sample set contains similar graphs to current graph, the learnt representation becomes sub optimal. Next is, unlike Computer Vision, the used augmentation methods are not matured enough to completely rely on their ability to capture the most important semantics of the graphs. Finally, this line of work does not have the capability to extract the relevant features eliminating other noise in order to obtain more robust representations. Our proposed model in Chapter 3 addresses these weaknesses without relying on other samples on the dataset or unreliable data augmentations.

Kernel Methods. Graph kernels have been widely used for representing graph structured data over decades. Main idea of graph kernels is to first find out best substructures which the graphs can be divided into and then enumerate and count the occurrences of these sub-structures to represent them as a high dimentional feature vector. Most common substructures are walks (Gärtner, Flach, and Wrobel, 2003), shortest paths (Borgwardt and Kriegel, 2005), subtrees (Shervashidze, Schweitzer, et al., 2011), or graphlets (Shervashidze, Vishwanathan, et al., 2009). Then, graph kernels are defined to calculate pair-wise substructure similarity between two given graphs. Earlier graph kernels decoupled the process of kernel based data representations and task based model training into two parts. Therefore, unlike GNNs, kernels can neither produce task dependent features, nor can train end to end. However, kernels have more expressive power and regularize properly than GNNs. More recent kernel based models like GCKN (D. Chen, Jacob, and Mairal, 2020) have tried to combine the best of both kernal and GNN worlds by extending convolution kernel networks. However, still these kernel based methods rely on enumeration of the substructure occurrences in graphs, where they gain better expressive power at the cost of efficiency. Kernel based

methods which use walk kernels are the most similar to GNN as GNN also uses walks for information propagation.

Skip-gram influenced Methods. Another set of methods were proposed inspired by the word2vec skip-gram model (Mikolov et al., 2013) from Natural Language Processing to encode neighbourhood information to a vectorized representation and consider it as the graph embedding. First models were node2vec (Grover and Leskovec, 2016) and sub2vec (Adhikari et al., 2018) which use random walks to identify each node's neighbourhood and encode that to a latent vector to represent each node or sub-graph respectively. graph2vec (Narayanan et al., 2017) uses Weisfeiler-Lehman kernel (Shervashidze, Schweitzer, et al., 2011) to calculate non-linear substructures opposed to linear sub-structures from sub2vec to vectorize full graphs. However, these methods are completely dependent of neighbourhood information and unable to utilize node features.

Autoencoder Based Methods. There are no autoencoder based models specifically aimed at graph-level representation learning. All existing methods (Thomas N. Kipf and Welling, 2016b; Pan et al., 2018; Park et al., 2019) are aimed at independent node information modelling and proposing better decoders for learning the graph structure. Main drawback of existing autoencoder based models are that they overemphasize proximity information (Hassani and Khasahmadi, 2020) and Yonglong Tian, Krishnan, and Isola (2020) mention, in normal autoencoders every feature is treated equal, not one set of features is important than the others. Feature importance is required for using these embeddings in downstream tasks. We propose an advanced graph autoencoder architecture in Chapter 3, which addresses both these limitations by pushing the model to extract a global component common to all nodes, so that there is a special component which is important and different from all other remaining information. Also, our model has an auxiliary objective for sharing information across graph despite proximity.

Other Methods. Inter-graph similarity based learning for graph-level representations is another emerging line of work. Bai et al. (2019) follow a pair-wise graph similarity calculation mechanism based on proximity calculations using Graph Edit Distance (Sanfeliu and Fu, 1983) for unsupervised graph representation learning. M. Xu et al. (2021) propose a hierarchical prototype based clustering mechanism to share similarity information across all samples of the dataset. They learn this as a pre-training step, and finetune with downstream labeled tasks. Graph coarsening is another direction for graph representation learning and T. Ma and J. Chen (2021) have followed that approach for graph representation learning with promising results. Similar to kernel methods, most of these methods require pair-wise graph similarity calculations which make these approaches less scalable and highly expensive.

2.3 Applications of Graph Deep Learning

In the scope of this thesis, our focus is on applying graph deep learning to Vision and Language domain. We selected this domain because, Computer Vision (CV) and Natural Language Processing (NLP) are two of the main fields of Artificial Intelligence

where human excels in, but deep learning and machine learning in general still struggles. Vision and Language is the combined research domain which aims at solving tasks which require understanding of both Computer Vision and Natural Language. There are so many real world applications of this domain, specially for people who have visual, hearing or speech impediment (Gurari et al., 2018; Zaman et al., 2019).

Motivated by these real life use cases, we are interested in how graph deep learning can be utilized to improve Vision and Language tasks. To provide a background, we start by discussing how graph deep learning is applied on images and text separately to understand the basic relational information contained in these data modalities. We then move to discuss Vision and Language multi-modal problems which were solved using graph learning methods and highlight the gaps in existing work.

Computer Vision Applications. In Computer Vision, images can be modelled as graphs in several different ways. First is in pixel level, where each pixel is considered as a node and edges are formed if two pixels are next to each other based on its spatial location. This was further improved by the superpixel concept which groups nearby pixels which are similar together and again use spatial relationships to create edges between superpixels and form graphs (Monti et al., 2017). Next is, a scene in an image can be modelled as a graph where objects are nodes and object relations including spatial, action and semantic connections such as “book-on-table" or “man-ride-bicycle" or “woman1-daughterof-woman2" can be used to form triplets (subject-predicate-object) and these are used to form scene graphs (D. Xu, Y. Zhu, Christopher B. Choy, et al., 2017b). A human’s poses can also be modelled as graphs (J. Wang et al., 2020) where specific key joints of human body with their position coordinates are treated as nodes and how the joints are connected forms the edges. Based on different poses of human, the output graph would vary. By encoding these graphs, position estimation models try to predict the human pose. Another Computer Vision application is 3-dimensional (3D) point cloud representation learning (Y. Wang et al., 2019; Pistilli et al., 2020; Yifei Tian et al., 2021). Point clouds are used to represent objects where with 3D points in the space are nodes and the graph is formed using the topology structure of the nodes. Graph encoders are mainly use to extract topology related information from the point clouds and use them for tasks like object detection and denoising.

Natural Language Applications. A sentence can be modelled as a dependency tree (Kulick, Bies, and Mott, 2012) where words are nodes and grammatical relations among words form edges. A sentence can also be modelled as a directed acyclic graph in the semantic level using Abstract Meaning Representation(AMR) (Sheng Zhang et al., 2019), where concepts of the sentence are nodes and edges are formed for semantic relations among concepts. Graph encoding based dependency parsing (Nivre and McDonald, 2008; T. Ji, Y. Wu, and Lan, 2019) and AMR parsing (Flanigan et al., 2014; D. Cai and Lam, 2020) are widely used techniques in semantic parsing domain of NLP. Next, a document can be modelled as a graph by extracting the most important sentences from the document and forming them as nodes and their conceptual relations as edges (Yasunaga et al., 2017). This graph based document representation is used for tasks like document summarization (Tan, Wan, and Xiao, 2017; Luo, S. Zhao, and Z. Cai, 2021) and sentiment analysis (Veyseh et al., 2020; Meng et al., 2020). A discussion

thread can also be modelled as a graph where the responses from users of the thread are nodes and edges are formed if one response is a reply to another (Hamilton, Ying, and Leskovec, 2017). These graph based modelling are used for tasks like forum discourse structure classification (L. Wang et al., 2011) and conversation summarization (Fabbri et al., 2021).

Vision and Language Applications. Inspired by the applied graph learning techniques of CV and NLP, Vision and Language multi-modal domain has also tried to model their problems as graphs. For tasks like visual question answering (Teney, L. Liu, and Hengel, 2017) and visual commonsense reasoning (W. Yu et al., 2019), the input questions, images and available concepts are modelled as graphs and graph alignment mechanisms are used to match the text with objects in the scenes to find answers. For image captioning, scene graph decomposition has proven to be very effective in order to extract most important information from an image along with semantic relationships, which are normally not available from a raw image (X. Yang et al., 2019; Zhong et al., 2020). Video is a modality that Vision and Language both are essentially incorporated. Therefore we mention Vision and Language tasks related with video also under this section. Zhiwang Zhang et al. (2021) propose a graph based approach for dense video captioning where they model relations among temporally evolving video segments as a graph and use aggregation methods to extract semantic words to generate captions. For situation recognition, a structured prediction task for action and semantic roles of a given image, R. Li et al. (2017) propose to model the semantic role structure of each action as a fully connected graph and use GNN to propagate information and obtain final role embeddings for label prediction.

One observation common for all these Vision and Language applications is that, they apply graph learning techniques either after completing multi-modal fusion, only to learn structure based relations among different elements of the graph or align separate graphs created for each modality to find object and text matches. They have not considered how multi-modal fusion can be incorporated in graph encoding algorithms for steps like AGGREGATE and COMBINE so that the inter-dependency among elements in the graph could be used to determine the amount of information required from each modality in the fusion operation. This could also be useful to decide how much information to be incorporated from the neighbours without directly combining all of them and causing bias towards frequently seen node co-occurrences of graphs hindering rare but accurate scenarios. We explore this direction in Chapter 5 and obtain very encouraging results to validate our motivation.

2.4 Attention based Vision and Language fusion

Multi-modal fusion strategies play a prominent role in Vision and Language tasks as both modalities carry important information essential for the final output. As mentioned in Section 2.3, we focus on Vision and Language domain for graph learning applications. Attention is a concept which is widely used in this domain as it facilitates determining the importance of each component during fusion. Attention based methods achieve the best performance in multi-modal fusion. Hence, here we provide an

overview of some of the widely used attention based multi-modal fusion mechanisms. Here, we discuss about image and text fusion methodologies.

First, images and natural language text need to be encoded with their modal specific deep learning strategies to extract the most important features from each input modality independently. For image encoding, there are two widely popular mechanisms. First is, the grid based encoded image output and the second is, object detection based region encoded output. For grid based output, a CNN is used to extract features in the form of height $\times$ width $\times$ channel count. For VGG-16 (Simonyan and Zisserman, 2015), this would be a $7 \times 7 \times 512$. Therefore a 7×7 grid, where each element in the grid has the feature size of 512. For object detection based regions, first an object detection model is employed such as Faster R-CNN (Ren et al., 2015) or RetinaNet(T.-Y. Lin, P. Goyal, et al., 2017) or YOLO (Bochkovskiy, C.-Y. Wang, and Liao, 2020) to extract image regions with salient objects. Then, each N regions with d features (E.g., $N = 100, d = 1024$) are used as input to fusion method. Object detected regions carry much more powerful features than grid based regions as grids encode the entire image without removing irrelevant information like background. However, when your target dataset contains long tailed object distributions which are rarely seen and do not have many samples from each object, it is very challenging to train an accurate object detection model. In such scenarios, using grid based regions would be more effective. For encoding text, a recurrent neural network like LSTM (Hochreiter and Schmidhuber, 1997) or a self-attention model such as Transformer (Vaswani et al., 2017) can be used. These enable either to obtain word-wise embeddings or a single embedding for the entire text.

There are several methods on how to fuse image and text together once image regions and encoded text are available. Top-Down Attention (TDA) (Anderson et al., 2018) uses attention mechanism to weight each image region using the entire sentence/text embedding to determine the similarity of each region to the text. Then the attention weighted sum of image regions and the text embedding are combined together using dot-product similarity to obtain the final fused multi-modal representation. TDA only focuses on identifying the most important image regions to match with the text, it does not have a mechanism to identify the important parts of the text to be included in the fused outcome. The concept of co-attention (Nam, Ha, and J. Kim, 2017; Z. Yu et al., 2018) was proposed to add attention to textual words as well as image regions to address this. They calculate attention distributions for textual input independently without input from image and use the attention weighted text to identify salient regions from image using attention and fuse them together. Bilinear attention network (J.-H. Kim, Jun, and B.-T. Zhang, 2018) further improves this by incorporating interactions among textual and visual features in their attention mechanism. They consider pair of each textual word and visual region and use low-rank bilinear pooling (J.-H. Kim, On, et al., 2017) to get a joint representation to each pair. Bilinear attention was calculated on top of that to extract most relevant visual features for the text and obtain the final output.

Chapter 3

Graph-wise Common Latent Factor Extraction for Unsupervised Graph Representation Learning

3.1 Introduction

Graph structured data has been very useful in representing a variety of data types including social networks (Newman and Girvan, 2004), protein-protein interactions (Krogan et al., 2006), scene graphs (Krishna, Y. Zhu, et al., 2016), customer purchasing patterns (Bhatia et al., 2016) and many more. Graph Neural Networks (GNNs) have recently become the prominent approach for representing graph structured data (Yujia Li, Tarlow, Brockschmidt, and R. S. Zemel, 2016; Gilmer et al., 2017; Thomas N. Kipf and Welling, 2017; Velickovic, Cucurull, et al., 2018; K. Xu et al., 2019). GNNs are capable of representing graphs in a permutation invariant manner, enabling information propagation among neighbours and mapping graphs to low dimensional spaces.

In this chapter, our main focus is on graph-level representation learning. Graph-level representation learning is crucial for tasks like molecular property identification (Duvenaud et al., 2015) and community classification based on the patterns of discussion threads (Yanardag and Vishwanathan, 2015), and they are useful for applications such as drug discovery, material design and recommendation systems. Availability of task specific labels plays a significant role in graph representation learning as much as its role in other domains such as images, text and speech. However, due to many specialized fields which graphs are utilized (e.g., biological sciences, quantum mechanics, chemical informatics), collecting labels has become very expensive as it needs expert knowledge (K. Yang et al., 2019; Wieder et al., 2020; F.-Y. Sun et al., 2020). Therefore, unsupervised graph representation learning has become crucial.

Unsupervised graph level representation learning has a very rich literature consisting of several main directions. Skip-gram influenced graph embedding methods (node2vec (Grover and Leskovec, 2016), sub2vec (Adhikari et al., 2018), graph2vec (Narayanan et al., 2017)) only rely on neighbourhood information and loses the advantage of using node features, making them less effective. Kernel methods (Random Walk (RW) (Gärtner, Flach, and Wrobel, 2003), Shortest Path (SP) (Borgwardt and Kriegel, 2005), Graphlet Kernel (GK) (Shervashidze, Vishwanathan, et al., 2009), GCKN (D. Chen, Jacob, and Mairal, 2020)) use pair-wise graph similarity making them more effective, but less efficient. Quality of learnt embeddings by this method heavily rely on the quality and variety of other graphs in the batch which it compares in pairs. Contrastive

learning (InfoGraph (F.-Y. Sun et al., 2020),CMV (utilizing multiple views) (Hassani and Khasahmadi, 2020), GCC (utilizing extra data) (Qiu et al., 2020) and GraphCL (utilizing data augmentations) (You, T. Chen, Sui, et al., 2020)) is the newest addition which is based on the Infomax principle (Linsker, 1988) which aims at obtaining an output which has maximum mutual information with the input. Main drawback of contrastive based learning methods (Grill et al., 2020) is that their heavy reliance on the selection procedure of negative samples for model performance. A careful selection of task-wise negative samples is required in order to obtain good performance. While both inter-graph similarity and contrastive methods achieve state-of-the-art for graph embedding learning, both of them suffer a lot if the quality of other graphs they compare with are low.

Autoencoder (Baldi and Hornik, 1989; Hinton and R. S. Zemel, 1993) based embedding methods solve this weakness by only utilizing its current input for representation learning. However, existing graph autoencoder models (Thomas N. Kipf and Welling, 2016b; Pan et al., 2018; Park et al., 2019) are only aimed at node level modelling and not aimed at graph level. These methods overemphasize proximity information (Hassani and Khasahmadi, 2020) in the learnt latent as the model optimization mainly relies on adjacency matrix reconstruction. Also they tend to treat all features equal (Yonglong Tian, Krishnan, and Isola, 2020), incapable of differentiating features required for graph level tasks from node specific local factors. Feature differentiation is very important when using embeddings in downstream tasks, because equally treated features could add noise and redundancy which leads to performance degradation.

These weaknesses of existing work motivate us to research on an approach which could both differentiate features crucial for graph level representation as well as capable of learning embeddings by only utilizing the current input sample. Although Graph Variational Autoencoder (GVAE) (Thomas N. Kipf and Welling, 2016b)(and its all existing variants such as (Pan et al., 2018; Park et al., 2019)) is inadequate to fulfill graph level feature differentiation, it empowers single sample based learning. Hence, we are motivated to follow a generative based mechanism while addressing the specific requirements to obtain a discriminative graph representation which GVAE lacks on.

Graph-wise common latent factor extraction (GCFX) Motivation. To draw inspiration to our generative based approach, we observe two real world graph formation examples; one from public discussion forums and another from chemistry. An online discussion thread can be represented as a graph where nodes represent users who have participated in a discussion thread, and edges represent interaction among users in the thread (Yanardag and Vishwanathan, 2015). This graph initializes with a single user who wants to discuss a particular topic and grows with nodes when subsequent users start responding about this topic. For the second example, a chemical compound can be represented as a graph where the nodes are atoms and edges are chemical bonds. Inverse molecule design (Sanchez-Lengeling and Aspuru-Guzik, 2018; Kuhn and Beratan, 1996; Zunger, 2018) is a molecule generation procedure which is initiated with the desired properties to be included in a molecule such as redox potential, solubility and toxicity. De novo (Schneider, 2013; Brown et al., 2019) inverse molecule design method initiates with the desired ranges of those properties and iteratively add atoms and chemical bonds conditioned on those properties to form molecular graphs.

Key observation from these examples is that, each node and edge added to the structure to form the graph was conditioned on one or more global graph factors. Topic is the common global factor for all elements of the discussion thread and toxicity and solubility levels are common for entire chemical compound. We can see that although each node have its own specific information such as personal details of a user or properties of an atom, the common factor is the one which differentiates this graph from another. Hence, useful for tasks like community identification and molecule selection for drug design.

Graph-wise common latent factor extraction. Motivated by this, we hypothesize extracting these common factors could be highly beneficial for a discrminative graph representation. Hence, this work proposes graph-wise common factor extraction in a latent manner. We further propose deepGCFX: a novel autoencoder based architecture which can explicitly extract common latent factors from the entire graph incorporating feature differentiation to autoencoders. Our enhanced decoding mechanism which enforces utilizing common latent factors for graph reconstruction, also regularizes normal autoencoders' heavy dependence on proximity.
We summarize contributions of this chapter as follows:

- We propose GCFX: a novel principle for unsupervised graph representation learning based on the notion of graph-wise common latent factors inspired by real-world examples.
- Existing autoencoder models are unable to learn graph-wise common factors due to their inability of feature differentiation. Therefore we propose deepGCFX: a novel autoencoder based approach with iterative query based reasoning and feature masking capability to extract common latent factors.
- We empirically demonstrate the effectiveness of deepGCFX in extracting graph-wise common latent factors.
- For the best of our knowledge, this is the first graph embedding learning method based GCFX. We show that deepGCFX can achieve state of the art results in *unsupervised graph level representation learning* as shown in standard downstream tasks.
- By extracting common factors from non-common latent, deepGCFX enables long distance inter-node information sharing ability to node representation learning achieving best results for unsupervised node representation learning on disassortative graphs.

3.2 Related Work

Unsupervised Graph Level Representation Learning Contrastive Learning is the most recent family of graph embedding methods where Infograph (F.-Y. Sun et al., 2020) was the very first model which proposed to learn a graph embedding by maximizing the mutual information of the vectorized graph embedding with its own patch embeddings while contrasting with other graphs by considering them all as negative samples. This initial idea was further improved in two main directions; First is utilizing graph augmentation (Hassani and Khasahmadi, 2020; You, T. Chen, Sui, et al.,

2020; Zeng and Xie, 2021; You, T. Chen, Shen, et al., 2021) for better graph embedding learning and the other is pre-training on large scale datasets (Qiu et al., 2020) to transfer knowledge. However the main drawback of contrastive based methods (Grill et al., 2020) is that their heavy reliance on the selection procedure of negative samples for model performance. A careful selection of task wise negative samples is required in order to obtain good performance. deepGCFX does not have this drawback as the entire learning process solely depends on the graph at hand. Kernel methods is the most prominent line of work for pairwise similarity based graph embedding learning. Main idea of graph kernels is to first find out best sub-structures which the graphs can be divided into and then enumerate and count the occurrences of these sub-structures to represent them as a high dimensional feature vector. Most common substructures are walks (Gärtner, Flach, and Wrobel, 2003), shortest paths (Borgwardt and Kriegel, 2005), subtrees (Shervashidze, Schweitzer, et al., 2011), or graphlets (Shervashidze, Vishwanathan, et al., 2009). More recent kernel based methods have combined GNN with original kernels (D. Chen, Jacob, and Mairal, 2020). However, these methods rely on enumeration of substructure occurrences in graphs, which provides them better expressive power over GNNs at the cost of efficiency. Other than kernels, there are other available inter-graph similarity based graph embedding methods as well (Bai et al., 2019; M. Xu et al., 2021). Although our objective is different from theirs and less expensive due to irrelevancy of explicit inter-graph similarity calculations, it is complementary to ours. Finally, skip-gram influenced graph embedding methods (Grover and Leskovec, 2016; Adhikari et al., 2018; Narayanan et al., 2017) only rely on neighbourhood information and loses the advantage of using node features, making them less effective.

Generative based Latent graph representation Learning Approaches In comparison to above methods, unsupervised graph representation learning domain has been nurtured by models with generative objectives as well. Main objective of these methods were not to generate realistic graphs, but to improve the quality of the latent compact representations by enhancing its capability of reconstructing the input graph. Autoencoders (Baldi and Hornik, 1989; Hinton and R. S. Zemel, 1993) are the mostly proposed architecture for this specific purpose. There are no autoencoder based models specifically aimed at graph level representation learning. All existing methods (Thomas N. Kipf and Welling, 2016b; Pan et al., 2018; Park et al., 2019) are aimed at independent node information modelling and proposing better decoders to enforce the latent to store better structural information. Main drawback of existing autoencoder based models are that they overemphasize proximity information (Hassani and Khasahmadi, 2020) and Tian et al. Yonglong Tian, Krishnan, and Isola (2020) mentions in normal autoencoders, every feature is treated equal as there is no mechanism to differentiate feature importance. Differentiating features is important when using these embeddings in downstream tasks as equally treated features could add noise and reduce performance. However in deepGCFX we address both these drawbacks by pushing the model to extract a latent factors common to all nodes, so that there is a common factor which is important and different from all other remaining local factors. Also extracting this common factor implicitly acts as a regularizer for the final proximity based reconstruction objective.

Graph Property Recovery Methods Main difference of our work from all existing unsupervised methods is that we propose to analyze the potential of common latent factors to become effective graph embeddings. This involves extracting common factors, making our method related to following areas; Embedding Inversion, Graph Recovery and Disentanglement Learning. Embedding Inversion is aimed at identifying which information are encoded from the original graph in an embedding and Graph Recovery focuses on evaluating how much structural similarity the reconstructed graph from that embedding has with the original (Chanpuriya, Musco, Sotiropoulos, and C. Tsourakakis, 2021; Chanpuriya, Musco, Sotiropoulos, and C. E. Tsourakakis, 2020; Hoskins et al., 2018; McGregor, 2014). While both these approaches are aiming only at learning embeddings with sufficient factors for successful recovery of original graph, our model has a very important main objective to enforce learning to explicitly extract common factors shared across entire graph. Above methods focus only including relevant factors in the latent embedding to increase graph recoverability, they do not consider whether those factors are entangled in the embedding. Disentanglement Learning (Desjardins, Courville, and Y. Bengio, 2012) tries to make the latent embedding factors disentangled with respect to factors of variation of input data. Existing graph related work have proposed to disentangle either neighbourhood information (J. Ma et al., 2019; Y. Liu et al., 2020) or node and edge features (Y. Yang et al., 2020; X. Guo et al., 2020) from graphs. Currently available unsupervised graph disentanglement learning (X. Guo et al., 2020) only aims at disentangling structure related factors such as nodes and edges. deepGCFX only requires extracting common latent factors filtering out local factors. It does not aim at optimizing the model to increase the mapping between latents and ground truth generative factors. However when ground truth common factors are known, we demonstrated in the evaluation section that, extracted common latents can capture them and transfer to other samples as an added bonus of our deepGCFX.

3.2.1 Main differences between GCFX principle and other main principles for graph representation learning

Contrastive learning and inter-graph similarity are the two main underlying principles in which models achieve state of the art performance in graph level downstream tasks at the moment. In this work we propose GCFX principle to address the common drawback of both of these methods; heavy reliance on other graph samples to learn better graph embeddings. In Table 3.1 we compare one baseline model from each principle which do not use any augmentations or special improvements and discuss what are the main differences among these principles. From that we can observe that not only GCFX principle and our proposed deepGCFX model remove state-of-the-art unsupervised graph representation learning models' overemphasis on depending on other samples for better performance, but it also explicitly enables removing irrelevant information from the embedding. Both Infograph and GCKN only aim at incorporating features from input graph to the embedding, they have no explicit mechanism to remove original features from input graph, which are irrelevant for the embedding. Our deepGCFX has that capability and it is beneficial in certain downstream tasks as shown in the evaluation section.

TABLE 3.1: Characteristic Comparison of DeepGCFX with Infograph (F.-Y. Sun et al., 2020) (the base graph embedding model for contrastive learning) and GCKN (D. Chen, Jacob, and Mairal, 2020)(State-of-the-art Kernel method)

	INFOGRAPH	GCKN	DEEPGCFX (OURS)
Underlying Principle	- Infomax Principle	- Inter graph similarity calculation	- GCFX principle
Fundamental idea	- Maximize the mutual information between original input graph and output latent graph embedding	- Minimize pair-wise graph distance between graph latent embeddings of each graph pair in the dataset	- Extract graph-wise common latent factors from the original graph only based on the model's reconstruction ability of the original graph.
Utilized technique to model the underlying principle	- Contrastive learning	- Kernel methods	- Iterative query based reasoning
Dependence on other data samples	- True (Quality of selected negative samples determine model performance)	-True (Variety and quality of samples used in pair-wise similarity calculation determine embedding quality)	- **False** (Only depend on input graph)
Ability to remove irrelevant information input	- False	- False	-**True** (Extracting graph-wise common factors filtering out local factors)

3.3 Methodology

3.3.1 Graph generation process

Let $\mathbb{D} = \{\mathbb{G}, C_f, L_f\}$ be the set that consists of graphs and their ground truth common and uncommon generative factors. We would call uncommon factors as local, hence the notation L. Each graph $G = (V, A)$, contains a set of nodes V and A is the adjacency matrix. C_f and L_f represent two sets of generative factors: C_f contains common factors $\mathbf{c_f} \subset C_f$ common for the entire graph (such as topic of a discussion thread or solubility level of a chemical compound) and $\mathbf{l_f} \subset L_f$ represents local factors which can differ from local node to node (ex: background information of each user participated in a discussion thread, atomic number and mass of each atom in a chemical compound). In this work we assume, $\mathbf{c_f}$ and $\mathbf{l_f}$ are conditionally independent given G, where $p(\mathbf{c_f}, \mathbf{l_f}|G) = p(\mathbf{c_f}|G) \cdot p(\mathbf{l_f}|G)$. We assume that the graph G is generated using a true world generator which uses the ground truth generative factors: $p(G|\mathbf{c_f}, \mathbf{l_f}) = \mathit{Gen}(\mathbf{c_f}, \mathbf{l_f})$.

3.3.2 GCFX: Graph-wise Common latent Factor EXtraction

We focus on the novel problem of graph-wise common latent factor extraction. Although we only focus on extracting common latent factors, identifying local factors is essential in order to filter them out. Hence, our goal is to develop an unsupervised deep graph generative model which can learn the joint distribution of graph G and the set of generative factors (includes both common and node-specific) $\mathbf{Z}$, using only samples from $\mathbb{G}$. This should be learnt in a way that the set of latent generative factors can generate the observed graph G, such that $p(G|\mathbf{Z}) \approx p(G|\mathbf{c_f}, \mathbf{l_f}) = \mathit{Gen}(\mathbf{c_f}, \mathbf{l_f})$. A suitable approach to fulfill this objective is to maximize the marginal log-likelihood for the observed graph G over the whole distribution of latent factors $\mathbf{Z}$.

$$\max_{\theta} \mathbb{E}_{p_\theta(\mathbf{Z})}[\log p_\theta(G|\mathbf{Z})] \tag{3.1}$$

For an observed graph G, the inferred posterior probability distribution of the latent factors $\mathbf{Z}$ can be described as $q_\phi(\mathbf{Z}|G)$. However, the graph generation process we described in Section 3.3.1 above assumes two independent sets of generative factors representing common and local level information relevant for a graph (from which we are explicitly interested in common factors). Therefore we consider a model where the latent factor set $\mathbf{Z}$ can be divided into two independent latent factor sets as $\mathbf{Z} = (\mathbf{Z}_c, \mathbf{Z}_l)$. $\mathbf{Z}_c$ represents the latent factors which capture the graph-wise common generative factors of G and $\mathbf{Z}_l$ captures its non-common node specific local counterpart. Therefore we can rewrite our inferred posterior distribution as follows:

$$q_\phi(\mathbf{Z}|G) = q_\phi(\mathbf{Z}_c, \mathbf{Z}_l|G) = q_\phi(\mathbf{Z}_c|G)q_\phi(\mathbf{Z}_l|G) \tag{3.2}$$

We discuss in detail these two posteriors: $q_\phi(\mathbf{Z}_c|G)$ and $q_\phi(\mathbf{Z}_l|G)$. The graph G consists of $|V|$ number of nodes. In a graph data structure, each node is not isolated. They are connected with its neighbours and propagates information. Therefore, we use the term *patch* to indicate the local neighbourhood centered at each node where the

node interacts with. Hence, $q_\phi(\mathbf{Z}_c|G)$ and $q_\phi(\mathbf{Z}_l|G)$ are the posterior distributions of all these $|V|$ patches. However, if we consider the common latent posterior, it is common for all $|V|$ patches, as the graph G was originally generated with $\mathbf{c_f}$ common for all V. Hence, we propose to use a single latent $\mathbf{z}_c$ to capture the common generative factors. In particular, we use $q_\phi(\mathbf{z}_c|G)$ to model this single posterior. On the other hand, the factors which contribute to generate each patch can vary significantly. Therefore in this model we assume the local latent factors are independent. Therefore, we update Eq. 3.2 as:

$$q_\phi(\mathbf{Z}|G) = q_\phi(\mathbf{z}_c, \mathbf{Z}_l|G) = q_\phi(\mathbf{z}_c|G) \prod_{i=1}^{|V|} q_\phi(\mathbf{z}_l(i)|G) \tag{3.3}$$

Here $\mathbf{z}_l(i)$ is the latent factor that captures the local generative factors for a patch centered at node i.

Now, our objective is to make sure the latent factors sampled from common and local latent posterior distributions can capture the common and local generative factors $\mathbf{c}_f$ and $\mathbf{l}_f$ respectively.

3.3.3 A constrained optimization formulation for GCFX

In this section we describe how to model the novel principle of GCFX described in Section 3.3.2 as a constrained optimization problem in order to solve it. First, we match common and local generative factors $\mathbf{c}_f$ and $\mathbf{l}_f$ to their respective priors $p(\mathbf{z}_c)$ and $p(\mathbf{z}_l)$ separately. We select unit Gaussians ($\mathcal{N}(0, 1)$) as priors. In particular, based on our modeling of common/local factors in Eq. 3.3, we can re-write Eq. 3.1 as follows (Higgins et al., 2017).

$$\begin{aligned} \max_{\theta,\phi} \quad & \mathbb{E}_{G\sim\mathbb{G}}\Big[\mathbb{E}_{q_\phi(\mathbf{z}_c,\mathbf{Z}_l|G)}[\log p_\theta(G|\mathbf{z}_c, \mathbf{Z}_l)]\Big] \\ \text{s.t.} \quad & KL(q_\phi(\mathbf{z}_c|G) \parallel p(\mathbf{z}_c)) < \epsilon \\ & KL(q_\phi(\mathbf{Z}_l|G) \parallel p(\mathbf{Z}_l)) < \eta \end{aligned} \tag{3.4}$$

where ϵ and η are strengths of each constraint. Following Higgins et al. (2017), Eq. 3.4 can be written to obtain the variational evidence lower bound (ELBO) of a Graph Variational Autoencoder (GVAE) (Thomas N. Kipf and Welling, 2016b) (Here we call this as GVAE because our input is a graph) with two separate latent representations with additional coefficients as follows:

$$\begin{aligned} \mathcal{F}(\theta, \phi; G, \mathbf{z}_c, \mathbf{Z}_l, \beta, \gamma) &\geq \mathcal{L}(\theta, \phi; G, \mathbf{z}_c, \mathbf{Z}_l, \beta, \gamma) \\ &= \mathbb{E}_{q_\phi(\mathbf{z}_c,\mathbf{Z}_l|G)}[\log p_\theta(G|\mathbf{z}_c, \mathbf{Z}_l)] \\ &- \beta\, KL(q_\phi(\mathbf{z}_c|G) \parallel p(\mathbf{z}_c)) \\ &- \gamma\, KL(q_\phi(\mathbf{Z}_l|G) \parallel p(\mathbf{Z}_l)) \end{aligned} \tag{3.5}$$

Based on Eq.3.3 we can expand the KL divergence term $KL(q_\phi(\mathbf{Z}_l|G) \parallel p(\mathbf{Z}_l))$ and rewrite our objective function for a single graph G as:

$$\begin{aligned}\mathcal{L}(\theta, \phi; G, \mathbf{z}_c, \mathbf{Z}_l, \beta, \gamma) = \mathbb{E}_{q_\phi(\mathbf{z}_c, \mathbf{Z}_l|G)}[\log p_\theta(G|\mathbf{z}_c, \mathbf{Z}_l)] \\ - \beta\ KL(q_\phi(\mathbf{z}_c|G)|p(\mathbf{z}_c)) \\ - \gamma \sum_{i=1}^{|V|} KL(q_\phi(\mathbf{z}_l(i)|G) \parallel p(\mathbf{z}_l(i)))\end{aligned} \tag{3.6}$$

Overall, **the learning objective of GCFX is to maximize this lower bound** for all the graphs in a minibatch $\mathbb{G}_b$ from the full dataset $\mathbb{G}$:

$$\mathcal{L}_{\theta,\phi}(\mathbb{G}_b) = \frac{1}{|\mathbb{G}_b|} \sum_{r=1}^{|\mathbb{G}_b|} \mathcal{L}(\theta, \phi; G, \mathbf{z}_c, \mathbf{Z}_l, \beta, \gamma) \tag{3.7}$$

3.4 deepGCFX: An autoencoder based approach for GCFX

Existing autoencoder models including GVAE does not have the capability to learn graph-wise common factors due to their inability to differentiate factors based on importance. Therefore we propose a deepGCFX, novel GVAE architecture based on GCFX principle with the capability of extracting graph-wise common factors. We propose an iterative query based reasoning mechanism with feature masking capabilities to achieve this ability. Figure 3.1 depicts the proposed deep Graph-wise Common Factor EXtractor (deepGCFX) model. deepGCFX consists of three main components; Graph Neural Network based encoder, Graph-wise common factor extractor and accumulator based on iterative query based reasoning and Commonality and relevance preserving decoder.

3.4.1 Graph Neural Network based encoder

We utilize a N-layer Graph Neural Network(GNN)(Thomas N. Kipf and Welling, 2017; Velickovic, Cucurull, et al., 2018; K. Xu et al., 2019) as the encoder. n^{th} layer of a GNN can be defined in general as

$$\mathbf{a}_v^{(n)} = \text{AGGREGATE}^{(n)} \left(\left\{ \left(\mathbf{h}_v^{(n-1)}, \mathbf{h}_u^{(n-1)}, e_{vu} \right) : u \in \mathcal{N}(v) \right\} \right) \tag{3.8}$$

$$\mathbf{h}_v^{(n)} = \text{COMBINE}^{(n)} \left(\mathbf{h}_v^{(n-1)}, \mathbf{a}_v^{(n)} \right) \tag{3.9}$$

where $\mathbf{h}_v^{(n)} \in \mathbb{R}^{d\_hidden}$ is the feature vector of a patch centered at node $v \in V$ at the n^{th} layer after propagating information from its neighbours $u \in \mathcal{N}(v)$. $\mathbf{e}_{vu}$ is the feature vector of the edge between u and v where $(v, u) \in A$. $\mathbf{h}_v^{(0)}$ is often initialized with node features. We use the term GNN to indicate any network which use layers described in Eq. 3.9. The neighbourhood aggregation function AGGREGATE and node update function COMBINE differs for each specific GNN architecture.

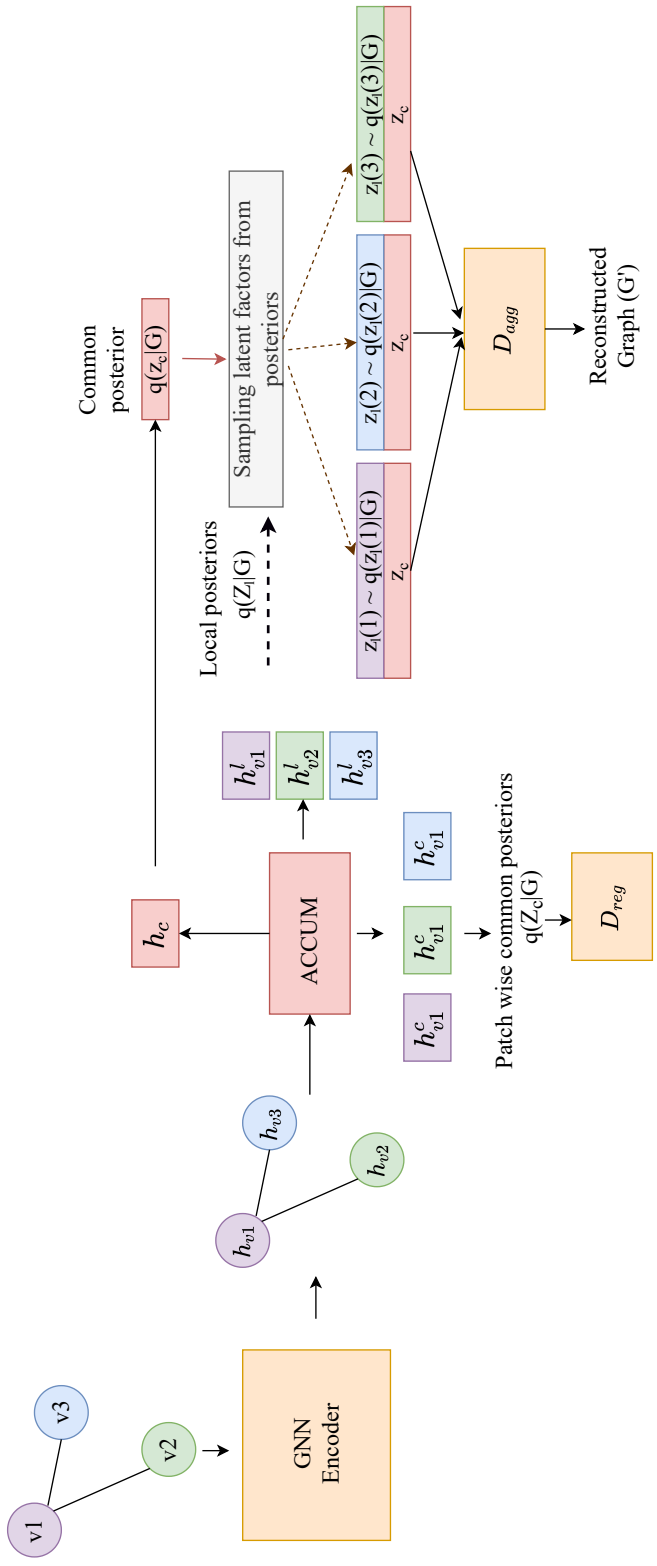


FIGURE 3.1: deepGCFX architecture: Given an input graph G, we first send it through a GNN Encoder to obtain individual node representations aggregated with neighbours, then we conduct an iterative procedure to filter graph-wise common ($\mathbf{h}^c$) and local factors ($\mathbf{h}^l$) from each node/patch in order to extract single graph-wise common latent factor representation $\mathbf{h}_c$ (This procedure is described Fig. 3.2). Sampled local latent factors from their respective posteriors $\mathbf{z}_l(j) \sim q(\mathbf{z}_l(j)|G)$, $\forall j \in \{1 \dots |V|\}$ are combined with the common latent $\mathbf{z}_c \sim q(\mathbf{z}_c|G)$, and this becomes the input to the decoder $\mathcal{D}_{agg}$ to reconstruct the graph. $\mathcal{D}_{reg}$ is used to enforce $\mathbf{z}_c$ to contain G related factors. Overall model is trained by optimizing the loss function in Eq.3.19.

3.4.2 ACCUM : Iterative query based reasoning with feature masking for common latent accumulation

Our main algorithmic invention is a novel mechanism to extract high-quality common latent factors, based on ideas of iterative query based reasoning and feature masking. As discussed, GVAE cannot extract common latent factor as it cannot differentiate feature importance. Specifically, in order to make sure the latent factors sampled from common and local latent posterior distributions can capture the common and local factors respectively, first we need a mechanism to differentiate and filter out what are common features from the output patch representations from our GNN encoder. To achieve this, we propose a novel mechanism which iteratively learns graph-wise common factors and extracts them from each patch and accumulate them to generate a single common factor embedding for each graph.

We model the common factor extraction from patches as an iterative query based reasoning problem, where our query is the accumulated common factor representation of the graph. We use that query on our input patch representations $\mathbf{h}_v^{(n)}$ (superscript will be omitted from here onwards) to determine which factors from $\mathbf{h}_v$ are similar to the existing common factors and filter them out from non-common node factors to update the accumulated common factor representation. At each iteration i, the process starts with the query $\mathbf{q_c}(i-1)$ containing graph-wise common factors extracted from all the patches at iteration $i-1$. It is used to query all patch representations $\mathbf{h}_v, v \in V$ at current iteration i to identify the amount of similarity each factor of $\mathbf{h}_v, v \in V$ have with currently extracted graph-wise common factors $\mathbf{q_c}(i-1)$ which is also our query. Factor wise similarity score for current iteration i, $\delta_v(i) \in \mathbb{R}^{d\_hidden}$ is calculated as follows:

$$\delta_v(i) = \sigma(f_s([\mathbf{h}_v\mathbf{W}_k, \mathbf{q_c}(i-1)\mathbf{W}_q])), \tag{3.10}$$

where $\mathbf{W}_k, \mathbf{W}_q, \in \mathbb{R}^{d\_hidden \times d\_hidden}$ are projection parameters for query and the keys, f_s is a non-linear network and $[\cdot]$ is used to denote the concatenation. Then we create two masks; one mask $\mathbf{m}_v^c(i)$ to filter out the factors of patch v which are similar to current common factors $\mathbf{q_c}(i-1)$ and the other $\mathbf{m}_v^l(i)$ to filter out factors which are considered local/non-common of current patch.

$$\mathbf{m}_v^c(i) = \mathbb{1}[\![\sigma(\mathbf{h}_v\mathbf{W}_k) \geq \delta_v(i)]\!], \tag{3.11}$$

$$\mathbf{m}_v^l(i) = \mathbb{1}[\![\sigma(\mathbf{h}_v\mathbf{W}_k) < \delta_v(i)]\!], \tag{3.12}$$

$$\mathbf{h}_v^c(i) = \mathbf{m}_v^c(i) \odot \mathbf{h}_v\mathbf{W}_v, \tag{3.13}$$

$$\mathbf{h}_v^l(i) = \mathbf{m}_v^l(i) \odot \mathbf{h}_v\mathbf{W}_v, \tag{3.14}$$

where $\mathbf{W}_v \in \mathbb{R}^{d\_hidden \times d\_hidden}$ is projection parameters and $\mathbf{m}_v^c(i), \mathbf{m}_v^l(i) \in \mathbb{R}^{d\_hidden}$. $\odot$ denotes element-wise multiplication used for masking. Now we use the filtered out common latent from each patch v and accumulate them to a single representation and use that to update our query $\mathbf{q_c}(i-1)$ for the current iteration i using a Gated Recurrent Unit(GRU) as follows:

$$\mathbf{q}_{update}(i) = \sum_{v \in V} \mathbf{h}_v^c(i), \tag{3.15}$$

$$\mathbf{q}_c(i) = \text{GRU}(\mathbf{q}_{update}(i), \mathbf{q}_c(i-1)) \tag{3.16}$$

This accumulation approach is depicted in Fig. 3.2.

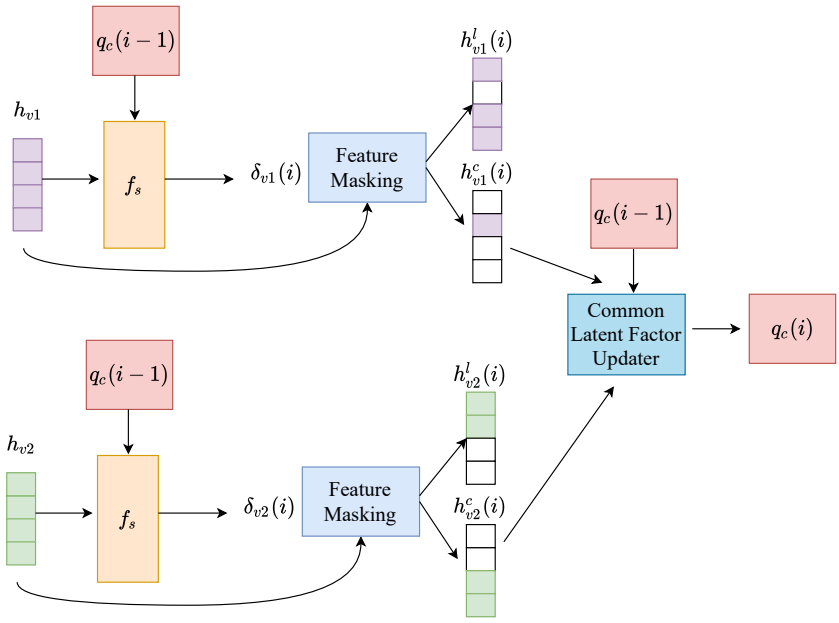


FIGURE 3.2: ACCUM : Our main algorithmic invention to enable extraction of high-quality common latent factor for deepGCFX based on query based reasoning and feature masking. For this example of two node graph, at the iteration i, the single common latent factor representation from previous iteration acts as the query $\mathbf{q}_c(i-1)$. For each node, $\delta(i)$ is calculated (Eq. 3.10) to determine the factor wise similarity each node has with current common latent factor. Then each node's latent factors are divided into common and local using a mask based on $\delta(i)$ (Eq. 3.11-3.14) and common factors from each node are accumulated together using Eq. 3.16 to update graph-wise common latent factor for current iteration.

Posterior distribution parameter generation. Once the iteration process is over after m iterations in our proposed ACCUM, we use $\mathbf{q_c}(m)$ as the final single common latent factor vector $\mathbf{h}_c$ for current graph and we continue the procedure from Eq. 10-14 of main paper to obtain final filtered factors for each patch v $(\mathbf{h}_v^c(m+1), \mathbf{h}_v^l(m+1))$. Now we use $\mathbf{h}_c$ and $\mathbf{h}_v^l(m+1)$ to generate parameters for our posterior distributions. We use fully connected layers f for this. $\boldsymbol{\mu}_\mathcal{C} = f_{\boldsymbol{\mu}_\mathcal{C}}(\mathbf{h}_c)$ and $log\ \boldsymbol{\sigma}_\mathcal{C} = f_{\boldsymbol{\sigma}_\mathcal{C}}(\mathbf{h}_c)$ are used to generate parameters for the single Gaussian posterior distribution for graph-wise common latent factors $\mathbf{h}_c$; $q_\phi(\mathbf{z}_c|G) = \mathcal{N}(\boldsymbol{\mu}_c, diag(\boldsymbol{\sigma}_c))$. Next, we obtain parameters for local latent posterior distributions; $q_\phi(\mathbf{z}_l(j)|G) = \mathcal{N}(\boldsymbol{\mu}_l(j), diag(\boldsymbol{\sigma}_l(j)),\ \forall j \in \{1 \dots |V|\}$, where $\boldsymbol{\mu}_l = f_{\boldsymbol{\mu}_l}(\mathbf{h}_v^l(m+1))$ and $\log\ \boldsymbol{\sigma}_l = f_{\boldsymbol{\sigma}_l}(\mathbf{h}_v^l(m+1))$.

3.4.3 Commonality and Relevance preserving Decoder

Aggregation decoder $\mathcal{D}_{agg}$ to enforce commonality of $\mathbf{z}_c$ for all patches v. In order to reconstruct the original graph properly, the model requires both common and local latents. Therefore, common and local latent factors are sampled from their respective posterior distributions ($\mathbf{z}_c \sim q_\phi(\mathbf{z}_c|G)$ and $\mathbf{z}_l(j) \sim q_\phi(\mathbf{z}_l(j)|G),\ \forall j \in \{1 \dots |V|\}$) and sent through the decoder $\mathcal{D}_{agg}$ for reconstructing the graph G. Note that the graph-wise common latent factor $\mathbf{z}_c$ is only sampled once for the entire graph using $q_\phi(\mathbf{z}_c|G)$. While the baseline GVAE simply obtain adjacency reconstruction via $p_\theta(A_{jk} = 1|\mathbf{z}_j, \mathbf{z}_k) = \mathbf{z}_j^T \mathbf{z}_k,\ \forall j,k \in \{1 \dots |V|\}$ by sampling latent, GCFX has two types of latent. Therefore to enhance the graph decoder capability to fuse both latent properly, we use a non-linear feed-forward network as our aggregated decoder $\mathcal{D}_{agg}$. The latent aggregated reconstruction of G can be achieved via adjacency matrix reconstruction A as follows:

$$\mathbb{E}_{q_\phi(\mathbf{z}_c,\mathbf{Z}_l|G)}[\log p_\theta(G|\mathbf{z}_c, \mathbf{Z}_l)] = \mathbb{E}_{q_\phi(\mathbf{z}_c,\mathbf{Z}_l|G)} \sum_{j \in V} \sum_{k \in V} \log p_\theta(A_{jk}|\mathbf{z}_c, \mathbf{z}_l(j), \mathbf{z}_l(k)), \tag{3.17}$$

where $p_\theta(A_{jk} = 1|\mathbf{z}_c, \mathbf{z}_l(j), \mathbf{z}_l(k)) = \mathcal{D}_{agg}([\mathbf{z}_c, \mathbf{z}_l(j)])^T\, \mathcal{D}_{agg}([\mathbf{z}_c, \mathbf{z}_l(k)]),\ \forall j,k \in \{1 \dots |V|\}$. $[\cdot]$ indicates concatenation operation.

Although $\mathcal{D}_{agg}$ enforces that $\mathbf{z}_c$ should contain factors common to all patches in order to enable proper graph reconstruction, $\mathcal{D}_{agg}$ cannot enforce what type of common factors $\mathbf{z}_c$ should posses. Since $\mathbf{z}_c$ is a constant for all patches, if the type of information that $\mathbf{z}_c$ should have was not enforced, there is a possibility that $\mathbf{z}_c$ gets ignored by $\mathcal{D}_{agg}$. To mitigate this, deepGCFX employ a regularization decoder $\mathcal{D}_{reg}$.

Regularization decoder $\mathcal{D}_{reg}$ to enforce relevance of $\mathbf{z}_c$ to current graph. This decoder is aimed at enforcing $\mathbf{z}_c$ that it should contain structural information about the graph G. To achieve that, we require patch specific common latent factors instead of the single common talent factor for the entire graph. Eq. 3.13 provides this. This decoder is only employed during training and we obtain parameters for patch specific common latent posterior distributions by; $q_\phi(\mathbf{z}_c(j)|G) = \mathcal{N}(\boldsymbol{\mu}_c(j), diag(\boldsymbol{\sigma}_c(j)),\ \forall j \in \{1 \dots |V|\}$, where $\boldsymbol{\mu}_c = f_{\boldsymbol{\mu}_c}(\mathbf{h}_v^c(n+1))$ and $log\ \boldsymbol{\sigma}_c = f_{\boldsymbol{\sigma}_c}(\mathbf{h}_v^c(n+1))$. Then the sampled patch specific common latent factor representations $\mathbf{z}_c(j) \sim q_\phi(\mathbf{z}_c(j)|G),\ \forall j \in \{1 \dots |V|\}$) are used to reconstruct the adjacency matrix as

$$\mathbb{E}_{q_\phi(\mathbf{Z}_c,\mathbf{Z}_l|G)}[\log p_\theta(G|\mathbf{Z}_c, \mathbf{Z}_l)] = \mathbb{E}_{q_\phi(\mathbf{Z}_c,\mathbf{Z}_l|G)} \sum_{j \in V} \sum_{k \in V} \log p_\theta(A_{jk}|\mathbf{z}_c(j), \mathbf{z}_c(k)), \tag{3.18}$$

where $p_\theta(A_{jk} = 1|\mathbf{z}_c(j), \mathbf{z}_c(k)) = \mathbf{z}_c(j)^T \mathbf{z}_c(k),\ \forall j,k \in \{1 \dots |V|\}$. The impact $\mathbf{z}_c$ gets from both $\mathcal{D}_{agg}$ and $\mathcal{D}_{reg}$ to maintain both commonality and relevancy to G enforces our proposed iterative accumulation method ACCUM to extract latent factors from patches that fulfil both commonality and relevance to current graph.

3.4.4 deepGCFX Training

We modify the objective function in Eq. 3.6 obtained in general GCFX framework for common latent factor extraction to following $\mathcal{L}_{deepGCFX}$ for training deepGCFX in an end-to-end manner.

$$\begin{aligned}\mathcal{L}_{deepGCFX} &= \mathcal{L}_{\mathcal{D}_{agg}} + \beta\mathcal{L}_{c\_prior} + \gamma\mathcal{L}_{l\_prior} + \mathcal{L}_{\mathcal{D}_{reg}} \\ &= \mathbb{E}_{q_\phi(\mathbf{z}_c,\mathbf{Z}_l|G)}[\log p_\theta(G|\mathbf{z}_c,\mathbf{Z}_c)] \\ &- \beta\ KL(q_\phi(\mathbf{z}_c|G)\parallel p(\mathbf{z}_c)) \\ &- \gamma \sum_{j=1}^{|V|} KL(q_\phi(\mathbf{z}_l(j)|G)\parallel p(\mathbf{z}_l(j))) \\ &+ \mathbb{E}_{q_\phi(\mathbf{z}_c,\mathbf{Z}_l|G)}[\log p_\theta(G|\mathbf{Z}_c)]\end{aligned} \tag{3.19}$$

3.4.5 deepGCFX Inference

After training deepGCFX, we utilize $\mathbf{z}_c \sim q_\phi(\mathbf{z}_c|G)$ as learnt single common latent factor representation for graph G and $\mathbf{z}_l(j) \sim q_\phi(\mathbf{z}_l(j)|G),\ \forall j \in \{1\dots|V|\}$ as local non-common patch specific latent factor representations for downstream tasks.

3.5 Novelty comparison with GVAE

Main objective of GVAE (Thomas N. Kipf and Welling, 2016b) architecture is to reconstruct the graph based on proximity. This heavy emphasis on proximity, makes it unable to differentiate the importance of different latent factors. GVAE treats all latent factors contribute to reconstruction equally. To follow GCFX principle, it is essential to differentiate graph wise common factors from local factors. Therefore GVAE is unable to solve GCFX. Our proposed deepGCFX on the other hand is equipped with ACCUM (Section 3.4.2), a query based reasoning approach with feature masking to explicitly extract common factors. Also compared to GVAE's proximity overemphasized decoder, we also proposed specilized aggregation decoder and regularization decoder (Section 3.4.3) to ensure that extracted factors by ACCUM maintains commonality and relevancy to the input graph. In the evaluation section we analyze the capability of these two novel components in extracting graph-wise common latent factors and their impact on downstream tasks against GVAE.

3.6 Experimental Setup

3.6.1 Datasets

Graph level datasets. We select six commonly used graph classification benchmark datasets as follows: MUTAG (Kriege and Mutzel, 2012) dataset contains mutagenic aromatic and heteroaromatic nitro compounds while PTC dataset (Kriege and Mutzel, 2012) consists of chemical compounds reported for carcinogenicity of rats. Apart from these bioinformatics datasets, next we evaluate on four social network datasets (Yanardag and Vishwanathan, 2015) namely IMDB-BINARY, IMDB-MULTI, REDDIT-BINARY

and REDDIT-MULTI-5K. IMDB datasets contain information about movies where the nodes are actors/actresses and they are connected by edges if they have acted in the same movie. IMDB-BINARY contains two genres of movies: *Action* and *Romance*. Multi-class version of IMDB dataset contains movies from *Comedy*, *Romance* and *Sci-Fi* genres. Reddit datasets were created using threads in different subreddits. Nodes in each graph are users who responded to that particular thread and edges are formed when one user responds to another user's comment. REDDIT-BINARY dataset labels each graph as question/answer-based community or a discussion-based community and REDDIT-MULTI-5K labels graphs into 5 labels according to their subreddit namely, *worldnews*, *videos*, *AdviceAnimals*, *aww* and *mildlyinteresting*. Table 3.2 contains statistics of all these datasets.

TABLE 3.2: Statistics of datasets used for graph level tasks

DATASET	MUTAG	PTC-MR	IMDB-BIN	IMDB-MUL	RED-BIN	RED-MUL-5K
# Graphs	188	344	1000	1500	2000	4999
Avg. Nodes	17.93	14.29	19.77	13.0	429.63	508.52
Avg. Edges	19.79	14.69	96.53	65.94	497.75	594.87
#Features	7	18	0	0	0	0
# Classes	2	2	2	3	2	5

For MUTAG and PTC-MR, node labels are used as node features and for all other datasets where node features are not available, we used a single constant node feature with value 1 following Infograph (F.-Y. Sun et al., 2020).

Node level datasets. We use the same node classification benchmark as Geom-GCN by Pei et al. Pei et al. (2020). CORA, CITESEER and PUBMED (Z. Yang, Cohen, and Salakhutdinov, 2016) citation networks are used as assortative (nearby neighbour nodes of a graph belong to same class label) while CHAMELEON and SQUIRREL topics from Wikipedia Network (Rozemberczki, Allen, and Sarkar, 2021) and ACTOR co-occurrence network (Tang et al., 2009) are used as disassortative (nearby neighbour nodes of a graph do not belong to same class label, nodes further away from each other belong to same class) graphs. Geom-GCN (Pei et al., 2020) introduced a metric to measure the amount of homophily of a graph as Eq. 3.20. Table 3.3 contains statistics of all these datasets.

$$H(\mathcal{G}) = \frac{1}{|V|} \sum_{v \in V} \frac{\text{No of } v\text{'s directly connected nodes with same label}}{\text{No of } v\text{'s directly connected nodes}}. \tag{3.20}$$

Synthetic datasets. Apart from above mentioned real world datasets, we employed two synthetic datasets based on Erdos-Renyi (Erdos and Renyi, 1960) and Random Geometric (Dıaz et al., 1999) graph generation models. These two datasets were specifically used to evaluate the effectiveness of deepGCFX itself rather than the effectiveness of learnt latent representations on downstream tasks. In Section 3.7.3 we provide

TABLE 3.3: Statistics of the datasets used for node level tasks. $H(\mathcal{G})$ can be used to distinguish assortative and disassortative graph datasets.

	ASSORTATIVE			DISASSORTATIVE		
DATASETS	**CORA**	**CITESEER**	**PUBMED**	**CHAMELEON**	**SQUIRREL**	**ACTOR**
$H(\mathcal{G})$	0.83	0.71	0.79	0.25	0.22	0.24
#Nodes	2708	3327	19717	2277	5201	7600
#Edges	5429	4732	44338	36101	217073	33544
#Features	1433	3703	500	2325	2089	931
#Classes	7	6	3	5	5	5

complete details on how we designed the graph generation process along with its relatedness to our evaluation criteria.

3.6.2 Selected Baselines

There are three main baselines we selected for our experiments along with state-of-the-art models as follows:

- **Graph Variational Autoencoder (GVAE)** (Thomas N. Kipf and Welling, 2016b) - For both graph and node levels. The reason for us to propose deepGCFX is the inability of existing graph autoencoders to model the GCFX principle we proposed. While deepGCFX can be applied to any existing variations of graph autoencoder (Thomas N. Kipf and Welling, 2016b; Pan et al., 2018; Park et al., 2019), we selected GVAE as the baseline of deepGCFX to directly align with the GCFX's main objective indicated in Eq. 3.6. GVAE cannot extract common latent factors as it cannot differentiate features. This happens because GVAE completely relies on proximity in graph reconstruction loss treating each latent equal. To address these weaknesses, we propose deepGCFX by proposing a novel model along with an improved objective function in Section 3.4. Hence for all our experiments, we use GVAE as our baseline to highlight the advantages deepGCFX have over GVAE.

- **Infograph** (F.-Y. Sun et al., 2020) - For graph level. Contrastive learning is one of the leading methods for unsupervised and self-supervised graph embedding learning. While more recent and better performing contrastive learning methods use extra pretraining datasets (Yanardag and Vishwanathan, 2015), multiple-views (Hassani and Khasahmadi, 2020), data augmentation (You, T. Chen, Sui, et al., 2020), Infograph is the most basic and the first model of this kind. deepGCFX also does not use any additional information same as Infograph. Hence when it comes to compare the effect of different principles of Infomax and GCFX, we select Infograph as our fair baseline for graph level tasks.

- **Deep Graph Infomax (DGI)** (Velickovic, Fedus, et al., 2019)- For node level. DGI is the first deep contrastive learning method based on Infomax principle proposed for graph representation learning. This was specifically aimed at node level tasks. F.-Y. Sun et al. (2020) proposed Infograph following the approach of DGI as well. Velickovic, Fedus, et al. (2019) claim DGI has the potential to handle long-distance node dependencies due to the specific constrastive learning

method DGI has employed. This is also a basic method without using any additional information such as extra data or augmentations. Hence we use this as our contrastive learning baseline to compare the impact of Infomax and GCFX principles on node level tasks.

While we compare the performance of deepGCFX against state of the art models with explicit inter-graph similarity (Ex: kernel methods (D. Chen, Jacob, and Mairal, 2020), Graph edit distance based proximity minimizing methods (Bai et al., 2019)), we do not consider explicit inter-graph similarity methods as a fair competitive baseline to GCFX principle. This is because while GCFX principle solely aims at single sample based representation learning, inter-graph similarity solely focuses on inter-graph similarity based representation learning. Hence principle wise, GCFX and inter-graph similarity are fundamentally different. Since fundamental idea of Infomax principle is to maximize mutual information between input and output, it also mainly focuses on single sample based representation learning (although the proposed methods deviate from this fundamental idea in order to achieve better performance). Hence we selected Infomax principle as our fair state-of-the-art baseline.

3.6.3 Experiment Details

Main Implementation Details. For both graph and node level tasks, we train deepGCFX using $\mathcal{L}_{deepGCFX}$ (Eq. 3.19) where number of GNN encoder layers, hidden dimension size of layers, β and γ were selected based on lowest graph reconstruction loss. For graph level tasks we trained deepGCFX models for all datasets maximum 500 epochs and for node level tasks it is 2000 epochs. We used early stopping based on the lowest combined loss of $\mathcal{L}_{\mathcal{D}_{agg}} + \mathcal{L}_{\mathcal{D}_{reg}}$. We used Graph Isomorphism Network (GIN) (K. Xu et al., 2019) as the GNN for graph level tasks following existing work and Graph Convolutional Network (GCN) (Thomas N. Kipf and Welling, 2017) for node level tasks for a fair comparison. We randomly initialize $\mathbf{q_c}(0)$(initial graph-wise common accumulated latent) to start the ACCUM process (Sec. 3.4.2). Validation split performance is used to determine the number of iteration steps of accumulation required for the downstream task. Following existing evaluation protocol, we use SVM (Chang and C.-J. Lin, 2011) as the downstream task classifier for graph level and a linear classifier for node level. Validation split performance is also used to set regularization parameter C for SVM and early stop training for linear classifier.

Handling randomness. Following CMV (Hassani and Khasahmadi, 2020), we set the random seed to 123 or its permutations for all our deepGCFX models and select the best seed based on validation split performance.

Utilized hardware and software infrastructure. As the GPU, we used one GeForce RTX 2080 Ti with 11GB memory. All implementations were done using Pytorch (Paszke, Gross, Massa, et al., 2019) and Pytorch Geometric (Fey and Lenssen, 2019).

Used evaluation criteria and metrics. For both graph level and node level comparisons with existing models we strictly followed the evaluation metrics used by existing

work for fair comparison. For both graph and node classification, mean accuracy and standard deviation is used as the metric. For graph classification, mean of 5 different runs using 10-fold cross validation and for node classification the mean of 10 different train-validation-test splits provided by Geom-GCN (Pei et al., 2020) is used. For the model analysis in Section 3.7.2, we selected Spearman's correlation and Mean Absolute Pair-wise Difference (MAPD) as evaluation metrics. We adopted these metrics from disentanglement learning domain. Although our model is not aimed at disentanglement of latent factors according to variations of input, we are also doing a filtering in latent level (but without explicit latent to input mapping). Therefore evaluation criteria which measures the correlation among latent variables are suitable for deepGCFX as well. While Spearman R evaluates the correlation between latent variables, MAPD (Higgins et al., 2017) measures feature similarity of extracted latent factors.

Hyper-parameter selection. Table 3.4 shows the final set of hyper parameters selected for graph level datasets. We used mini batch size of 128 and learning rate of 0.001 with Adam optimizer for all datasets. Hidden dimensions, number of GNN layers, β, γ were selected from $\{32, 128, 512\}$, $\{2, 3, 4\}$, $\{0.01, 0.1, 1, 10\}$, $\{0.01, 0.1, 1, 10\}$ based on the lowest $\mathcal{L}_{\mathcal{D}_{agg}}$. Regularization parameter C for SVM, number of ACCUM iterations, gate value α (which determines the contribution of common and non common latent representations for the final embedding) for deepGCFX++ were selected from $\{1, 10, 100, 1000\}$, $\{0, 1, 2, 3, 4\}$, $\{0, 0.05, 0.1, 0.15, \ldots, 1.0\}$ respectively based on the validation split performance. The validation split was selected from extracting one fold from 9 training folds at each run on the 10 fold cross validation. We used the evaluation script from GraphCL (You, T. Chen, Sui, et al., 2020) for this. For baseline GVAE also we used same number of hidden encoder GNN layers, hidden dimension but γ set to 1 as it gave lowest reconstruction loss in this case. Minimum reconstruction loss was used for early stopping of GVAE.

TABLE 3.4: Selected final hyper-parameters for graph level tasks

DATASET	MUTAG	PTC-MR	IMDB-BIN	IMDB-MUL	RED-BIN	RED-MUL-5K
# GNN layers	4	3	3	3	3	3
Hidden size	32	32	128	128	128	128
# ACCUM i	2	2	3	3	3	3
β	0.1	0.1	0.1	0.1	0.1	0.1
γ	1	1	1	1	1	1
C of SVM	10	10	10	10	10	10
α	0.7	0.85	0.95	0.85	0.9	0.85

For node level tasks, all three unsupervised models (DGI, GVAE and deepGCFX) used 1 GCN layer as the GNN encoder with 512 hidden size and 0.001 learning rate. All models used early stopping in training where DGI used lowest contrastive loss and GVAE and deepGCFX used lowest reconstruction loss. β is set to 0.1 and γ is 1 for deepGCFX model and by dafault we used 2 ACCUM iterations. A linear layer with hidden size 512 is used as the classifier with 0.01 learning rate. Validation accuracy was

used for early stopping of linear classifier for all three models. Average accuracy of fifty randomly initialized models, each trained for maximum 100 epochs is used as the final accuracy for each of 10 Geom-GCN splits. α for deepGCFX++ was also selected from $\{0, 0.05, 0.1, 0.15, \ldots, 1.0\}$ using validation split.

3.7 Experimental Results and Analysis

We evaluate our proposed GCFX principle and novel deepGCFX model for their capabilities of extracting graph-wise common factors and its performance and advantages over existing unsupervised graph representation learning methods by answering following questions:

Q1 Can extracted latent factors $\mathbf{z}_c$ actually capture graph-wise common information? - Section 3.7.2

Q2 Is there a correspondence between learnt graph-wise common latent factors $\mathbf{z}_c$ and original common graph generative factors (C_f)? - Section 3.7.3

Q3 What is the contribution of graph-wise common latent factors $\mathbf{z}_c$ to obtain a graph embedding with discriminative abilities which can be utilized as graph level representation, when evaluated on downstream tasks? - Section 3.7.4

Q4 What is the impact of graph-wise common latent factors $\mathbf{z}_c$ on node level tasks? - Section 3.7.5

Q5 Is there any impact of the number of ACCUM iterations on downstream task performance? - Section 3.7.6

3.7.1 Main findings of the evaluation

Before diving into detailed discussions about each of our experiments and their results, this section highlights main answers and findings we obtained for our questions.

A1 Proposed iterative ACCUM approach enables feature differentiation of deepGCFX by increasing the mutual exclusion of features captured in common and local latent factors with the increase of the iteration steps. Also with the increase of iterations, the commonality of $\mathbf{z}_c$ across the graph increases while maintaining the relevance to input. This demonstrates the effectiveness of GCFX principle and proposed deepGCFX architecture for graph-wise common latent factor extraction.

A2 When ground truth common graph generative factors C_f are available, deepGCFX becomes an interpretable model as extracted common latent factors $\mathbf{z}_c$ demonstrate a clear correspondence with C_f. Additionally, these extracted latent factors, both common ($\mathbf{z}_c$) and local ($\mathbf{Z}_l$) are transferable. Pre-trained deepGCFX can successfully generate novel graphs with transferred generative factors.

A3 Graph-wise common latent factors $\mathbf{z}_c$ is a very effective representation to obtain a graph embedding with discriminative abilities which can be utilized as graph embedding beneficial for downstream graph level tasks. We achieve state-of-the-art performance surpassing existing contrastive learning models without using any additional features and very competitive results to pair-wise inter-graph similarity based models with higher efficiency and better scalability.

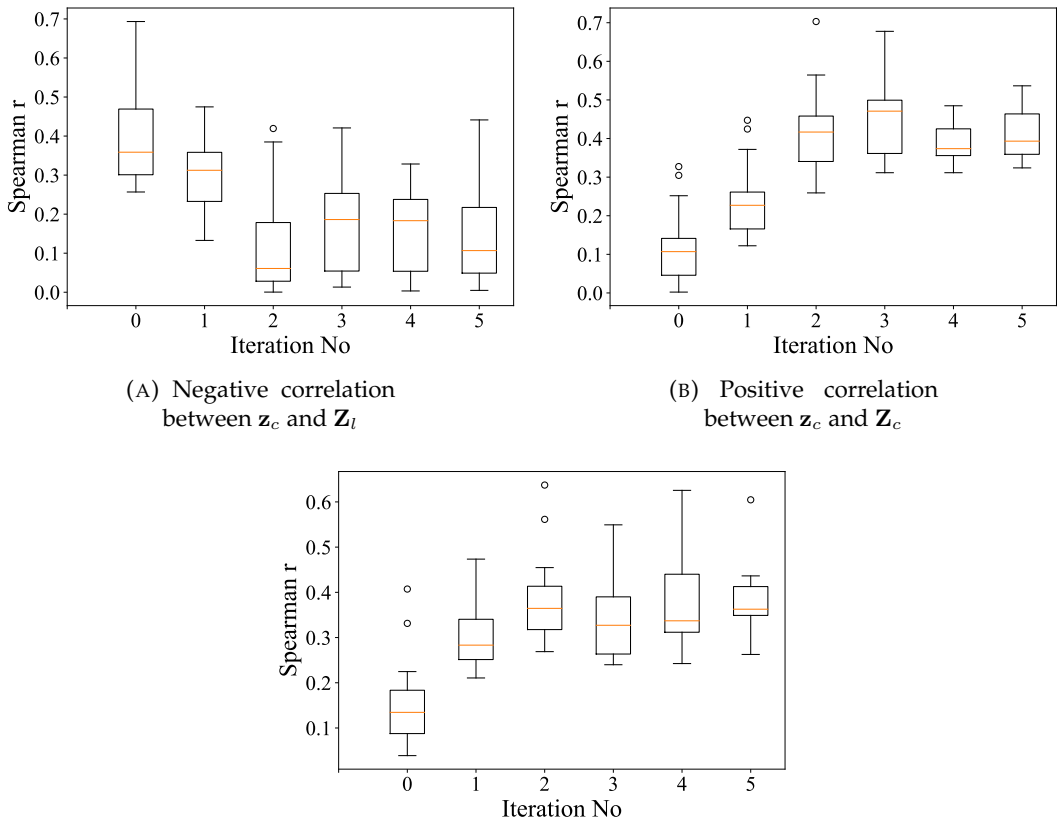
A4 deepGCFX's ability in extracting graph-wise common latent factors is beneficial for node level tasks in three ways. First is, Graph-wise common latent factors $\mathbf{z}_c$ is capable of incorporating long-range node dependencies to node embeddings which improves node classification performance. Second is, separating out common features from non-common enables deepGCFX to preserve node specific local features in the latent factors maintaining original characteristics of nodes without hindering them by overemphasizing homophily assumption like existing work. Third is, extracting common factors from local provides flexibility in determining the contribution required from graph-wise common latent factors to obtain optimal node level task performance. While these capabilities are beneficial for assortative, they are specifically advantageous for disassortative graphs compared to existing work.

A5 With the increase of ACCUM iteration steps, the commonality and input relatedness of extracted $\mathbf{z}_c$ increases. As discussed in A3, graph-wise common factors are crucial to capture graph level relevant factors. Hence the downstream task performance and the contribution of $\mathbf{z}_c$ to obtain the optimal results also increases.

3.7.2 Effectiveness analysis of deepGCFX for graph-wise common latent factor extraction

To evaluate the effectiveness of deepGCFX in extracting graph-wise common latent factors, we consider the capability of deepGCFX in three parts. (i) First is to verify whether proposed components of deepGCFX is capable of feature differentiation and elimination of overemphasis on proximity as it claims. (i) Second is to compare the feature differentiation ability of deepGCFX with baseline GVAE to further verify the superiority of proposed deepGCFX over existing baseline. (iii) Third and final is to evaluate whether the extracted common latent factors obtained by filtering out non-common local factors are indeed common across the graph in order to confirm the effectiveness of deepGCFX and GCFX principle in extracting graph-wise common latent factors. Our proposed ACCUM method extracts a single latent representation $\mathbf{z}_c$ for an entire graph which we utilize for downstream tasks. However only for the analysis of this section, we directly use patch specific common latent factors $\mathbf{Z}_c$ (which were aggregated to obtain $\mathbf{z}_c$) as well, in order to clearly discuss the capabilities of deepGCFX.

To analyze part (i), we measure how the correlation of extracted common latent $\mathbf{z}_c$ varies with local latent factors $\mathbf{Z}_l$, patch specific common latent factors $\mathbf{Z}_c$ and the output from the aggregation decoder $\mathcal{D}_{agg}$. We use Spearman's correlation for this following, graph property recovery methods such as UDR (Duan et al., 2020). Fig. 3.3 shows how each of these correlations change with the increase of the number of iterations for a sample graph from MUTAG dataset. We start with the accumulation



(A) Negative correlation between $\mathbf{z}_c$ and $\mathbf{Z}_l$

(B) Positive correlation between $\mathbf{z}_c$ and $\mathbf{Z}_c$

(C) Positive correlation between $\mathbf{z}_c$ and output from $\mathcal{D}_{agg}$

FIGURE 3.3: Analyzing feature differentiation and elimination of proximity overemphasis ability of deepGCFX: Figures show variation of Spearman R absolute correlation of extracted common latent factors $\mathbf{z}_c$ with local $\mathbf{Z}_l$, patch specific common $\mathbf{Z}_c$ and output from $\mathcal{D}_{agg}$ with the increase of number accumulation iterations. With the increment of iterations (A) shows negative correlation between common and non-common latent factors highlighting that deepGCFX learn to differentiate which features are common and filtering out rest of the features as non-common, (B) shows that $\mathbf{z}_c$ learns to extract more and more information from all patches of the graph and (C) highlights how the requirement of $\mathbf{z}_c$ increases for $\mathcal{D}_{agg}$ in reconstructing the graph showing both commonality of extracted $\mathbf{z}_c$ across the entire graph as well as its relevance to input.

iteration 0 where the common and local latent filtering done at random to show how well the proposed ACCUM performs against random filtering.

We can observe that, with the increase of ACCUM iterations, correlation between local factors $\mathbf{Z}_l$ and common factors $\mathbf{z}_c$ decreases (Fig 3.3(a)) showing our proposed ACCUM method's effectiveness in feature differentiation by being able to filter out different types of information as common and local latent factors respectively. Fig 3.3(b)

shows the positive relationship aggregated common latent factors $\mathbf{z}_c$ have with its patch specific counterparts $\mathbf{Z}_c$. This shows that although ACCUM process was started with $\mathbf{z}_c$ initialized randomly, the iterative ACCUM has indeed updated $\mathbf{z}_c$ with graph relevant features extracted from all the patches from the input graph. Finally, in Fig 3.3(c) the correlation of $\mathbf{z}_c$ with the output of $\mathcal{D}_{agg}$ increases showing that $\mathcal{D}_{agg}$ incorporates common global graph level factors for graph reconstruction without overemphasizing and completely relying on local proximity. Therefore we can confirm that $\mathbf{z}_c$ does not get ignored during the decoding process. Since $\mathbf{z}_c$ is a non-ignored common input for all nodes in $\mathcal{D}_{agg}$, its positive correlation with reconstructed output from $\mathcal{D}_{agg}$ also ensures that $\mathbf{z}_c$ indeed captures latent information which are common for all the nodes and edges of the graph and those common information are indeed related to the current input graph, not some random noise.

After two ACCUM iterations we observe that the correlations have stabilized without further increasing or decreasing significantly. This is due to the impact of the regularization provided by the proposed dual decoder based decoding process of deepGCFX. By maintaining commonality across entire graph, there is a upper limit of graph reconstruction ability $\mathbf{z}_c$ can achieve in $\mathcal{D}_{reg}$. It cannot fully reconstruct original graph like $\mathcal{D}_{agg}$ due to lack of local factors. Until certain number of ACCUM iteration steps, $\mathbf{z}_c$ extracts as much as graph-wise common latent factors which are also crucial for graph reconstruction. After that, deepGCFX is unable to extract common factors while maintaining those factors' relevance to graph reconstruction. Therefore, increasing more ACCUM iterations after this point yields no added benefit in improving deepGCFX's ability of graph-wise common latent factor extraction.

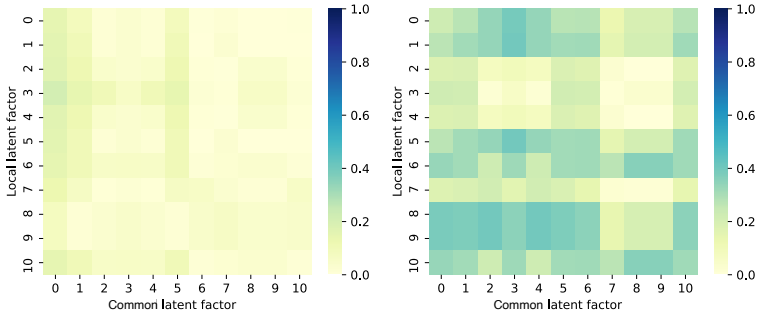


FIGURE 3.4: Feature differentiation ability of deepGCFX(left) against baseline GVAE(right): Absolute correlation between extracted common global and non-common local latent factors by deepGCFX is very low compared to global-local (obtained by dividing latent to halves) correlation of GVAE, which does not have feature differentiation capability.

After confirming that proposed deepGCFX can indeed differentiate common and non-common latent factors from an input graph, we move to part (ii) to compare it against baseline GVAE which does not have feature differentiation capability. We measure the similarity and dependence between common $\mathbf{z}_c$ and local $\mathbf{Z}_l$ latent factors for a given graph G. Here we get inspiration from graph disentangled learning (J. Ma et al., 2019) for our evaluation. We calculate correlation between $\mathbf{z}_c$ and $\mathbf{Z}_l$ for graphs from MUTAG (Kriege and Mutzel, 2012) dataset and visualize in Fig. 3.4. Following UDR$_s$

(Duan et al., 2020), we use Spearman's correlation to calculate the similarity/ correlation matrix. We use our best selected converged deepGCFX model for this. As the reference to demonstrate the superiority, we use the output correlation for the same graph from our baseline GVAE (we divide its single latent representation to halves and consider as $\mathbf{Z}_c$ and $\mathbf{Z}_l$ each half). Entry (i, j) of the correlation matrix indicates the absolute correlation value between $\mathbf{z}_c(i)$ and $\mathbf{z}_l(j)$, $\forall i, j \in \{1 \dots |V|\}$.

The diagonals of the two correlation matrices in Fig. 3.4 shows the correlation between $\mathbf{Z}_c$ and $\mathbf{Z}_l$ extracted by deepGCFX(left) and GVAE(right) for each patch of the graph. We can observe that the correlation values in diagonal is very low, closed to 0.0 for deepGCFX, while correlations between $\mathbf{Z}_c$ and $\mathbf{Z}_l$ latent factors of GVAE have higher values. This confirms the superiority of proposed deepGCFX in feature differentiation against the baseline GVAE which has no ability to differentiate latent factors, hence the global and local latent division was random and all latent were treated equally by GVAE.

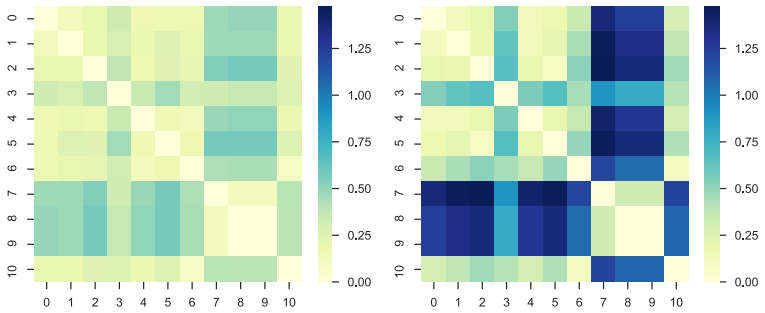


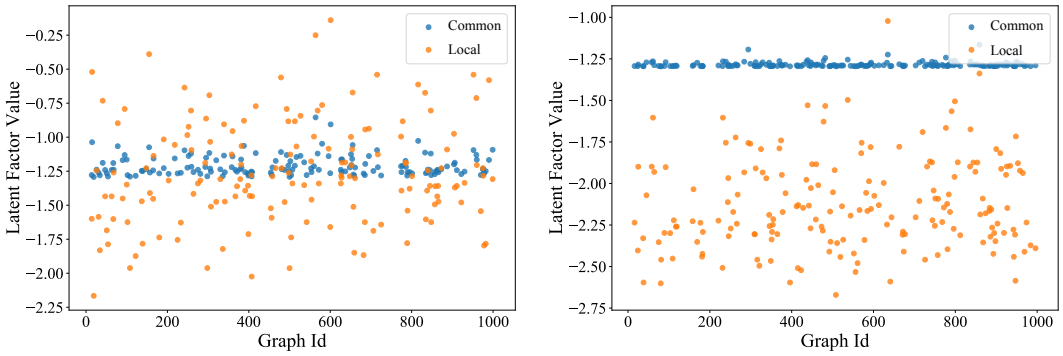
FIGURE 3.5: Inter-patch MAPD among Common(left) and local(right) latent factors. Lower MAPD for $\mathbf{Z}_c$ indicates the latent factors extracted in $\mathbf{Z}_c$ (which are aggregated to obtain $\mathbf{z}_c$) by deepGCFX is indeed shared among all patches of the entire graph; hence common, unlike $\mathbf{Z}_l$ which are specific to certain patches (therefore higher MAPD).

Finally, in part (iii) to further verify whether the extracted latent $\mathbf{z}_c$ indeed captures graph-wise common latent factors from all the patches of the graph filtering non-common local factors, we conduct another analysis. For this, we need to find the inter-patch similarity of all learnt $\mathbf{z}_c(i)$, $\forall i \in \{1 \dots |V|\}$ against inter-patch similarity of $\mathbf{z}_l(i)$, $\forall i \in \{1 \dots |V|\}$. Since $\mathbf{z}_c(i)$ for all the patches of the same graph are expected to be same, we expect $\mathbf{z}_c(i)$ to only have very small variance among them compared to $\mathbf{z}_l(i)$ which are locally specific to each patch. Hence, we check if $\mathbf{z}_c(i)$ extracted from each patch should be similar to each-other than their local latent factors $\mathbf{z}_l(i)$. Since the information propagation is different from patch to patch due to neighbourhood differences, we cannot assume that the extracted $\mathbf{z}_l(i)$ to be exactly the same for all patches. But they should be more similar than extracted $\mathbf{z}_l(i)$. We evaluate this using the Mean Absolute Pairwise Difference (MAPD) measure used by Higgins et al. (2017). MAPD matrix for $\mathbf{Z}_c$ is calculated as $\text{MAPD}_c(i, j) = \text{MEAN}(\left|\mathbf{z}_c(i) - \mathbf{z}_c(j)\right|)$ and $\text{MAPD}_l(i, j) = \text{MEAN}(\left|\mathbf{z}_l(i) - \mathbf{z}_l(j)\right|)$ used for $\mathbf{Z}_l$. We average over all the dimensions of the latent.

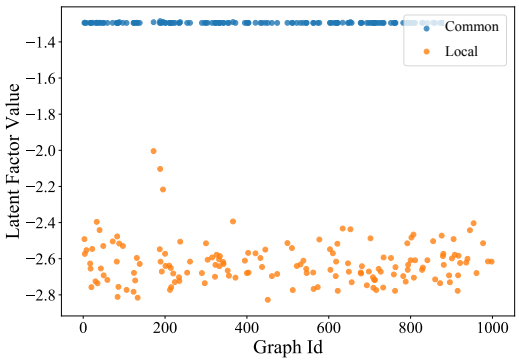
In Fig. 3.5, we can observe that inter-patch common latent factor similarity (left) is very high compared to inter-patch local factor similarity(right) demonstrating that the latent factors extracted by our ACCUM method indeed common across the graph. Hence we can confirm that our proposed deepGCFX based on novel GCFX principle indeed is capable of graph-wise common latent factor extraction.

3.7.3 Correspondence between ground truth common graph generative factors (C_f) and extracted common latent factors($\mathbf{z}_c$)

In order to verify whether the extracted common latent factors $\mathbf{z}_c$ correspond to the ground truth common generative factors of G (C_f), we consider the scenario where graphs were generated with known common and non-common generative factors. Since it is very hard to access ground truth generative factors of real world graphs, we utilize two synthetic datasets for this purpose.

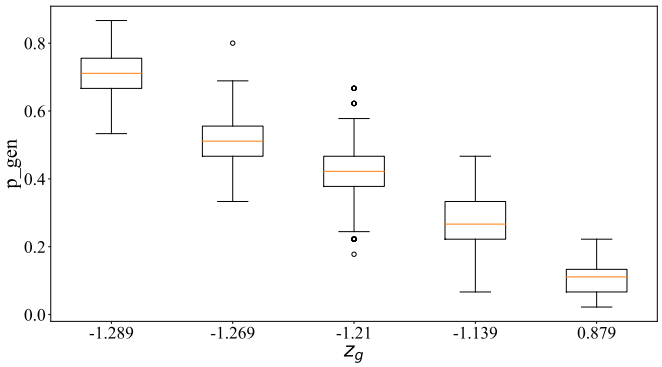


(A) $p = 0.3$ (B) $p = 0.5$

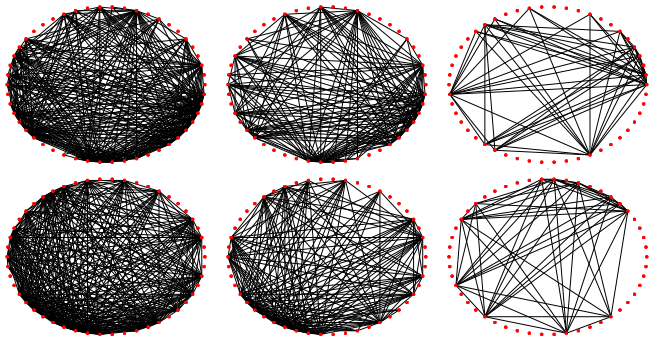


(C) $p = 0.7$

FIGURE 3.6: Variation of common global latent factor $\mathbf{z}_c$(Blue) and non-common local latent factor $\mathbf{z}_l$(Orange) of each generated graph with three different p values. Very minimum variation of $\mathbf{z}_c$ with fixed p demonstrates extracted $\mathbf{z}_c$'s correspondence to ground truth common generative factors.



(A) Impact on p_gen(p value of reconstructed graph) value with the increase of $\mathbf{z}_c$



(B) Variation of generated graphs with the increase of $\mathbf{z}_c$ value

FIGURE 3.7: Impact analysis of common latent representation $\mathbf{z}_c$ on the generation process of deepGCFX and recovering the ground truth common generative factor p. (A) plots how the distribution of the edge density probability p_gen (the recovered p) changes with the increase of $\mathbf{z}_c$ value. (B) visualizes generated graphs where in each row non-common latent representation $\mathbf{Z}_l$ is fixed and in each column $\mathbf{z}_c$ is fixed. This shows that $\mathbf{z}_c$ has a strong negative correlation with the ground truth common generative factor p.

Erdos-Renyi Graphs

First synthetic graph dataset was generated using Erdos-Renyi (ER) model (Erdos and Renyi, 1960). The ER(n, p) graphs are synthetic graphs with two global properties: number of nodes n and a parameter $p \in [0, 1]$ for the synthetic graph to include an edge (i, j) for $1 \leq i < j \leq n$ with probability p. There is randomness in each generated graph for a single p value due to many different edge combinations can represent p. In our experiments, we focus on evaluating parameter p as the common global generative factor C_f, as n is too easy to learn. Therefore, we create a training dataset of ER(n, p) with fixed $n = 50$ and varying p. We generated 4000 graphs in our dataset where 3000 is used for training and the remaining 1000 for testing in which adjacency matrix is the only input for deepGCFX. We use a simple 2 layer deepGCFX model with hidden

dimension size 2 and dimensions of $\mathbf{z}_c$ and $\mathbf{Z}_l$ are 1. We analyze two aspects from this experiment.

First is, whether extracted $\mathbf{z}_c$ corresponds to the fixed common global generative factor p while $\mathbf{Z}_l$ does not. To achieve this, we analyzed how $\mathbf{z}_c$ and $\mathbf{z}_l$ vary with fixed p. Fig. 3.6 shows scatter plots for 3 different values of p. From the testing set, we selected the set of graphs generated using the given p and we sampled $\mathbf{z}_c$ as the common latent representations for each graph. Then we calculated an accumulated single local latent representation ($\mathbf{z}_l$) for each graph by summing $\mathbf{z}_l(i),\ \forall i \in \{1 \dots |V|\}$ to observe non-common latent factor variation with p. We plotted the values of $\mathbf{z}_c$ and $\mathbf{z}_l$ for each graph for the given p in Figure 3.6. We can observe that while values of $\mathbf{z}_c$ are scattered within a very small range (almost similar to a constant) when the common global generative factor p is fixed, $\mathbf{z}_l$ demonstrates very high variation. This shows us that the extracted common latent factor $\mathbf{z}_c$ from deepGCFX has been able to accurately capture latent factors correspond to graph's original common generative factors. Since local latent factors are only optimized to capture non-common local factors, they were unable to capture common generative factors.

Next, we calculated the Pearson r correlation as well as co-variance to understand how $\mathbf{z}_c$ changes with p. We observed that $\mathbf{z}_c$ has a very strong negative correlation with p (Fig. 3.7 (A)) with correlation value 0.93 and the co-variance is 0.248. We further confirm this using the generative process of deepGCFX in Fig. 3.7(B) where we qualitatively visualize how the edge density of the generated graphs from $\mathcal{D}_{agg}$ vary with the increase of $\mathbf{z}_c$ values. For each row of Fig. 3.7(B), $\mathbf{Z}_l$ is fixed and for each column $\mathbf{z}_c$ value is fixed. From left to right, $\mathbf{z}_c$ value increases and we can observe that the edge density (how the common generative factor can be interpreted in the visualization) decreases accordingly. Node neighbourhoods based on connected edges differ in first row from the second due to different local latent factors $\mathbf{Z}_l$ fed to $\mathcal{D}_{agg}$ in each case.

Random Geometric Graphs

Next we use a more complex graph model to generate our synthetic dataset. The Random Geometric Graph (RGG) (Dıaz et al., 1999) model places n nodes uniformly at random in the unit cube space and if two nodes are at most at radius r distance from each other, an edge is created. RGG is considered as an abstractions of many real world graphs such as social networks and ad-hoc networks where nodes within close proximity are connected together than far away nodes. While each node of a given RGG has its own specific attributes of its position in the Euclidean space, the distance threshold value r is common for the entire graph. Therefore in this analysis, we consider r as our C_f, while position information of nodes are as ground truth local factors L_f. We generate a dataset of 10000 graphs with 50 nodes each and varying $r \in \{0.1, 0.2, \dots, 0.9\}$. We use 8000 graphs for training and 2000 for testing. deepGCFX with 128 hidden size and a 3 layer GNN is used. Combination of both node and adjacency matrix reconstruction losses are used as $\mathcal{L}_{\mathcal{D}_{agg}}$.

To analyze the correspondence between C_f and $\mathbf{z}_c$ in this case, we observed how the longest edge of the reconstructed graph from $\mathcal{D}_{agg}$ vary with the variation of $\mathbf{z}_c$. In Fig 3.8, we can observe that when $\mathbf{Z}_l$ is fixed and $\mathbf{z}_c$ is changed from a $\mathbf{z}_c$ of a graph whose r is lower(0.2) to higher(0.9), the node distance between the longest constructed edge

FIGURE 3.8: **Ability of deepGCFX in mapping extracted common latent factors $\mathbf{z}_c$ with the ground truth common generative factors C_f:** First row and first column (blue boxes) are reconstructed samples from original test set of our synthetic Random Geometric Graph dataset, where we treat the radius threshold r as common generative factor C_f and node positions as local factors. r values for first column are 0.2 and 0.9. The rest are swapped reconstructed samples (black boxes). Each row of black boxes is fixed $\mathbf{z}_c$ and each column is fixed $\mathbf{Z}_l$. With fixed $\mathbf{z}_c$ of $r = 0.2$ on first row of black boxes, we can see the edge lengths have been decreased and for the second row of black boxes, edge lengths have increased showing that extracted $\mathbf{z}_c$ has been able to capture the ground truth common factor C_f from original input.

also increases. This again shows that our deepGCFX can accurately model the correspondence between ground truth common generative factors of G (C_f) and extracted common latent factors $\mathbf{z}_c$.

3.7.4 Impact of learnt graph-wise common factors $\mathbf{z}_c$ on downstream graph level task performance.

To evaluate the discriminative ability of extracted common latent factors $\mathbf{z}_c$ on downstream tasks, we select graph classification. We select a widely popular graph benchmark of 6 datasets (MUTAG(Kriege and Mutzel, 2012), PTC-MR(Kriege and Mutzel, 2012), IMDB-BINARY, IMDB-MULTI, REDDIT-BINARY and REDDIT-MULTI-5K (Yanardag and Vishwanathan, 2015)) (Refer Sec. 3.6.1 for details). We evaluate our method against three different types of graph embedding approaches; One Explicit inter-graph similarity(GCKN-random walk (D. Chen, Jacob, and Mairal, 2020) and UGraphEmb (Bai et al., 2019)) based and two method types which do not utilize explicit inter-graph similarity. Those are Skip-gram (node2vec (Grover and Leskovec, 2016), sub2vec (Adhikari et al., 2018), graph2vec (Narayanan et al., 2017)) and contrastive learning (InfoGraph (F.-Y. Sun et al., 2020), CMV (Multi-view) (Hassani and Khasahmadi, 2020), GCC (Extra data) (Qiu et al., 2020) and GraphCL(Augmentations) (You, T. Chen, Sui, et al., 2020)). We report results for GVAE (Thomas N. Kipf and Welling, 2016b) as our baseline. We follow the evaluation protocol practice by all existing work for a fair comparison. 10-fold cross validation accuracy on SVM (Chang and C.-J. Lin, 2011) is used to report the performance and the mean accuracy and standard variation of 5 repeated runs is used as the final result. For hyper-parameter tuning, a validation split from 9 training data folds were used. Refer to Sec. 3.6.3 for complete details of the experiment.

In order to determine fairly the contribution of extracted common latent factors, our evaluation considers both extracted common latent factors $\mathbf{z}_c$ and local latent factors $\mathbf{Z}_l$ as potential contributors for graph embeddings. Since ground truth graph labels are not available during training of deepGCFX, it is not possible for the model to learn which latent would make the most discrminative graph embedding in the downstream task. There are three possibilities of making a graph embedding from extracted latents from trained deepGCFX; $\mathbf{z}_c$ only, $\mathbf{Z}_l$ only and combining $\mathbf{z}_c$ and $\mathbf{Z}_l$. While $\mathbf{z}_c$ contains graph-wise common factors, we know that $\mathbf{Z}_l$ contains non-common factors of the graph making $\mathbf{z}_l = \sum_{j=1}^{|V|} \mathbf{z}_l(j)$ a graph embedding of non-redundant local factors. Due to different characteristics of both embeddings, there is a potential in combining both in order to bring best of both worlds. To achieve that, we introduce a weight hyper-parameter α (best α was chosen using validation split along with other SVM related parameters.) to determine the amount of contribution coming from $\mathbf{z}_c$ (therefore, $1-\alpha$ is the contribution of $\mathbf{z}_l$). We report results for deepGCFX when only $\mathbf{z}_c$ is used as the graph embedding. deepGCFX$_l$ is when local factors $\mathbf{z}_l$ only is used as the graph embedding. deepGCFX++ model combines both common and local latent factors with a gating mechanism as $\alpha\mathbf{z}_c + (1-\alpha)\mathbf{z}_l$, where α denotes the contribution from graph-wise common factors.

Table 3.5 contains all results. Very last section of the table indicates results of three types of graph embeddings obtained by our proposed model; deepGCFX (graph embedding only consists of common factors), deepGCFX$_l$ (graph embedding only consists of local factors), deepGCFX++ (gated combination of common and local factors). When

DATASET	MUTAG	PTC-MR	IMDB-BIN	IMDB-MUL	RED-BIN	RED-MUL-5K
Explicit inter-graph similarity based methods						
GCKN-walk	**92.8 ± 6.1**	65.9 ± 2.0	**75.9 ± 3.7**	**53.4 ± 4.7**	–	–
UGraphEmb	-	**72.5**	-	50.1	–	–
Non-Explicit inter-graph similarity based methods						
Skip-gram based methods						
node2vec	72.6 ± 10.2	58.6 ± 8.0	–	–	–	-
sub2vec	61.1 ± 15.8	60.0 ± 6.4	55.3 ± 1.5	36.7 ± 0.8	71.5 ± 0.4	36.7 ± 0.4
graph2vec	83.2 ± 9.6	60.2 ± 6.9	71.1 ± 0.5	50.4 ± 0.9	75.8 ± 1.0	47.9 ± 0.3
Contrastive Learning based methods						
InfoGraph	89.0 ± 1.1	61.7 ± 1.4	73.0 ± 0.9	49.7 ± 0.5	82.5 ± 1.4	53.5 ± 1.0
CMV	89.7 ± 1.1	62.5 ± 1.7	74.2 ± 0.7	51.2 ± 0.5	84.5 ± 0.6	–
GCC	–	–	72.0	49.4	89.8	53.7
GraphCL	86.8 ± 1.3	–	71.1 ± 0.4	–	89.5 ± 0.4	**56.0 ± 0.3**
GVAE based methods						
GVAE(baseline)	87.7 ± 0.7	61.2± 1.8	70.7 ± 0.7	49.3 ± 0.4	87.1 ± 0.1	52.8 ± 0.2
deepGCFX (Ours) - α value for best results is in the brackets						
deepGCFX	89.8 ± 1.1	66.5 ± 1.0	72.9 ± 0.4	51.1±0.5	89.7±0.4	54.1±0.2
deepGCFX$_l$	87.1±0.7	60.6 ±1.5	70.2± 0.2	48.8±0.4	86.5±0.3	52.3±0.3
deepGCFX++	**92.2 ±0.9(0.7)**	**69.6± 1.4(0.85)**	**74.4± 0.2 (0.95)**	**52.7± 0.4(0.85)**	**90.9± 0.3(0.9)**	55.1± 0.2(0.85)

TABLE 3.5: Mean 10-fold cross validation accuracy on graph classification. Results in **bold** indicate the best accuracy for both inter-graph similarity based and non-inter-graph similarity based separately. Underlined results show the second best performances. We follow strictly the experiment and evaluation setup and datasets as in F.-Y. Sun et al., 2020; Hassani and Khasahmadi, 2020 for deepGCFX and GVAE baseline. Results of other methods are taken from their respective papers.

compared to deepGCFX$_l$, deepGCFX has achieved better results for all six datasets indicating that graph-wise common factors contain more useful features for the downstream graph classification than non-local factors. However we achieved our best results from deepGCFX++, which combines both common and local factors. This shows that both graph-wise common and local factors are beneficial to obtain the most discriminative graph embedding. However when combining those factors, we can observe from the final gating value α that the contribution from graph-wise common factors $\mathbf{z}_c$ are very high compared to local factors. This shows us that extracting graph-wise common latent factors is highly beneficial for graph level tasks. Next, we consider the results obtained by our GVAE baseline (sum of all nodes as the graph embedding. Same as $\mathbf{z}_l$). Overall, we can see that its performance is comparably lower than all deepGCFX embeddings which utilizes graph-wise common factors. However, GVAE performance is higher than local factors only deepGCFX$_l$. This is because, although both GVAE and deepGCFX$_l$ focuses on learning local neighbourhood related factors in their latent representations, GVAE latent representation is unable to differentiate and remove common latent factors like deepGCFX$_l$. Common latent factors are very crucial for graph level task performance and GVAE based embedding contains them. Hence its performance is better than deepGCFX$_l$ which does not have common latent factors.

Compared to state-of-the-art models of other non-explicit inter-graph similarity based graph embedding methods (skip-gram and contrastive learning), we can observe that deepGCFX (graph-wise common latent factors only) achieves very competitive or better results. Among skip-gram and contrastive learning, contrastive learning is the currently leading method. We consider InfoGraph (F.-Y. Sun et al., 2020) as the baseline model from them as it does not use any additional features like data augmentation or extra data. deepGCFX also does not use any additional features. Our extracted common latent factors only graph embedding (deepGCFX row) clearly surpasses InfoGraph highlighting the effectiveness of GCFX principle over infomax principle for graph level tasks. Our deepGCFX++ significantly surpasses all datasets except REDDIT-MULTI-5K for all latest contrastive learning methods which utilizes additional features as well showing the significance of deepGCFX. For REDDIT-MULTI-5K, GraphCL (You, T. Chen, Sui, et al., 2020) which uses additional features of dataset specific data augmentations achieves better performance than deepGCFX and deepGCFX++. However compared to those manually curated dataset specific features, our GCFX principle can be universally applied to any dataset.

Finally we compare deepGCFX against explicit inter-graph similarity based methods. Although inter-graph similarity is a different direction from our the objective of GCFX principle, its methods also achieve state-of-the-art performance. Hence for the completion of the evaluation, we compare deepGCFX with them. GCKN (D. Chen, Jacob, and Mairal, 2020) more complex input features based on kernel function achieves marginally higher average results for four datasets than deepGCFX++, but with very high variation. UGraphEmb (Bai et al., 2019) reports results for two datasets only and it surpasses deepGCFX for PTC-MR dataset. However training both these models are very expensive and time consuming. Authors of GCKN model, D. Chen, Jacob, and Mairal (2020) mention that they did not conduct experiments of very large datasets due to scalability issues inherent to pair-wise graph similarity calculation methods in general and kernel methods in specific. One of the major aspect of kernel methods is

they use manual processes (graph traversals like depth first search) to find all possible paths for substructures like random walks, trees or graphlets. Then they compare all those pairs of paths in each pair of graphs to calculate kernel values to find similarities. This is a very expensive operation. However for small graphs this gives better results as it covers all possible neighbourhoods. However as the GCKN (D. Chen, Jacob, and Mairal, 2020) mentions, when there are very large dense graphs, they are unable to extend this method. This can be a reason that kernel based methods do not evaluate on denser datasets like REDDIT. On the other hand, GNNs achieve efficiency by eliminating from manual path and graph to graph pairwise comparison and reducing neighborhoods for only random walks. However even with limited neighbourhood, we could see that GNNs specially with our common latent factor extraction mechanism have been able to achieve almost similar performance. Authors of UGraphEmb, Bai et al. (2019) mention that they conducted the training for 72 hours (3 days) for all the datasets to obtain reported results, showing the high time consumability of pair-wise inter-graph similarity based methods. In comparison, the maximum time required from deepGCFX was around 5 Hrs. That is only for REDDIT-MULTI-5K.

In conclusion, graph-wise common latent factors $\mathbf{z}_c$ is a very effective representation to obtain a graph embedding with discriminative abilities which can be utilized as graph embedding beneficial for downstream tasks. We achieve state-of-the-art performance surpassing existing contrastive learning models without using any additional features and very competitive results to pair-wise inter-graph similarity based models with higher efficiency and better scalability.

3.7.5 Impact of $\mathbf{z}_c$ for node level tasks

To analyze how common latent factors affect node level tasks, we select node classification task on both assortative (nodes belong to same class are remaining in close proximity neighbourhood) and disassotative (nodes from same class are distant from eachother). Since $\mathbf{z}_c$ is common for all the nodes in the graph despite the inter-node distance, we want to analyze whether it is beneficial to improve long-range node dependencies. Non-local aggregating (Pei et al., 2020; M. Liu, Z. Wang, and S. Ji, 2020) for graph learning has drawn attention in supervised graph representation learning research due to GNN's inability of long-range information propagation. Geom-GCN (Pei et al., 2020) proposed a benchmark to evaluate non-local aggregation for both assortative and disassortative graphs in the supervised setup.

For the best of our knowledge, existing unsupervised graph representation learning methods have not evaluated learnt node embeddings' capability for disassotative graphs. Hence, this benchmark haven't been tested on unsupervised based graph representations. Therefore when comparing model performances, we consider both supervised and unsupervised in order to showcase the current state of the art for this benchmark and how comparable deepGCFX is. We select two supervised (GCN(Thomas N. Kipf and Welling, 2017), Geom-GCN(Pei et al., 2020)) for the reference and two unsupervised (Deep Graph Infomax (DGI)(Velickovic, Fedus, et al., 2019) (base model for state of the art contrastive node embedding methods), GVAEThomas N. Kipf and Welling, 2016b) as our baselines. Cora, Citrseer and Pubmed (Z. Yang, Cohen, and Salakhutdinov, 2016) citation networks are used as assortative datasets for this benchmark while Chameleon and Squirrel topics from Wikipedia Network (Rozemberczki,

	ASSORTATIVE			DISASSORTATIVE		
DATASETS	**CORA**	**CITESEER**	**PUBMED**	**CHAMELEON**	**SQUIRREL**	**ACTOR**
Supervised Reference						
GCN	**85.77**	73.68	88.13	28.18	23.96	26.86
Geom-GCN	85.27	**77.99**	**90.05**	**60.90**	**38.14**	**31.63**
Unsupervised baselines						
DGI	**82.16 ±1.2**	**67.01± 1.3**	**81.34±0.6**	59.45±2.4	36.33±1.2	27.09± 1.2
GVAE	78.22 ±1.4	63.9 ±1.6	77.5± 0.7	56.88±2.9	33.05±1.6	25.12 ±1.4
deepGCFX - Ours						
deepGCFX	30.33± 1.2	20.75± 1.1	39.82± 0.5	19.30±2.7	19.23±0.8	10.5±1.2
deepGCFX$_l$	81.26 ±1.2	65.51 ±1.4	79.85± 0.7	57.67±3.1	35.64±1.7	26.88± 1.0
deepGCFX++	81.96 ±1.7(0.15)	66.71±1.6(0.1)	80.3± 0.7(0.2)	**61.05±2.4**(0.35)	**39.20±1.4**(0.4)	**28.80± 1.4**(0.4)

TABLE 3.6: Mean node classification accuracy for supervised and unsupervised models for assortative and disassortative graphs.Results in **bold** indicate best supervised and unsupervised accuracy for each dataset and underlined is the second best for unsupervised. Considerable increase of α value in deepGCFX++ from assortative to disassortative graphs highlights the important contribution graph-wise common latent factors provide for incorporating long-range node dependencies to improve node classification performance for disassortative graphs.

Allen, and Sarkar, 2021) and Actor co-occurrence network (Tang et al., 2009) are used as disassortative graphs. We follow the same evaluation protocol and data splits provided by Geom-GCN(Pei et al., 2020). Each dataset is divided into 60%, 20%, 20% train, validation and test splits and average classification accuracy of 10 random splits are used to report results. Complete experiment details can be found in Sec. 3.6.3.

Same as for graph level tasks, we evaluate all three possibilities to make node embeddings from extracted latents from trained deepGCFX; $\mathbf{z}_c$ only, $\mathbf{Z}_l$ only and combining $\mathbf{z}_c$ and $\mathbf{Z}_l$. While $\mathbf{z}_l(i),\ \forall i \in \{1 \dots |V|\}$ consist of non-redundant, patch specific local latent factors, we know that $\mathbf{z}_c$ contains latent factors common for all the nodes. Therefore, $\mathbf{Z}_l$ consists of factors which preserve inter-node similarity among neighbourhood nodes, while $\mathbf{z}_c$ could contain potentially beneficial factors which could indicate inter-node similarity among nodes which are spatially far away from eachother based on graph adjacency matrix. Due to different characteristics of both embeddings, there is a potential in combining both in order to bring best of both worlds. To achieve that, we utilize a weight hyper-parameter α (best α was chosen using validation split along with other SVM related parameters.) to determine the amount of contribution coming from $\mathbf{z}_c$ (therefore, $1 - \alpha$ is the contribution of $\mathbf{Z}_l$). We report results for deepGCFX when only $\mathbf{z}_c$ is used as node embedding. Embeddings for all the nodes in a given graph are similar in this case as $\mathbf{z}_c$ is a constant to the graph. deepGCFX$_l$ is when local factor set for all the nodes $\mathbf{Z}_l$ only are used as node embedding. deepGCFX++ model combines both common and local latent factors with a gating mechanism as $\alpha\mathbf{z}_c + (1-\alpha)\mathbf{z}_l(i),\ \forall i \in \{1 \dots |V|\}$, where α denotes the contribution from graph-wise common factors.

Table 3.6 reports results for all models. deepGCFX (only $\mathbf{z}_c$) reports the lowest results as expected since $\mathbf{z}_c$ is a constant to all the nodes. Hence $\mathbf{z}_c$ does not reflect node specific features. Compared to deepGCFX$_l$, deepGCFX++ which incorporates common latent factors for local factors achieve higher results demonstrating the effectiveness of common latent factors for node level tasks. For both assortative and disassortative datasets, our deepGCFX$_l$ performed better than GVAE. This is because while both models are based on local proximity, our non-redundant local latent factors are less noisy compared to GVAE. GVAE only pays attention to proximity information, hence node features which do not contribute to proximity have the tendency to be ignored by GVAE. When comparing our proposed deepGCFX based embeddings against DGI, we observe that for assortative graphs, deepGCFX++ has achieved best results for disassortative graphs highlighting extracted common latent factors' ability in enabling long-distance node dependencies. In the next paragraph we analyze the underlying principle difference which caused this result.

DGI is a contrastive learning based model (based on Infomax principle (Linsker, 1988)), from which many later models such as Infograph(F.-Y. Sun et al., 2020), CMV(Hassani and Khasahmadi, 2020), GraphCL(You, T. Chen, Sui, et al., 2020) got inspired. Since deepGCFX doesn't employ additional features such as data augmentations, larger pretrained datasets or multiple views of data, we selected DGI as the most similar model to ours from contrastive learning domain for node tasks. Main difference between DGI and our deepGCFX is the underlying principles these models are based on. DGI depends on maximizing mutual information between node/patch representations and

corresponding high-level graph summary. deepGCFX depends on extracting common factors from local factors. DGI increases each node's mutual information with the graph summary despite inter-node distance, hence Velickovic, Fedus, et al. (2019) suggest that DGI has the potential to enable long range node dependencies. However, an indirect effect of this maximizing mutual information between graph summary and nodes is that, this further increases the mutual information among neighbourhood nodes which are already somewhat similar due to GNN's information propagation. This over-increase in neighbour node similarity is beneficial for assortative graphs compared to extraction of common factors in deepGCFX. Hence, DGI have slight performance gain than deepGCFX for assortative graphs. However, for disassortative graphs where nearby nodes have different labels, this overemphasize on neighbour similarity in DGI becomes harmful (M. Liu, Z. Wang, and S. Ji, 2020) as it hinders node specific features which are different from neighbours. Hence DGI's performance degrades. But deepGCFX can mitigate that weakness successfully as it intentionally preserves node specific non-common factors useful to distinguish the nodes while extracting graph-wise common latent factors. Hence, we get better performance. This shows the effectiveness of GCFX principle over Infomax principle for non-local networks.

3.7.6 Ablation study on the impact of number of accumulation iterations on downstream task performance

DATASET	MUTAG	IMDB-MUL
i=0	$87.2 \pm 1.2(0.1)$	$48.8 \pm 0.5(0.0)$
i=1	$89.1 \pm 0.9(0.4)$	$49.4 \pm 0.3(0.3)$
i=2	$\mathbf{92.2} \pm \mathbf{0.9}(0.7)$	$51.2 \pm 0.4(0.7)$
i=3	$90.5 \pm 1.2(0.8)$	$\mathbf{52.7} \pm 0.4(0.85)$
i=4	$89.3 \pm 0.9(0.4)$	$50.8 \pm 0.5(0.7)$

TABLE 3.7: Mean 10-fold cross validation accuracy on graph classification with varying number of accumulation iterations by deepGCFX++ (α in brackets). Best results are indicated in **bold**.

In Section 3.7.2 we demonstrated the requirement of iterative procedure for our proposed query based common latent extraction accumulation (ACCUM) (described in Sec. 3.4.2) in order to extract common factors properly filtering local factors out. Now we analyze how the number of accumulation iterations impact the downstream task performance. We report results of the best combined results obtained by deepGCFX++ for MUTAG and IMDB-MULTI graph classification datasets. We observe that at iteration 0, where the graph-wise common factors haven't been learnt and only initialized with random values, has the lowest performance and the α(amount of contribution from common latent factors) is also very low as the filtering of common and local factors are done at random. Hence the model tend to ignore $\mathbf{z}_c$ and only learn $\mathbf{z}_l$ based on proximity. Therefore for the downstream task performance also, almost entire contribution comes from $\mathbf{z}_l$. With the increase of iterations where $\mathbf{z}_c$ gets learnt with common and graph relevant information, the performance increases as well as the contribution from $\mathbf{z}_c$ to final result and then we observe a fluctuation in performance. This shows that with more than one iteration, $\mathbf{z}_c$ starts to capture better graph relevant information

and its contribution improves the discriminative quality of the graph embedding more than using local latent factors $\mathbf{z}_l$ only.

3.8 Summary

Unsupervised graph-level representation learning plays a crucial role in a variety of tasks such as molecular property prediction and community analysis, specially when data annotation is expensive. Currently, most of the best performing graph embedding methods are based on Infomax principle. Performance of these methods highly depend on the selection of negative samples and hurt the performance if the samples were not carefully selected. Inter-graph similarity based methods also suffer if the selected set of graphs for similarity matching are low in quality. To address this, we focus only on utilizing the current input graph for embedding learning. We are motivated by an observation from real world graph generation process where the graphs are formed based on one or more global factors which are common to all elements (Ex: topic of a discussion thread, solubility level of a molecule) of the graph. We hypothesize extracting these common factors could be highly beneficial. Hence, this work proposes a new principle for unsupervised graph representation learning: Graph-wise Common latent Factor EXtraction (GCFX). We further propose a deep model for GCFX, deep-GCFX, based on the idea of reversing the above mentioned graph generation process which could explicitly extract common latent factors from an input graph and achieve improved results on downstream tasks to current state-of-the art. Through extensive experiments and analysis we demonstrate that, while extracting common latent factors is beneficial for graph level tasks to alleviate distractions caused by local variations of individual nodes or local neighbourhoods, it also benefits node level tasks by enabling long-range node dependencies specially for disassortative graphs.

Chapter 4

Situation Recognition via Graph and Transfer Learning

We move to applications of deep learning on graph structured data from this chapter on-wards. Our focus is on visual reasoning tasks which require advanced multi-modal reasoning capabilities and analyze how those can be further improved by modeling them as graph learning problems. We give an overview of our selected task, Situation Recognition and its task specific challenges which motivated us on our visual reasoning enhanced inter-dependent query reasoning based neighbourhood information propagation. We propose two different approaches for this influenced by transfer learning and graph learning. We discuss our first approach in this chapter and the second method is explained in Chapter 5.

4.1 Introduction

Visual reasoning is the process of analyzing visual information in order to achieve a final conclusion. There are a variety of visual reasoning tasks being researched in the computer vision domain beginning with the basic building blocks of object (Krizhevsky, Sutskever, and Hinton, 2012; Simonyan and Zisserman, 2015; Ren et al., 2015; Y. Guo and Cheung, 2018) and action (Carreira et al., 2016; Sharma, Jurie, and Schmid, 2013; Song et al., 2018) classification. Scene Graph Generation (Johnson, Krishna, et al., 2015; Yikang Li et al., 2017; D. Xu, Y. Zhu, Christopher B Choy, et al., 2017a) was introduced in order to expand the visual reasoning capabilities of computer vision models beyond mere object and action classification and brought visual reasoning to the next level by combining all the predicted visual relations in an image and constructing a knowledge graph out of it.

However, these relations in scene graphs were captured in a triplet (subject-predicate-object) manner which limits the expressibility when it comes to describe actions, as the objects participate in an action expand beyond subject and object elements. In order to address this limitation, Yatskar, Zettlemoyer, and Farhadi (2016) introduced *Situation Recognition (SR)*. In SR, the model is expected to not only predict the salient action of the image, but also predict all the objects that participate in the action. Relationships between individual objects and the action are indicated by a concept called *semantic roles*. A situation is a structure which comprises of an action along with its semantic roles making this a structured prediction task.

Figure 4.1 shows two instances of action "Brushing" in the *imSitu* dataset (Yatskar, Zettlemoyer, and Farhadi, 2016), the prime dataset for SR. Semantic roles of "Brushing"

Brushing			
Role	**Value**	**Role**	**Value**
Agent	Woman	Agent	Man
Target	Hair	Target	Teeth
Tool	Brush	Tool	Toothbrush
Substance	-	Substance	Toothpaste

FIGURE 4.1: Situation recognition (SR) (Yatskar, Zettlemoyer, and Farhadi, 2016): Two different situations for the same action (verb). The SR task is to predict the action (verb) and the values of all the associated semantic roles.

are *agent* (person who is brushing), *target* (entity or object the agent is brushing), *tool* (the tool being used for brushing), *substance* (any substance being used for brushing). Note that *place* is also a semantic role for brushing, but we omit it in this example for clarity as it is not significant here. Also note that different actions may have different semantic roles. For example, action "eating" has roles: *food*, *place*, *container*, *agent*, *tool*. SR is a very challenging reasoning task, as the number of different role types and possible values are very large (Mallya and Lazebnik, 2017; R. Li et al., 2017). Furthermore, even for the same action (verb), the possible values for individual roles can be very different as illustrated in Figure 4.1.

Semantic role prediction has drawn the most attention compared to action prediction due to its more challenging requirement of capturing all action related objects in the image, regardless of its visible salience. Existing work has focused on modelling inter-dependency among semantic roles using Recurrent Neural Networks (Mallya and Lazebnik, 2017) and Graph Neural Networks (R. Li et al., 2017). They have not given much attention on how to improve the multi-modal reasoning capabilities for this task.

In this work, we take a radically different approach for SR inspired by another visual reasoning task. The Visual Question Answering (VQA) task takes an image and a natural language question about the image, and outputs a natural language answer (Y. Goyal et al., 2017b; Johnson, Hariharan, et al., 2017; Hudson and Christopher D Manning, 2018; Anderson et al., 2018; J.-H. Kim, Jun, and B.-T. Zhang, 2018; Cadène et al., 2019). Fig. 4.2 shows example images from VQA 2.0 dataset (Y. Goyal et al., 2017a) along with their question lists. This is the widely used standard benchmark dataset for VQA. We can observe that the images and questions have some resemblance to situation recognition. They query about object counts, object properties and actions. These urge VQA systems to acquire many AI capabilities such as object recognition, object detection, activity recognition and commonsense reasoning. These capabilities are vital for SR as well.

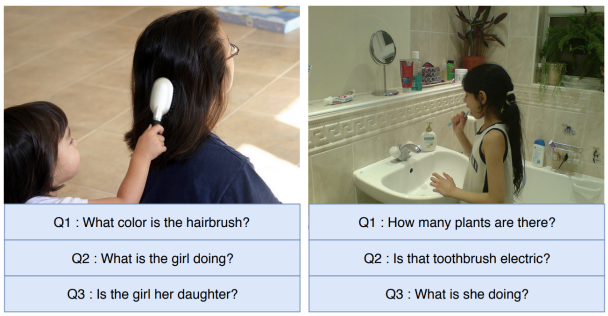


FIGURE 4.2: VQA 2.0 (Y. Goyal et al., 2017a) dataset sample images and questions.

Therefore, inspired by query-based visual reasoning models (Y. Goyal et al., 2017b; Johnson, Hariharan, et al., 2017; Zheng, Yali Li, and S. Wang, 2018; Krishna, Bernstein, and Fei-Fei, 2019) which have proven to be successful in analyzing an image conditioned on a given query (natural language question, object name etc.) to obtain an answer, we propose to model SR as a query-based visual reasoning task. However, we argue that there are subtle differences between VQA and SR that require careful design considerations:

- **Question dependency.** In VQA, questions are asked and scored independently. However, for SR, there is dependency between different verbs and roles. Thus, it may be more appropriate to ask *related* questions to accomplish SR. As an example, in first image of Figure 4.1, suppose the answer of "what is the agent?" is "woman", this answer can be used to ask about the action as "what is the woman doing?". Then the answer "brushing" can be used as context in the role label prediction questions. In particular, a complex question such as "what is the tool used for brushing?" could be used instead of a simple and general question "what is the tool?". Therefore it is essential to answer questions in a context aware manner for SR.

- **Question correctness.** However, a unique challenge for automatic generation of context aware questions is the accuracy of the context included in the question. Specifically, in recent VQA datasets, questions are composed by human and are assumed to be correct, e.g. in VQA "what color is the hairbrush?", there is indeed a hairbrush in the image. However, in a complex question where the context of the question is provided by *the answer of another question via a VQA system*, this complex question may not be always correct. For example, in Figure 4.1, suppose the VQA system returns an incorrect answer "drying" to the question "what is the woman doing?". Then the question "what is the tool used for drying?" is incorrect, and the system may perform poorly. This is also an important aspect which needs to be considered in SR.

Based on above discussion, VQA and SR can be regarded as different but related problems. This chapter discusses about two different approaches that we propose to address SR as a VQA problem. First is, a transfer learning approach to apply the

learned knowledge from VQA to address SR. This is not only a novel approach to address SR, but also a novel application of VQA. **To the best of our knowledge, this is the first attempt of transfer learning from a VQA system to another visual reasoning task. This is also the first model which utilizes transferred visual reasoning to address situation recognition.**

In the second approach, we propose a novel soft query based visual reasoning model (which does not require natural language questions like the first approach) with a novel contextualization module to incorporate information from related queries to address inter-query relational reasoning. **We make the first effort of inter-dependent question answering where our proposed contextualization mechanism explicitly allows both multi-modal reasoning and neighbour information integration together. This enables the model to dynamically combine the information for optimal predictions. We propose a method to generate the context using attention, and propose different mechanisms to incorporate the generated context to improve reasoning.**

4.2 Related Work

Yatskar, Zettlemoyer, and Farhadi (2016) introduced the SR task along with the *imSitu* dataset whose actions and frames are based on FrameNet (Baker, Fillmore, and Lowe, 1998). They proposed a baseline model which consists of a Convolutional Neural Network (CNN) (Lecun and Y. Bengio, 1995) for image encoding followed by a Conditional Random Field to predict actions and labels for semantic roles. As mentioned by Yatskar, Zettlemoyer, and Farhadi (2016), this dataset suffers from huge sparsity issues in both object labels as well as situations because some objects can participate in many roles while other objects can only be seen few times. To address this sparsity issue, Yatskar, Ordonez, et al. (2017) later proposed another model which maps roles and labels to a lower dimensional vector space and have also used additional images to reduce data sparsity. Then two models were presented by Mallya and Lazebnik (2017) and R. Li et al. (2017) focusing on improving role predictions by explicitly modelling dependency among semantic roles. Mallya and Lazebnik (2017) use a Recurrent Neural Net to model role dependencies and predict labels as a sequence labelling problem while using a Fusion Network (Mallya and Lazebnik, 2016) for action prediction. R. Li et al. (2017) argue that all roles in a frame should depend on each other without manually assigning any priority to roles like in sequence labelling. Therefore they propose a Gated Graph Neural Network (GGNN) (Yujia Li, Tarlow, Brockschmidt, and R. Zemel, 2015) based role modelling method. These two models achieve the highest results for frame prediction emphasizing the importance of modelling role inter-dependency for this task.

On the subject of improving multi-modal reasoning for independent query predictions, Visual Question Answering (VQA) (Y. Goyal et al., 2017b; Johnson, Hariharan, et al., 2017; Hudson and Christopher D Manning, 2018; Anderson et al., 2018; J.-H. Kim, Jun, and B.-T. Zhang, 2018; Cadène et al., 2019) task leads the way with numerous highly capable multi-modal reasoning methods. Inspired by these, we utilize a very simple, but effective VQA method by Anderson et al. (2018) to fill the lack of sophisticated multi-modal reasoning application in SR. However, existing VQA tasks only require answering questions independently or use answers from previous questions

to answer the current question (ex: Visual Dialog Das, Kottur, et al., 2017 and Visual Commonsense Reasoning (VCR) (Zellers et al., 2019)). SR stands out from these as mentioned earlier that each role (the query to which we try to find an answer) depends on all other roles of its action without any defined order like in Visual Dialog or VCR.

Inter-dependent question answering is a novel requirement in SR which has not been raised before. We believe this has the potential to be useful for other tasks such as Embodied Question Answering (Das, Datta, et al., 2018) in multi-agent environments where agents can utilize information from each-other along with its own surrounding to answer questions. Therefore in this work, we propose several models which are capable of inter-dependent VQA, aiming to solve semantic role prediction in SR.

Visual Semantic Role Labelling (VSRL) is a task which goes hand in hand with Situation Recognition. VSRL was first introduced by S. Gupta and Malik (2015) where they had annotated MSCOCO (T.-Y. Lin, Maire, et al., 2014) dataset for 26 actions and localized 3 roles; agent, object and instrument. Motivation behind VSRL was to get a thorough understanding about actions by being able to reason on objects and people related to it. This vision was brought forward by Yatskar, Zettlemoyer, and Farhadi (2016) by introducing a more comprehensive dataset consisting of 504 actions and 190 unique semantic roles extracted from FrameNet (Baker, Fillmore, and Lowe, 1998).

Grounding semantic roles in images is another related task. S. Yang et al. (2016) have proposed a method and a dataset to ground objects in video clips referring to semantic roles in a given sentence. This differs from our task as we do not use sentences to first find out verb and labels for its semantic roles. Silberer and Pinkal (2018) have introduced another semantic role grounding dataset based on *Flickr30k Entities* (Plummer et al., 2015) dataset. Their task is to select the most relevant region for each semantic role of the given frame from a set of image regions. Our proposed approach can be applied to this task as well. However we are unable to evaluate as the dataset is not released to the public.

4.3 Formal Task Definition of Situation Recognition

Situation Recognition defines a space which consists of a discrete set of verbs V, nouns N, roles R and frames F. Each verb $v \in \{1, \ldots, |V|\}$ is mapped with a frame $f \in F$ which consists of semantic roles $R_v \subset R$. Each semantic role is paired with a noun value $n \in N \cup \{\emptyset\}$. An instance of an action v in an image I forms a realized frame $F_{(I,v)} = \{(r_i, n_i) : r_i \in R_v, n_i \in N \cup \{\emptyset\}, i = 1, \ldots, \mid R_v \mid\}$. Given an image, the full task of SR is to predict the pair of action and its associated realized frame which is called a situation $S = \{v, F_{(I,v)}\}$. Action prediction is considered as a separate classification task independent from role prediction in existing work (Mallya and Lazebnik, 2017; R. Li et al., 2017).

4.4 Transfer learning from VQA

In this section we explain our first methodology for query based visual reasoning for SR. This is a transfer learning procedure from a VQA model. Our proposed model not only incorporate advanced multi-modal reasoning capabilities to SR, it also enhances this transfer learning procedure to graph data structures.

4.4.1 Main Contributions

- To the best of our knowledge, this is the first attempt of transfer learning from a VQA system to another visual reasoning task.
- This is also the first model which utilizes transferred visual reasoning to address situation recognition.
- We use contextual information obtained from semantic role labels for action prediction instead of only relying on image features like existing SR systems.

4.4.2 Source dataset

We use the popular VQA 2.0 (Y. Goyal et al., 2017b) real image dataset as our source dataset to train the VQA model. This dataset consists of more than 204K images, 614K natural language questions in English and more than 6 million answers. The dataset consists of variety of questions mainly categorized by the expected answer types as "yes/no" (38%), "numbers" (13%) and "other" (49%). Question types that we would require in our target task belongs to the "other" type. Most of the questions in "other" type are object detection, and action related questions are limited. However, the object detection ability is a vital capability for identifying labels for semantic roles in the target dataset and we believe the knowledge about objects in the image can act as context to benefit action detection. The Bottom-up Top-down (BuTd) model we are using by Anderson et al. (2018) was the winner of 2017 VQA challenge on this dataset.

4.4.3 Source model

We started with the Bottom-up Top-down Attention (BuTd) (Anderson et al., 2018) model as our source VQA model which we will transfer to our target Situation Recognition task. Main reason for selecting this over the latest Bilinear Attention Network (BAN) (J.-H. Kim, Jun, and B.-T. Zhang, 2018) is its ability to perform well with both Faster R-CNN (Ren et al., 2015) object region features as well as grid based features. Bottom-up attention is used to extract object regions from Faster R-CNN and the well established Top-down attention mechanism which assigns normalized attention weights to all the target features based on a condition and get the weighted sum of features as the potential answer, used to highlight important visual features given the encoded question. In our experiments we use grid features from CNN as image features, hence only utilizing top-down attention from BuTd model. Here we explain Top-down attention in detail.

In Top-down attention, first we obtain image region features $\mathbf{E}_I = \{\mathbf{e}_n\}_{n=1}^{N_e}$ by encoding the image $\mathbf{I}$ using a CNN and obtaining the grid features just after the last pooling layer. N_e is the number of regions of the image. Word embeddings for words in the natural language question $\mathbf{Q}$ are fed to LSTM network to obtain the query encoding $\mathbf{q}$.

$$\mathbf{E}_I = \text{CNN}(\mathbf{I}), \tag{4.1}$$

$$\mathbf{q} = \text{LSTM}(\mathbf{Q}), \tag{4.2}$$

where $\mathbf{E}_I \in \mathbb{R}^{N_e \times d\_img}$ and f_q is a non-linear layer. $[\cdot]$ is used to denote the concatenation. $\mathbf{q} \in \mathbb{R}^{d\_q}$.

Then we calculate the image region-level attention weights based on the query encoding, and derive updated image encoding,

$$s_n = \mathbf{w}_a f_a([\mathbf{e}_n, \mathbf{q}])^T, \tag{4.3}$$

$$\alpha_n = \frac{\exp(s_n)}{\sum_{i=1}^{N_e} \exp(s_i)}, \quad \tilde{\mathbf{E}} = \sum_{n=1}^{N_e} \alpha_n \mathbf{e}_n, \tag{4.4}$$

s_n denotes un-normalized region-level attention weights obtained for current query $\mathbf{q}$. α_n denotes the normalized attention weight for region n, and $\tilde{\mathbf{E}}$ is the aggregated image encoding for the query. $\mathbf{w}_a \in \mathbb{R}^{d\_hidden}$ are model parameters and f_a is a non-linear layer.

Then updated image encoding $\tilde{\mathbf{E}}$ and query encoding $\mathbf{q}$ are fused together to obtain the un-normalized hidden representation $\mathbf{h_u} \in \mathbb{R}^{d\_hidden}$,

$$\mathbf{h_u} = f_{pq}(\mathbf{q}) \circ f_{pi}(\tilde{\mathbf{E}}), \tag{4.5}$$

where f_{pq} and f_{pi} non-linear layers are used to project query and image encoding to a different space and $\circ$ denotes element-wise multiplication.

Element-wise multiplication can cause model convergence to an unsatisfactory local minimum (Z. Yu et al., 2018). In order to avoid this, Z. Yu et al. (2018) have used the power normalization ($z \leftarrow \text{sign}(z)|z|^{0.5}$) and ℓ_2 normalization ($z \leftarrow z/\|z\|$) layers. Following their approach, we also modified the original TDA model by adding a Dropout (Srivastava et al., 2014) layer and normalization after element-wise multiplication to produce the *normalised hidden representation* $\mathbf{h}$:

$$\mathbf{h} = \ell_2\text{Norm}(\text{PowerNorm}(\text{Dropout}(\mathbf{h_u}))), \tag{4.6}$$

Classifier Finally the normalized hidden representation is sent through a non linear network $f_{classifier}$ followed by a SoftMax function to obtain final probability distributions of each role label prediction.

$$p = \text{SoftMax}(f_{classifier}(\mathbf{h})), \tag{4.7}$$

Learning and Inference Cross entropy loss is used to train the model.

4.4.4 Target Dataset

Yatskar, Zettlemoyer, and Farhadi (2016) released the *ImSitu* dataset while introducing Situation Recognition task. This dataset consists of 126,102 images with annotations for 504 actions and 190 unique roles annotated using FrameNet (Baker, Fillmore, and Lowe, 1998). This dataset contains considerable amount of action annotation overlaps, however the image set is balanced in terms of actions. Dataset suffers from lots of sparsity (Yatskar, Zettlemoyer, and Farhadi, 2016) regarding the role label annotations as it is not practical to gather all possible role-object combinations. Action and semantic

role names are single word. In addition, the authors have provided descriptions about roles and how they associate with the action. We leverage these meta information in our transfer learning procedure.

4.4.5 Challenges in transferring knowledge from VQA to SR

Situation Recognition is not merely another VQA dataset as explained above. Hence, transferring knowledge from VQA is quite challenging. First challenge is question dependency. VQA normally handles questions independently despite multiple questions were asked about the same image. On the other hand in SR, semantic roles depend on both its action (A) as well as the neighbouring roles (N) of that frame. From the example in Fig. 4.3 we can observe how incorporating action and neighbour role dependencies can help to narrow down search space.

Context		Question	Possible answers
A	N		
✗	✗	What is the tool ?	Phone, toothbrush, hair brush, hair dryer …
✓	✗	What is the tool brushes?	Toothbrush, hair brush
✓	✓	What is the tool woman brushes hair with?	Hair brush

FIGURE 4.3: Example of querying about semantic role “Tool" and how the answer space narrows down when the question is conditioned on the action (A) and neighbouring roles (N)

However, unlike VQA which has human generated questions, SR dataset does not provide natural language questions. Hence we have to automatically generate those questions. This means we have to rely on an action and semantic role prediction models to provide us the context of the question. This can be error-prone. As shown in Fig. 4.4, if the action predictor (A) provided an erroneous answer, it will badly effect the role prediction. Keeping this in mind, we discuss the question generation process in the next section.

Context	Question	Possible answers
A		
✗	What is the tool ?	Phone, toothbrush, hair brush, hair dryer …
Incorrect	What is the tool woman drying hair with?	Hair dryer

FIGURE 4.4: Example of adverse impact on the final predictions when the used context of the generated question is wrong

4.4.6 Question Generation

We leverage meta information provided by *ImSitu* (Yatskar, Zettlemoyer, and Farhadi, 2016) dataset for question generation. These meta information about verbs contains how each semantic role in verb fits together in a natural language sentence and the simple textual meaning of each role. Table 4.1 shows an example of these details for the action "writing".

Roles	*ImSitu* meta information
Full frame	AGENT writes on TARGET using a TOOL at a PLACE
Agent	The entity doing the write action
Target	The object on which writing is applied
Tool	The tool used for writing
Place	The location where the write event happening

TABLE 4.1: Meta information provided by situation recognition dataset *ImSitu* (Yatskar, Zettlemoyer, and Farhadi, 2016) for action "writing".

First we generate independent questions using the definition of each single role provided in meta data. Next, we need complex questions which contain information about verb and role dependency. We use a template based approach for generating those questions, where we complete questions by filling the slots using label predictions. We programatically generate the templates for context aware questions using frame descriptions provided in *ImSitu* dataset meta information. Table 4.2 shows general questions and templates for context aware questions generated for the action "writing".

Roles	Generated General Questions and Context Aware Question Templates
Agent	who is doing the write action? who is the agent writes on <TARGET> using <TOOL> at <PLACE>?
Target	on what object the writing is applied? what is the target <AGENT> writes on using <TOOL> at <PLACE>?
Tool	what is the tool used for writing? what is the tool <AGENT> writes on <TARGET> using at <PLACE>?
Place	where is the write event happening? where is the place <AGENT> writes on <TARGET> using <TOOL>?

TABLE 4.2: Generated questions for action "writing"

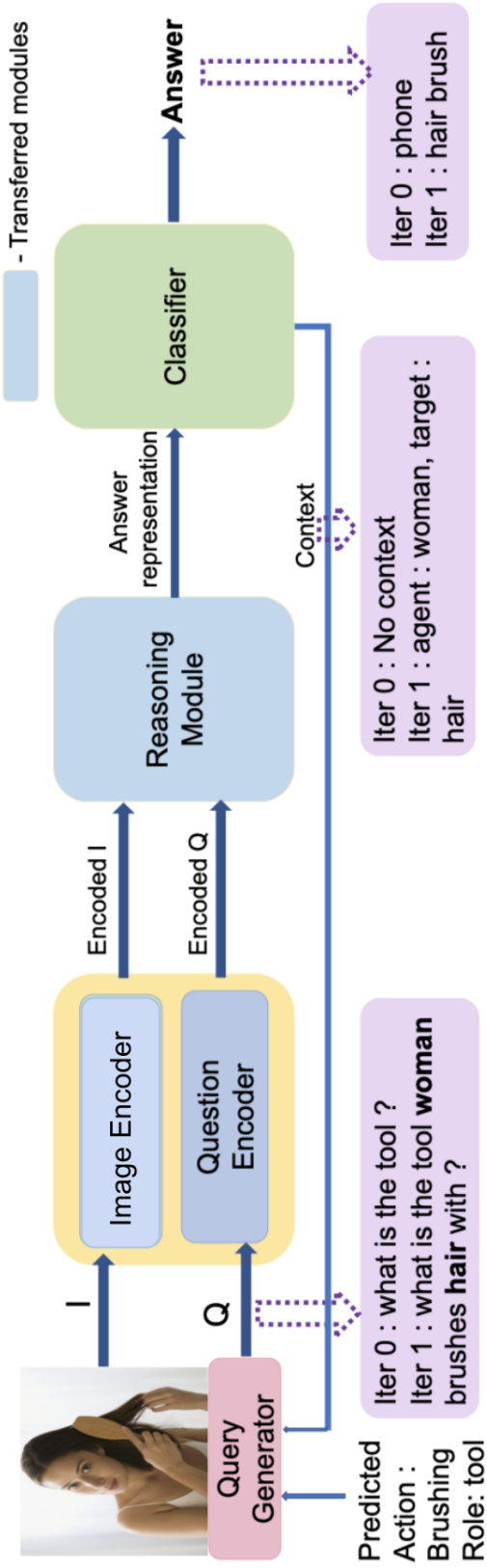


FIGURE 4.5: Proposed model architecture for Transfer learning from VQA to SR

4.4.7 Proposed Transfer Learning model from VQA to SR

Fig. 4.5 shows our proposed model. Image encoder is used to convert original image to its tensor representation. We use a LSTM network to encode the natural language question. We use Top-Down attention as the reasoning module. All these components are transferred from a pretrained Bottom-up Top-down Attention (BuTd) (Anderson et al., 2018) model as our source VQA model which we will transfer to our target Situation Recognition task. Top-down attention mechanism which assigns normalized attention weights to all the target features based on question and get the weighted sum of features as the potential answer is used to highlight important visual features given the encoded question. Finally, we have the classifier for predictions.

SR as VQA is an iterative process, where we insert the model with our general independent question in the first iteration. For an example the question can be "what is the tool?". Then in the second iteration, we use the predictions from the first iteration as the context and fill our context aware question template with them. This is an end to end process and the model is optimized using cross entropy loss.

4.4.8 Implementation Details

We used the Pytorch implementation of BuTd [1] for training the source VQA model. However we used gated hyperbolic tangent activations (Dauphin et al., 2017) in non-linear transformation instead of ReLU following the original paper (Anderson et al., 2018). Grid features for images were extracted using ImageNet (Deng et al., 2009) pre-trained ResNet-50(He et al., 2016) finetuned to *ImSitu* dataset. We finetuned the pre-trained VQA model for the target task with Adamax and exponential learning rate scheduling with $\gamma = 0.9$. For both verb and role label prediction models, we randomly initialized the classifier and used GloVe embeddings (Pennington, Socher, and Christopher D. Manning, 2014) for word representations. Both were trained with initial learning rate of 10^{-3}. Question encoder was finetuned with initial learning rate of 1×10^{-4} and the rest of the model (reasoning module) with 5×10^{-5} initial rate. We used mini-batch size of 64 and obtained the best model by early stopping using development set performance. We trained the role label model to predict the most frequent 2000 nouns following R. Li et al. (2017) as it cover up to 95% of noun occurrences of the training set and only using these 2000 nouns we can represent at least one valid situation annotation in 98% of training set images.

4.4.9 Evaluation

We follow the experiment setup and evaluation criteria from Yatskar, Zettlemoyer, and Farhadi (2016). Here we report results for three metrics. verb: verb prediction, value: verb-role-label tuple is considered correct, if it matches any of the 3 annotations, value-all: when the entire situation is correct, meaning all verb-role-value tuples of the verb matches at least one ground truth annotation. Accuracy % is the performance metric. *ImSitu* dataset contains 75K train, 25K development and 25K test set samples.

We evaluate the performance of transferred models for several criteria. First, we evaluate verb prediction (i) when the iterative model only runs for a single iteration

[1] https://github.com/hengyuan-hu/bottom-up-attention-vqa

	top-1 predicted verb			top-5 predicted verbs			ground truth verbs		mean
	verb	value	value-all	verb	value	value-all	value	value-all	
Verb Prediction Only									
Context aware q ($T = 1$)	34.78	-	-	62.14	-	-	-	-	-
Context aware q ($T = 2$)	35.24	-	-	62.99	-	-	-	-	-
Role Label Prediction Only									
General questions	-	-	-	-	-	-	72.73	37.07	-
Context aware questions	-	-	-	-	-	-	73.19	37.67	-
Reference Model									
Gold questions	43.08	34.05	20.64	69.12	53.26	30.35	74.00	38.48	45.37
Comparison with Existing Work									
CNN + CRF (Yatskar, Zettlemoyer, and Farhadi, 2016)	32.25	24.56	14.28	58.64	42.68	22.75	65.90	29.50	36.32
Tensor Composition (Yatskar, Ordonez, et al., 2017)	32.91	25.39	14.87	59.92	44.50	24.04	69.39	33.17	38.02
Above + DataAug (Yatskar, Ordonez, et al., 2017)	34.2	26.56	15.61	62.21	46.72	25.66	70.80	34.82	39.57
RNN (Mallya and Lazebnik, 2017)	36.11	27.74	16.60	63.11	47.09	26.48	70.48	35.56	40.40
GGNN † (R. Li et al., 2017)	**36.83**	**28.31**	16.55	**63.48**	47.27	25.77	69.63	33.58	40.18
Our model	35.24	27.77	**16.80**	*62.99*	**48.43**	**27.59**	**73.19**	**37.67**	**41.21**

TABLE 4.3: Situation prediction results on *ImSitu* development set. † denotes results of our implementation. **Bold** and Underlined indicate best and second best performances. T refers to the number of timesteps

	top-1 predicted verb			top-5 predicted verbs			ground truth verbs		mean
	verb	value	value-all	verb	value	value-all	value	value-all	
CNN + CRF (Yatskar, Zettlemoyer, and Farhadi, 2016)	32.34	24.64	14.19	58.88	42.76	22.55	65.66	28.96	36.25
Tensor Composition (Yatskar, Ordonez, et al., 2017)	32.96	25.32	14.57	60.12	44.64	24.00	69.20	32.97	37.97
Above + DataAug (Yatskar, Ordonez, et al., 2017)	34.12	26.45	15.51	62.59	46.88	25.46	70.44	34.38	39.48
RNN (Mallya and Lazebnik, 2017)	35.90	27.45	16.36	63.08	46.88	26.06	70.27	35.25	40.16
GGNN † (R. Li et al., 2017)	**36.97**	**28.21**	16.27	**63.62**	47.16	25.32	69.34	33.29	40.02
Our model	34.91	27.55	**16.74**	63.10	**48.46**	**27.45**	**73.08**	**37.60**	**41.11**

TABLE 4.4: Situation prediction results on *imSitu* test set. **Bold** and Underlined indicate best and second best performances

where *agent* and *place* predictions required for verb question, were made without including which verb it aims at. (ii) iterative model runs twice in which on the second iteration *agent* and *place* was queried given the verb prediction of first iteration. Second, we evaluate role label predictions under two types of questions. (i) general question, (ii) context aware question generated with predicted labels. Next, we consider creating verb and role questions based on ground truth context, so to evaluate the impact of incorrect questions on model performance. We call this the *Reference Model*.

Table 5.4 shows performance of our transferred model based on above criteria on the development set assessing different aspects. Performance of verb prediction increases when context was predicted given a verb showing the importance of semantic roles towards verbs. Context aware questions surpass general question performance in role label prediction emphasizing the importance of incorporating role interdependency. On the other hand, the very high performance of our *Reference Model* shows the huge impact of question accuracy has on VQA model performance.

We also compare our proposed model with the existing work in both development set and test set. Overall, the performance of our transfer learning approach is competitive to the state-of-the-art. For "verb" prediction performance, our model is very competitive. However, we can observe that the normal image classification based verb predictors have performed better than our VQA based verb predictor. This implies that our transfer learning source VQA model was not very capable of verb prediction. This is expected as source dataset does not have huge variety of verbs as SR dataset. Our semantic role prediction is very impressive as we achieve highest performance surpassing all existing work. Even with low performing verb predictor, we can observe our transfer learning model perform really well, highlighting the importance of advanced visual reasoning for SRP, proving that VQA is a very suitable source task for SR. We achieve highest results for "value-all" criterion in all sections. But degraded performance compared to using ground-truth labels (*Reference Model*) can be observed due to inaccurate context in the questions. Further work needs to address this issue.

4.4.10 Discussion

In this approach, we proposed a transfer learning based method to solve SR as a VQA problem. SR imposes several challenges when solved as a VQA problem and lack of human generated reliable questions is the main issue. A template based method is used to generate natural language questions, where predicted object labels from previous timestep is used to create neighbour role aware questions.

One of the main observations is the degraded performance when context aware questions were generated with predicted labels compared to ground truth labels. Unlike the source dataset which has questions composed manually by human, our automatic approach suffers from inaccurate question context as the role prediction model could make mistakes.

Based on our model performance, we can say that solving SR as a VQA problem proves to be a promising direction. However, due to the error-prone procedure of natural language question generation, we are left with a problem of **"is transfer learning with hard prediction based natural language questions the best approach?"**. We will be answering this question by removing transfer learning in our next proposed approach and analyzing its outcome.

Chapter 5

Situation Recognition via Graph and Context Aware Reasoning

We further extend deep graph learning application to address Situation Recognition in this chapter, by proposing a graph learning based context aware reasoning mechanism. This is a novel approach based on attention-guided latent soft-query generation, which does not require natural language question generation like in the Chapter 4 and it does not use transfer learning as well.

5.1 Main Contributions

- We are proposing to address situation recognition via query-based visual reasoning
- We enable inter-dependent query handling for query-based visual reasoning models to enable semantic role dependency. Although we model reasoning semantic roles independently in our VQA approach, SR is a structured prediction task. Existing VQA models are unable to handle dependent queries. Hence, we propose novel methods to share information among multiple related questions in a graph learning manner.
- While all existing work address verb prediction as an independent task, we utilize the same proposed query based reasoning method to provide semantic role information as context for verb prediction.

5.2 Frame Recognition and Backbone Model

We formulate Frame Recognition(FR) as a Visual Question Answering (VQA) problem; Given an image I and query q, we want to find the most relevant information from the image to answer q. We formulate queries for each semantic role of the frame as the joint embedding of current frame's verb name and semantic role name. The model needs to answer all of them to retrieve the final realized frame.

We adopt the Top-Down Attention (TDA) model proposed by Anderson et al. (2018) as our backbone VQA mechanism due to its simplicity and effectiveness as well as its less dependency towards the structure of the query compared to other state-of-the-art VQA models such as BAN (J.-H. Kim, Jun, and B.-T. Zhang, 2018), which relies on

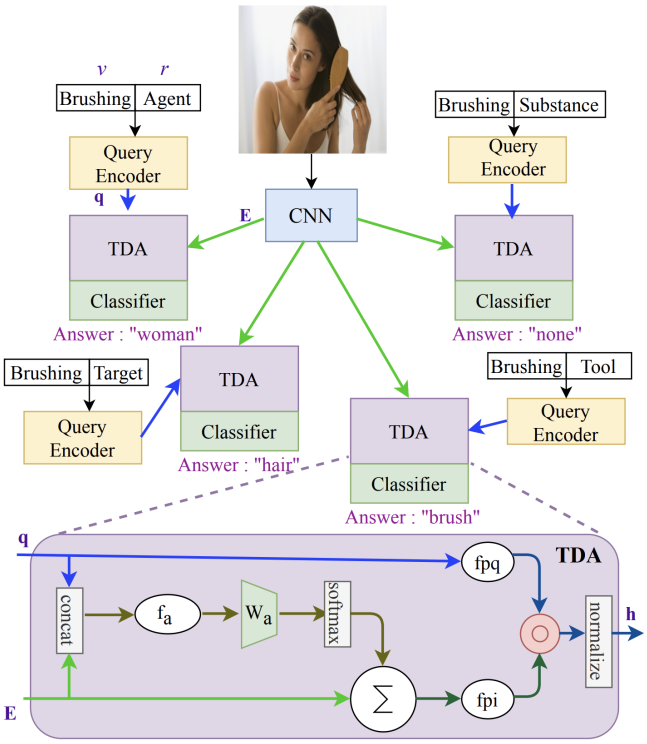


FIGURE 5.1: Top-Down Attention (TDA) model for Frame Recognition in SR. Each role of the verb "Brushing" forms a query, receives the image encoding and goes through the TDA network and the classifier as an independent query to obtain the final noun prediction. Nodes with the same colour indicates the same network which shares parameters.

multiple channel query representations. Hence TDA allows us to use VQA with simple single channel queries which is sufficient for FR.

Given a set of region features of an image and a query embedding, TDA calculates the relevancy score for each image region feature with respect to query embedding. Then all image region features are weighted according to relevancy scores and summed together and fused with the query embedding. This creates the feature representation of the answer to the current query which then be sent through the classifier to obtain the final answer label.

5.2.1 Top-Down Attention for Frame Recognition

Figure 5.1 visualizes how we utilize TDA model for semantic role prediction to obtain the final frame $F_{(I,v)}$. First we consider each semantic role in the current frame as a separate query to our TDA model (handling inter-dependent queries will be discussed next). In the model, first we obtain image region features $\mathbf{E}_I = \{\mathbf{e}_n\}_{n=1}^{N_e}$ by encoding the image $\mathbf{I}$ using a CNN and obtaining the grid features just after the last pooling layer. N_e is the number of regions of the image. We use word embeddings for semantic role r and verb v of the current frame to generate the query encoding $\mathbf{q}$.

$$\mathbf{E}_I = \text{CNN}(\mathbf{I}), \tag{5.1}$$

$$\mathbf{q} = f_q([\mathbf{w}_v, \mathbf{w}_r]), \tag{5.2}$$

where $\mathbf{E}_I \in \mathbb{R}^{N_e \times d\_img}$ and f_q is a non-linear layer. $[\cdot]$ is used to denote the concatenation. $\mathbf{q} \in \mathbb{R}^{d\_q}$ and embedding vectors for verb and role are $\mathbf{w}_v, \mathbf{w}_r \in \mathbb{R}^{d\_wemb}$. These embeddings are randomly initialized and learnt during model training. (Details of all networks (e.g., f_q) are provided in Supplementary).

Then we calculate the image region-level attention weights based on the query encoding, and derive updated image encoding,

$$s_n = \mathbf{w}_a f_a([\mathbf{e}_n, \mathbf{q}])^T, \tag{5.3}$$

$$\alpha_n = \frac{\exp(s_n)}{\sum_{i=1}^{N_e} \exp(s_i)}, \quad \tilde{\mathbf{E}} = \sum_{n=1}^{N_e} \alpha_n \mathbf{e}_n, \tag{5.4}$$

s_n denotes un-normalized region-level attention weights obtained for current query $\mathbf{q}$. α_n denotes the normalized attention weight for region n, and $\tilde{\mathbf{E}}$ is the aggregated image encoding for the query. $\mathbf{w}_a \in \mathbb{R}^{d\_hidden}$ are model parameters and f_a is a non-linear layer.

Then updated image encoding $\tilde{\mathbf{E}}$ and query encoding $\mathbf{q}$ are fused together to obtain the un-normalized hidden representation $\mathbf{h_u} \in \mathbb{R}^{d\_hidden}$,

$$\mathbf{h_u} = f_{pq}(\mathbf{q}) \circ f_{pi}(\tilde{\mathbf{E}}), \tag{5.5}$$

where f_{pq} and f_{pi} non-linear layers are used to project query and image encoding to a different space and $\circ$ denotes element-wise multiplication.

Element-wise multiplication can cause model convergence to an unsatisfactory local minimum (Z. Yu et al., 2018). In order to avoid this Z. Yu et al. (2018) have used the power normalization ($z \leftarrow \text{sign}(z)|z|^{0.5}$) and ℓ_2 normalization ($z \leftarrow z/\|z\|$) layers. Following their approach, we also modified the original TDA model by adding a Dropout (Srivastava et al., 2014) layer and normalization after element-wise multiplication to produce the *normalised hidden representation* $\mathbf{h}$:

$$\mathbf{h} = \ell_2\text{Norm}(\text{PowerNorm}(\text{Dropout}(\mathbf{h_u}))), \tag{5.6}$$

Classifier Finally the normalized hidden representation is sent through a non linear network $f_{classifier}$ followed by a SoftMax function to obtain final probability distributions of each role label prediction.

$$p = \text{SoftMax}(f_{classifier}(\mathbf{h})), \tag{5.7}$$

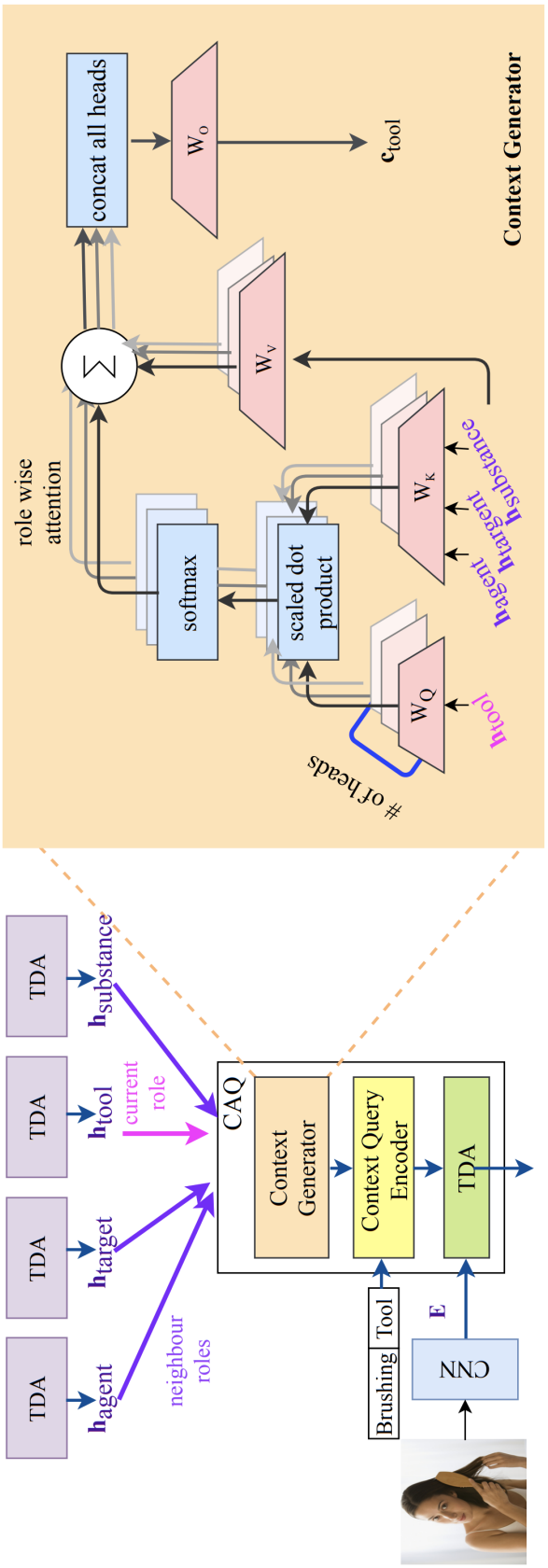


FIGURE 5.2: Context Aware Query (CAQ) based reasoning. In this example, the context is generated for the query of semantic role "tool", using its neighbour roles "agent", "target" and "substance", in the frame of verb "brushing". The context generator is discussed in Sec. 5.3.1. Diagram best viewed in colored version. Inputs to original TDA components (depicted in purple) are same as Figure 5.1.

Learning and Inference We use cross entropy loss to train the model as follows:

$$Loss = \sum_{j=1}^{F_I} \left(-\sum_{i=1}^{|N|} y_{(j,i)} \log(p_i) \right) \tag{5.8}$$

$y_{(j,i)} \in \{0, 1\}$ is the ground truth encoding from the j^{th} realized frame for the noun i, where we can have F_I realized frames for each image. Also note that $p_i \in p$. This Situation Recognition dataset *imSitu* (Yatskar, Zettlemoyer, and Farhadi, 2016) contains three realized frame annotations for each image.

For the complete frame prediction, first we obtain the required role list R_v for the given verb v to be queried in the model to retrieve noun label predictions $\hat{i} = \arg\max_i p_i^r$ for each role $r \in R_v$.

5.3 Handling Inter-dependent Semantic Roles

As we mentioned, the above system answers role queries independently. However, a semantic role not only depends on its action but also on its fellow semantic roles of the current frame, which we refer as its *neighbor roles*. For example in Figure 4.1, for the action "Brushing", *neighbor roles* for semantic role *Tool* are *Agent, Target* and *Substance*.

Existing query-based visual reasoning approaches (Y. Goyal et al., 2017b; Anderson et al., 2018; J.-H. Kim, Jun, and B.-T. Zhang, 2018) aim at answering questions individually. It has not been investigated how to incorporate information from inter-dependent queries to improve single query performance. Hence our backbone TDA model also suffers from this limitation. However, for structured prediction tasks like FR, modelling inter-dependency is important. Therefore, to address the gap between existing query-based visual reasoning approaches and inter-dependency models, we propose three different novel methods: (i) Context Aware Query (CAQ), (ii) Context Aware Image (CAI), and (iii) Context Aware Image Reconstruction (CAIR).

5.3.1 Context Aware Query (CAQ) for Inter-dependent Semantic Role Prediction

CAQ proposes to update the original query encoding with information from neighbour roles as a mechanism to incorporate structure to the existing TDA model. We call the aggregated information retrieved from *neighbour roles* as ***context***. Figure 5.2 depicts the system.

Context Generation We use hidden representations of all the roles of current verb v, $\mathbf{h}^r$, where $r = \{r_1, \ldots, r_{|R_v|}\}$ from TDA model, for the context generation. When generating context for role r, we calculate attention for all other roles in the current frame based on the hidden representation of r to decide how much each neighbour role is important to the current role. Then we weigh hidden representation of each neighbour role and aggregate all of them to generate the context for r.

$$d_k^b = \frac{\mathbf{h}\mathbf{W}_Q^b(\mathbf{h}^{r_k}\mathbf{W}_K^b)^T}{\sqrt{d\_hidden}}, \quad r_k \in R_v \setminus \{r\}, \tag{5.9}$$

$$\alpha_k^b = \frac{\exp(d_k^b)}{\sum_{i;r_i \neq r}^{|R_v|} \exp(d_i^b)}, \quad \mathbf{c}^b = \sum_{r_k \in R_v \setminus \{r\}} \alpha_k^b \mathbf{h}^{r_k}\mathbf{W}_V^b, \tag{5.10}$$

$$\mathbf{c} = [\mathbf{c}^1, \ldots, \mathbf{c}^b, \ldots, \mathbf{c}^B]\mathbf{W}_O, \quad b \in \{1 \ldots B\} \tag{5.11}$$

We use multi-head attention (Vaswani et al., 2017) for this to calculate the context in different representation sub-spaces and join them together to obtain the final context $\mathbf{c}$ (for the current role r). B is the number of heads. $\mathbf{W}_K^b \in \mathbb{R}^{d\_hidden \times d\_head}$, $\mathbf{W}_Q^b \in \mathbb{R}^{d\_hidden \times d\_head}$ and $\mathbf{W}_V^b \in \mathbb{R}^{d\_hidden \times d\_head}$ are model parameters to project hidden representations of key, query and value to a smaller B different subspaces. In our case, key and value are equal and they represent neighbour roles while query is the current role. $d\_head = d\_hidden/B$.

Context Aware Query Generation and Reasoning Now we incorporate the obtained context to query as follows and get the context aware query encoding $\mathbf{q}_c$.

$$\mathbf{q}_c = f_{cq}([\mathbf{c}, \mathbf{w}_v, \mathbf{w}_r]) \tag{5.12}$$

Comparing with Equation 5.2, Equation 5.12 can be seen as adapting the query encoding using context $\mathbf{c}$ which is derived from hidden representations $\mathbf{h}^{r_k}$ of neighbor roles of current role r.

Then we input updated query encoding $\mathbf{q}_c$ and original image encoding $\mathbf{E}_I$ to Equation 5.3. Similar reasoning process to TDA is carried out until Equation 5.6 to obtain the new hidden representation $\mathbf{h}_c$. Finally $\mathbf{h}_c$ will be sent to the classifier for final prediction.

5.3.2 Context aware image (CAI)

In CAI, we add context $\mathbf{c}$ obtained in Equation 5.11 by adding it to the image instead of the question. This allows us to only extract information from image directly related to the context. This approach provides a way to highlight information now seems important at the presence of context prior to the reasoning. We use the following Equation 5.13 to incorporate context information generated in Equation 5.11 to image region encoding:

$$\mathbf{e}_n^c = \sigma([\mathbf{c}, \mathbf{e}_n]\mathbf{W}_{ic}) \circ \mathbf{e}_n, \quad \mathbf{e}_n \in \mathbf{E}_I \tag{5.13}$$

First, we concatenate the context with all the $\mathbf{e}_n, \quad n \in N_e$ regions of the original image $\mathbf{E}_I$ and do a linear transformation using $\mathbf{W}_{ic} \in \mathbb{R}^{(d\_hidden+d\_img) \times d\_img}$. Finally, this is passed through a $sigmoid$ gate to determine how much information of each region needs to be sent for the reasoning step based on the context. Once we obtained the updated image regions, we input it to Equation 5.3 instead of original image regions along with original query encoding $\mathbf{q}$ and continue the TDA mechanism.

5.3.3 Context Aware Image Re-construction (CAIR)

CAIR aims at improving inter-role agreement in the frame by encouraging the model to reconstruct the original image using hidden state $\mathbf{h}$ of all roles. If at least one of the role label representations is incorrect, the image reconstructed by the predicted realized frame differs from the original image. Therefore to construct an image similar to the original, the entire frame needs to be accurate. We use a non-linear layer f_{recon} to generate the reconstructed image from hidden representations $\mathbf{h}$ output from Equation 5.6 for all the roles of the current frame and send the original grid features $\mathbf{E}_I$ of the image through a linear network $f_{flatten\_img}$ to obtain the vector representation of the original image.

$$\hat{\mathbf{E}} = f_{recon}([\mathbf{h}_1, \ldots, \mathbf{h}_{|R_v|}]) \tag{5.14}$$

$$\mathbf{E}_{org} = f_{flatten\_img}(\mathbf{E}_I) \tag{5.15}$$

We add an auxiliary ℓ_2 loss to the original cross entropy loss in Equation 5.8 to encourage the model to make role label predictions which the combined frame prediction can reconstruct the original image as correctly as possible.

$$Loss_{recon} = \left\| \mathbf{E}_{org} - \hat{\mathbf{E}} \right\|_2 \tag{5.16}$$

When using this approach, the final loss for training the model is as following. β is a hyperparameter.

$$L = Loss + \beta Loss_{recon} \tag{5.17}$$

5.4 Reasoning enhanced verb prediction

We remark that our main contribution is *role prediction* (or frame recognition (FR)), which details have been discussed in Section 5.2 and 5.3.

In this section we explain how TDA can be utilized to incorporate context information for verb prediction modelling.

5.4.1 Role label prediction component of the Verb model

TDA model expects a query condition $\mathbf{q}$ as we discussed in Equation 5.3 in order to condition the image and find the relevant answer for the query. For verb prediction, we decided to form our query based on labels of the two most frequent roles in *imSitu* dataset; *Agent* and *Place*.

We decided to use a modified version of TDA based FR model to predict *Agent* and *Place* role labels, which are going to be input for our verb model. The reason we had to modify the original FR model is because, when the query is encoded in Equation 5.2, we use concatenation operation between verb embedding and role name embedding. However when we want to use this FR model to provide us label predictions of *Agent* and *Place* roles to input for verb prediction, FR model should have the capability to process queries which do not have verb embedding. Since concatenation operation cannot

support this requirement, we replaced the original TDA FR model's concatenation in query encoding (Equation 5.2) to an addition operation as follows:

$$\mathbf{q} = f_{role\_q}(\mathbf{w}_v + \mathbf{w}_r) \tag{5.18}$$

During model training, we use Equation 5.18 and we use the following after removing the verb embedding during inference.

$$\mathbf{q} = f_{role\_q}(\mathbf{w}_r) \tag{5.19}$$

First we train this model separately and this pretrained FR model is used to predict *Agent* and *Place* labels during verb model training.

5.4.2 TDA for verb prediction

We use the original TDA model only with slight modifications for verb prediction task. First we modify the query encoding step (Equation 5.2) to our new query condition as follows.

$$\mathbf{q\_verb} = f_{verb\_q}([\mathbf{w}_{agent}, \mathbf{w}_{place}]) \tag{5.20}$$

$[\cdot]$ is used to denote the concatenation. Embedding vectors for *Agent* and *Place* role labels are $\mathbf{w}_{agent}, \mathbf{w}_{place} \in \mathbb{R}^{d\_wemb}$. These embeddings are randomly initialized and learnt during model training. $agent \in \{1, \ldots, |N|\}$ is the *Agent* id and $place \in \{1, \ldots, |N|\}$ is the *Place* id.

Then we use this $\mathbf{q\_verb}$ as our query and continue with the original TDA model from Equation 5.3-5.5 . We observed from our experiments that normalization layers did not help for verb prediction like they did with FR. Therefore we did not execute Equation 5.6 for verb prediction model. As seen in the caption of the Table 5.2, when gold *Agent* and *Place* role labels are used, verb prediction accuracy is very high. But when we replace them with predicted labels, performance drops significantly due to the prediction errors of the FR model. Therefore we understood that completely relying the model on predicted role labels is unwise as they are known to be incorrect sometimes.

As a remedy to this, we decided to use hidden representations of *Agent* and *Place* role labels ($\mathbf{h}_{agent}, \mathbf{h}_{place} \in \mathbf{h}$) generated in Equation 5.6 in this reasoning process as well. We use them as a soft query and fuse them to the original image encoding to provide more contextual information to support verb prediction. We generate this contextual information as follows

$$\mathbf{soft\_query} = \mathbf{h}_{agent} + \mathbf{h}_{place} \tag{5.21}$$

$$\mathbf{E}_{flat} = \text{AvgPool}(\mathbf{E}_I)\mathbf{W}_{flt\_img} \tag{5.22}$$

$$\mathbf{context} = \mathbf{E}_{flat} \circ \mathbf{soft\_query} \tag{5.23}$$

where $\mathbf{W}_{flt\_img} \in \mathbb{R}^{d\_img \times d\_hidden}$. Then we add this with the output from Equation 5.5 to obtain our final hidden representation ($\hat{\mathbf{h}}$) that will be sent to the classifier to get the verb prediction. This is the model we reported results in Table 5.2 row 2.

$$\hat{\mathbf{h}} = \mathbf{h_u} + \mathbf{context} \tag{5.24}$$

$$p_{verb} = \text{SoftMax}(f_{v\_classifier}(\hat{\mathbf{h}})) \tag{5.25}$$

We train this model with cross entropy loss as follows.

$$Loss = -\sum_{i=1}^{|V|} y_i \log(p_{verb}(i)) \tag{5.26}$$

$y_i \in \{0, 1\}$ is the ground truth encoding of the verb i. Also note that $p_{verb}(i) \in p_{verb}$. We get the final verb prediction (v_{pred}) as follows:

$$v_{pred} = \arg\max_i p_{verb}(i) \tag{5.27}$$

5.5 Evaluation

5.5.1 Dataset and Implementation Details

We use *imSitu* (Yatskar, Zettlemoyer, and Farhadi, 2016) dataset for our experiments and we follow the experiment setup and evaluation criteria from Yatskar, Zettlemoyer, and Farhadi (2016). Here we report results for three metrics. *Verb*: verb prediction, *Value*: role-label tuple is considered correct given the verb, if it matches any of the F_I annotations, *Value-all*: when the entire frame is correct, meaning all role-value tuples of the predicted frame matches at least one ground truth annotation. Accuracy % of each of the three metrics is used to compare performance. *imSitu* dataset contains 75K train, 25K development and 25K test set samples which spreads across $V = 504$ verbs, $R = 190$ roles and $N = 2001$ nouns including *UNK* token for unknowns. Each image has $F_I = 3$ realized frames.

We implemented our models using PyTorch (Paszke, Gross, Chintala, et al., 2017) framework. We use VGG-16 (Simonyan and Zisserman, 2015) as our backbone CNN architecture to encode images following all existing work (Yatskar, Zettlemoyer, and Farhadi, 2016; Yatskar, Ordonez, et al., 2017; Mallya and Lazebnik, 2017; R. Li et al., 2017) for SR. We extract grid features of size $7 \times 7 \times 512$ after the final max pooling layer as our regions where $N_e = 49$. Final dimensions of different components of our models as follows: $d\_img = 512$, $d\_q = 1024$, $d\_womb = 300$, $d\_hidden = 1024$, $B = 4$ and $\beta = 10$. Bias terms in all our equations have not been included for the simplicity of notation. Dropout value used in Equation 5.6 is 0.1 and 0.5 is the dropout value used in both FR and verb classifiers. We trained our FR models to predict the most frequent 2000 nouns following (R. Li et al., 2017) as it covers more than 95% of samples. $max\_role\_count = 6$ is the number of maximum roles exist in a frame of *imSitu* dataset. We train the model end-to-end including the CNN where CNN is finetuned with initial

learning rate 5×10^{-5} and the rest of the model with learning rate of 1×10^{-3} using AdaMax (Kingma and Ba, 2015) optimizer and Exponential scheduler. We used mini-batch size of 64 and obtained the best model by early stopping using development set performance. For CAQ and CAI models, we use the pre-trained TDA model to provide the hidden representations for context generation in Equation 9-10 and the rest of the model is trained end to end.

All our non-linear layers ($f_q, f_a, f_{pq}, f_{pi}, f_{cq}$ and f_{recon} and f_{role_q} and f_{verb_q} from this) are using gated hyperbolic tangent activation (Teney, Anderson, et al., 2017) as follows:

$$\tilde{\mathbf{y}} = tanh(\mathbf{x}\text{WeightNorm}(\mathbf{W})) \tag{5.28}$$
$$\mathbf{g} = \sigma(\mathbf{x}\text{WeightNorm}(\mathbf{W}')) \tag{5.29}$$
$$\mathbf{y} = \tilde{\mathbf{y}} \circ \mathbf{g} \tag{5.30}$$

Layer Name	f_input	f_output
f_q	$d\_wemb \times 2$	d_q
f_a	$d\_img + d\_q$	d_hidden
f_{pq}	d_q	d_hidden
f_{pi}	d_img	d_hidden
f_{cq}	$d\_hidden + d\_wemb \times 2$	d_q
f_{recon}	$d\_hidden \times max\_role\_count$	d_hidden
f_{role_q}	d_wemb	d_q
f_{verb_q}	$d\_wemb \times 2$	d_q

TABLE 5.1: Dimensions of all used non-linear layers.

Both $\mathbf{W}$ and $\mathbf{W}'$ have same dimensions $\mathbb{R}^{f\_input \times f\_output}$. Sigmoid function works as a gate to control each element of the input vector $\mathbf{x} \in \mathbb{R}^{f\_input}$ and output $\mathbf{y} \in \mathbb{R}^{f\_output}$. Table 5.1 includes exact dimensions we used for each non-linear layer.

$f_{classifier}$ and $f_{v\_classifier}$ use a Multilayer Perceptron network shown in the box below. For both $f_{classifier}$ and $f_{v\_classifier}$, $\mathbf{W}_{cls\_in} \in \mathbb{R}^{d\_hidden \times (d\_hidden \times 2)}$. For the last layer, the output size differs for each of them as the number of classes are N and V respectively. Therefore $\mathbf{W}^{roles}_{cls\_out} \in \mathbb{R}^{(d\_hidden \times 2) \times N}$ and $\mathbf{W}^{verb}_{cls\_out} \in \mathbb{R}^{(d\_hidden \times 2) \times V}$.

WeightNorm($\mathbf{W}_{cls\_in}$)
$ReLU$
Dropout
WeightNorm($\mathbf{W}_{cls\_out}$)

$f_{flatten\_img}$ contains a linear layer $\mathbf{W} \in \mathbb{R}^{(d\_img \times N_e) \times d\_hidden}$ followed by a BatchNorm (Ioffe and Szegedy, 2015) layer.

5.5.2 Reasoning Enhanced Verb Prediction Performance

Verb Model	Top 1 Verb	Top 5 Verb
VGG Classifier (Mallya and Lazebnik, 2017; R. Li et al., 2017)	36.83	63.48
Predicted Query Model	35.70	62.19
RE-VGG Classifier	**37.96**	**64.99**

TABLE 5.2: Verb only prediction performance in accuracy %. For model using *gold queries*, Top-1: 43.21, Top-5: 68.83.

In this section, we discuss experiments for verb prediction only. Main experiments on FR using our proposed context-aware reasoning will be discussed in the next section.

We analyse the performance of verb prediction when visual reasoning is expanded beyond CNN. Table 5.2 shows performance of multiple approaches we followed. First we report results for the CNN (Lecun and Y. Bengio, 1995) verb classifier, the model which was used by many of the existing work (Yatskar, Zettlemoyer, and Farhadi, 2016; Mallya and Lazebnik, 2017; R. Li et al., 2017) as the baseline. For reasoning enhanced predictions, we use the proposed TDA architecture explained in Section 5.4 and use *Agent* and *Place* role labels as the query in Equation 5.20 to reason the image for verb. We use ground truth *Agent* and *Place* label annotations to form *gold queries* in our reference gold query model. In the Predicted Query Model model, *predicted queries* are formulated using *Agent* and *Place* label predictions from our TDA based pre-trained FR model. Due to FR model's prediction errors, we observe a considerable performance drop in results. Finally, we have our Reasoning Enhanced verb prediction model (RE-VGG) in which we incorporate visual reasoning capabilities of the predicted-role based TDA verb model to the VGG classifier by summing verb wise scores output from the last FC layer of both models to obtain our best verb model.

5.5.3 Context Aware Reasoning for Frame Recognition

In this section we discuss results for the main contribution of this work on how well the context incorporation helps to improve Frame Recognition and results are shown in Table 5.3.

FR Model	Value	Value-all
TDA	72.96	37.60
CAQ	**73.62**	**38.71**
CAI	73.17	37.95
CAIR	73.30	38.17

TABLE 5.3: Frame recognition only performance in accuracy % of proposed context aware methods.

Our TDA model answers queries independently without considering its neighbour roles of the current frame. Next we have performance of our three proposed models

for handling inter-dependent queries. CAQ has outperformed both CAI and CAIR becoming the best approach for inter-dependent query answering. The reason is that it only uses context information as a guidance for the reasoning and if the model feels original image's features are more important to answer query than the context, CAQ allows that too. But in CAI, as the original image is altered using the context, it does not have the opportunity to use original image information at all. CAIR only distantly encourages for role inter-dependency and does not explicitly force like CAQ, hence it cannot perform as good as CAQ.

5.5.4 Comparison with Existing Work

Table 5.4 and Table 5.5 show the performance comparison of our models against existing work. The results of different methods are obtained by either running the authors' provided implementation if they are available, or taking from their papers if the implementations are not available. However, for GGNN based model (R. Li et al., 2017), the authors' provided implementation could not converge. After communicating with the authors, we have re-implemented the model ourselves, and our results are similar to the reported ones by the authors except for "value-all", which we observed lower accuracy than what was reported in R. Li et al. (2017).

We report results for both TDA model and our best inter-dependent query handling CAQ model. Our TDA model which handles role predictions independently has already outperformed all existing work including models which explicitly model role dependencies (Mallya and Lazebnik, 2017; R. Li et al., 2017). This not only proves the effectiveness of sophisticated multi-modal reasoning but also shows how visual reasoning tasks other than VQA can benefit from adopting to query-based reasoning methods. We further improve our performance with CAQ and achieves the new state-of-the-art results for FR. We report verb prediction results for both our CNN based verb classifier *VGG Verb* as well as our reasoning enhanced *RE-VGG* models and we achieve new state-of-the-art results for verb prediction as well.

5.5.5 Qualitative Analysis

Figure 5.3 shows two sample predictions from the *imSitu* development set for verbs "Assembling" and "Igniting" with predicted attention heat maps output from Equation 5.4 for all roles in both TDA and CAQ models. Role dependency matrices were generated by combining un-normalized neighbour role weights generated for all roles from Equation 5.9. For verb "Assembling", TDA model has predicted role *Tool* incorrectly. When CAQ model generates the context for role *Tool*, roles *Component* and *Goal Item* provide the most impact according to the second row of the matrix. We can see the correct predictions of those roles have guided *Tool* in the CAQ model to correct its prediction by adjusting the attention directly to the "Drill". In the second sample also the correct prediction of role *Item* (most important neighbour for *Tool* in verb "Igniting") has guided to correct the attention error of *Tool* happened in TDA via the context information in CAQ. These results show both the effectiveness of our model as well as its interpretability.

We have included further qualitative results in Fig. 5.4,5.5 and 5.6. They contain many additional examples of sample predictions, attention maps and role dependency

	top-1 predicted verb			top-5 predicted verbs			ground truth verbs		mean
	verb	value	value-all	verb	value	value-all	value	value-all	
CNN + CRF (Yatskar, Zettlemoyer, and Farhadi, 2016)	32.25	24.56	14.28	58.64	42.68	22.75	65.90	29.50	36.32
Tensor Composition (Yatskar, Ordonez, et al., 2017)	32.91	25.39	14.87	59.92	44.50	24.04	69.39	33.17	38.02
Above + DataAug (Yatskar, Ordonez, et al., 2017)	34.2	26.56	15.61	62.21	46.72	25.66	70.80	34.82	39.57
RNN (Mallya and Lazebnik, 2017)	36.11	27.74	16.60	63.11	47.09	26.48	70.48	35.56	40.40
VGG Verb, GGNN † (R. Li et al., 2017)	36.83	28.31	16.55	63.48	47.27	25.77	69.63	33.58	40.18
VGG Verb, TDA (Ours)	36.83	29.01	17.52	63.48	48.82	27.91	72.96	37.60	41.77
VGG Verb, CAQ (Ours)	36.83	29.24	18.02	63.48	49.22	28.62	**73.62**	**38.71**	42.22
RE-VGG, CAQ (Ours)	**37.96**	**30.15**	**18.58**	**64.99**	**50.30**	**29.17**	**73.62**	**38.71**	**42.94**

TABLE 5.4: Situation prediction results on *imSitu* development set. † denotes results of our implementation. Best performance in each column is highlighted in **bold** and second best is underlined.

	top-1 predicted verb			top-5 predicted verbs			ground truth verbs		mean
	verb	value	value-all	verb	value	value-all	value	value-all	
CNN + CRF (Yatskar, Zettlemoyer, and Farhadi, 2016)	32.34	24.64	14.19	58.88	42.76	22.55	65.66	28.96	36.25
Tensor Composition (Yatskar, Ordonez, et al., 2017)	32.96	25.32	14.57	60.12	44.64	24.00	69.20	32.97	37.97
Above + DataAug (Yatskar, Ordonez, et al., 2017)	34.12	26.45	15.51	62.59	46.88	25.46	70.44	34.38	39.48
RNN (Mallya and Lazebnik, 2017)	35.90	27.45	16.36	63.08	46.88	26.06	70.27	35.25	40.16
VGG Verb, GGNN † (R. Li et al., 2017)	36.97	28.21	16.27	63.62	47.16	25.32	69.34	33.29	40.02
VGG Verb, TDA (Ours)	36.97	29.04	17.56	63.62	48.81	27.80	72.80	37.46	41.75
VGG Verb, CAQ (Ours)	36.97	29.29	17.98	63.62	49.22	28.45	**73.41**	**38.52**	42.18
RE-VGG, CAQ (Ours)	**38.19**	**30.23**	**18.47**	**65.05**	**50.21**	**28.93**	**73.41**	**38.52**	**42.88**

TABLE 5.5: Situation prediction results on *imSitu* test set. Best performance in each column is highlighted in **bold** and second best is underlined.

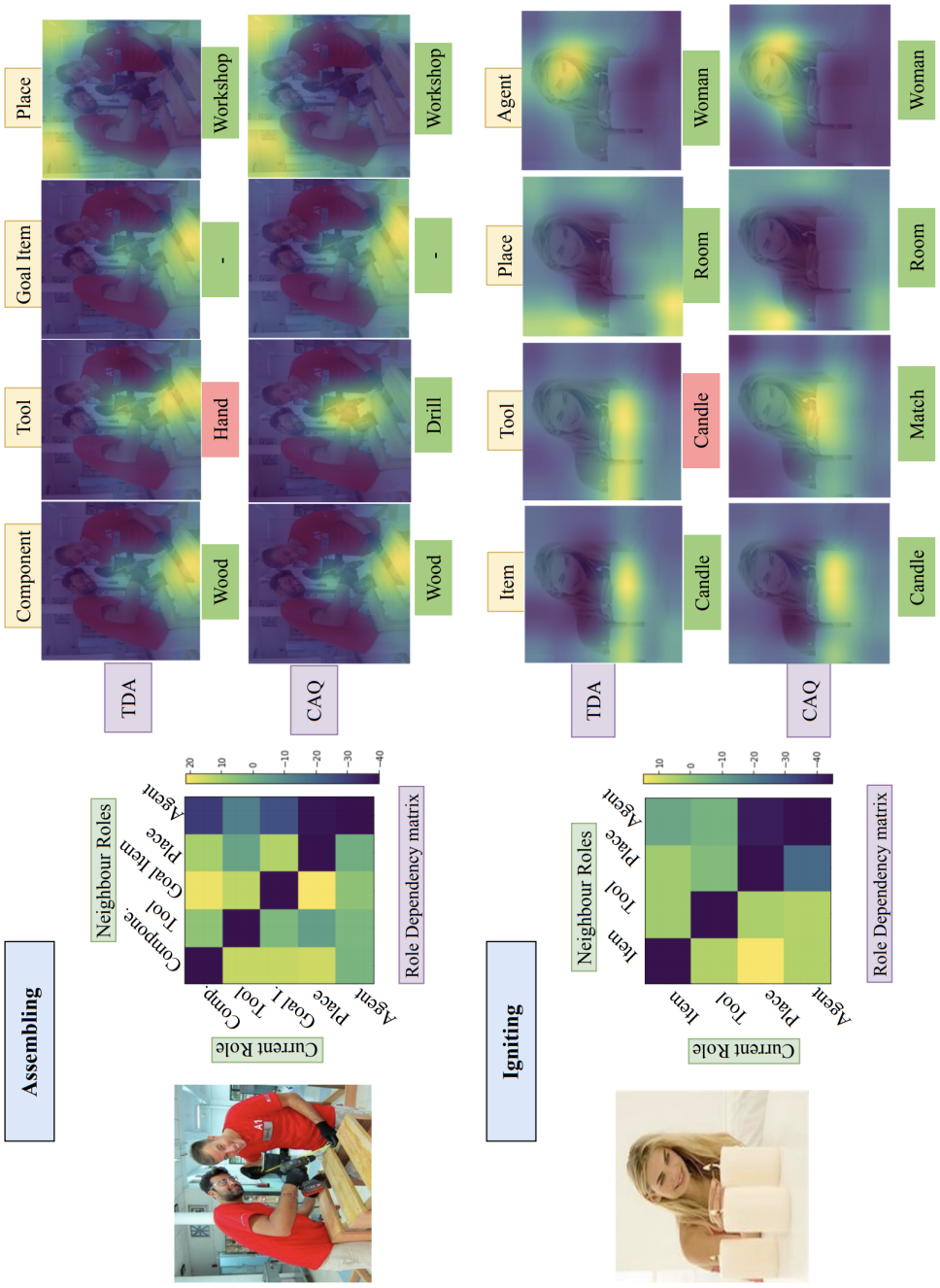


FIGURE 5.3: Visualization of attention maps for multi-modal reasoning and role dependency matrices for two verbs. In both attention maps and matrices, lighter the colour represents higher the value. Diagonal elements of the matrix are indicated in the darkest color to show that own value of current role is not considered as a neighbour role in context generation. Predicted nouns for each role is indicted after each attention map and coloured in green if its correct, red otherwise. **Note the improved attention in "Tool" prediction using context from neighbor roles.** We have removed attention maps for the least important *Agent* role of verb "Assembling" due to the space limitation. Best viewed in colored version.

matrices output from our TDA and CAQ models to showcase how CAQ has been able to improve its attention based reasoning and output accurate predictions compared to TDA, using contextual information. We discuss few more examples in detail here to show how context helped to improve the performance in CAQ.

For verb “Weeding" (sample 1, Fig. 5.4), TDA has predicted the label for role *Tool* wrong. We can observe in the attention map, that TDA has highlighted the entire area around the man including his hands when finding the answer for role *Tool*, hence caused the prediction error. However TDA has correctly attended the image for roles *Place* and *Agent* and predicted them correctly. Next in CAQ, when the context is generated for role *Tool*, we can see from the role dependency matrix that *Agent* has the most impact for *Tool* and *Place* has second most. Thanks to the context provided by *Agent* and *Place*, we can observe that CAQ has been able to provide more focused attention to the “Hoe" and predict accurately.

Another example is verb “Nipping" in Fig. 5.5, sample 1. TDA has not been able to correctly locate which object from the image should be the answer for role *Item*. We can see this error from its attention map. However it has been able to correctly predict the *Agent* by locating the “Dog". From role dependency matrix we can see that, *Agent* provides the most information when generating context for *Item*. Using this context which provides the details that “Dog" is the *Agent*, CAQ has been able to correctly adjust its attention to clearly focus on the “Woman" and hence been able to correct its final prediction.

Final example we are discussing is verb “Fixing" (Fig. 5.5, sample 4). TDA has incorrectly predicted the used *Tool* as “Hand" due to the attention map which has highlighted the entire area of hands and wrench. However CAQ as been able to correct this error using the context generated from neighbour roles' information and focus directly to the *Tool*, “Wrench".

These samples emphasize how our proposed contextualization module contributes to improve inter-dependent query handing. The context generated using neighbour roles has proven to be able to guide the attention mechanism in CAQ to improve its answer localization more accurately than TDA, which only uses the verb_role embedding as guidance to generate attention.

5.5.6 Role inter-dependency differences among verbs

Next in Figure 5.7, we have role dependency matrices for several more verbs along with their sample images. These role dependency matrices are generated combining the unnormalized neighbour role weights generated for all roles in a single frame from Equation 5.9-5.11. Each row of our dependency matrix shows the current role, to which we generate the context using neighbour roles. Each column is for each neighbour role in the current frame. Each cell indicates the value which represents the impact a given neighbour role has on the current role. Diagonal elements have assigned the lowest value to indicate that current role does not consider itself when generating the context.

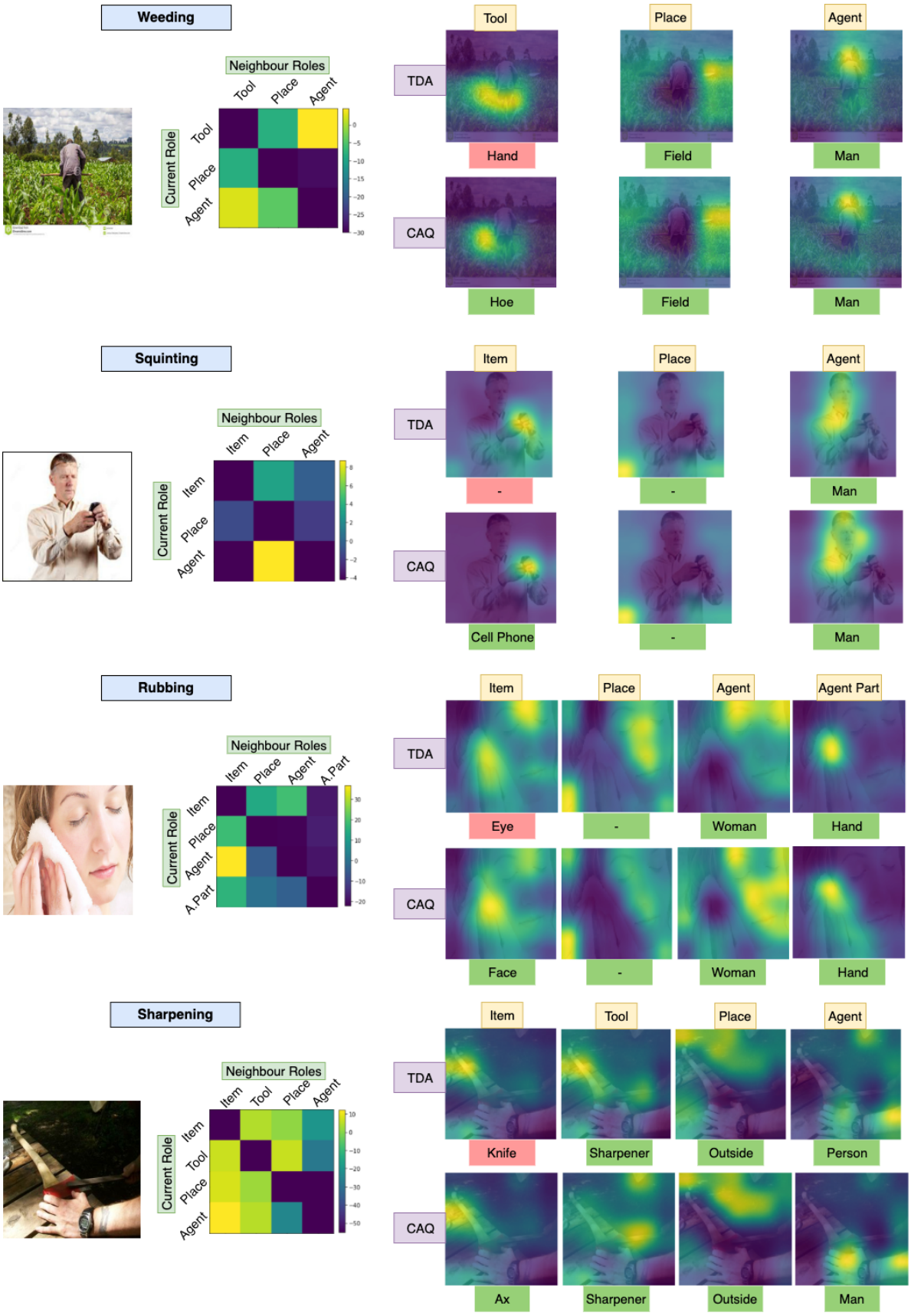


FIGURE 5.4: More qualitative analysis - Part 1

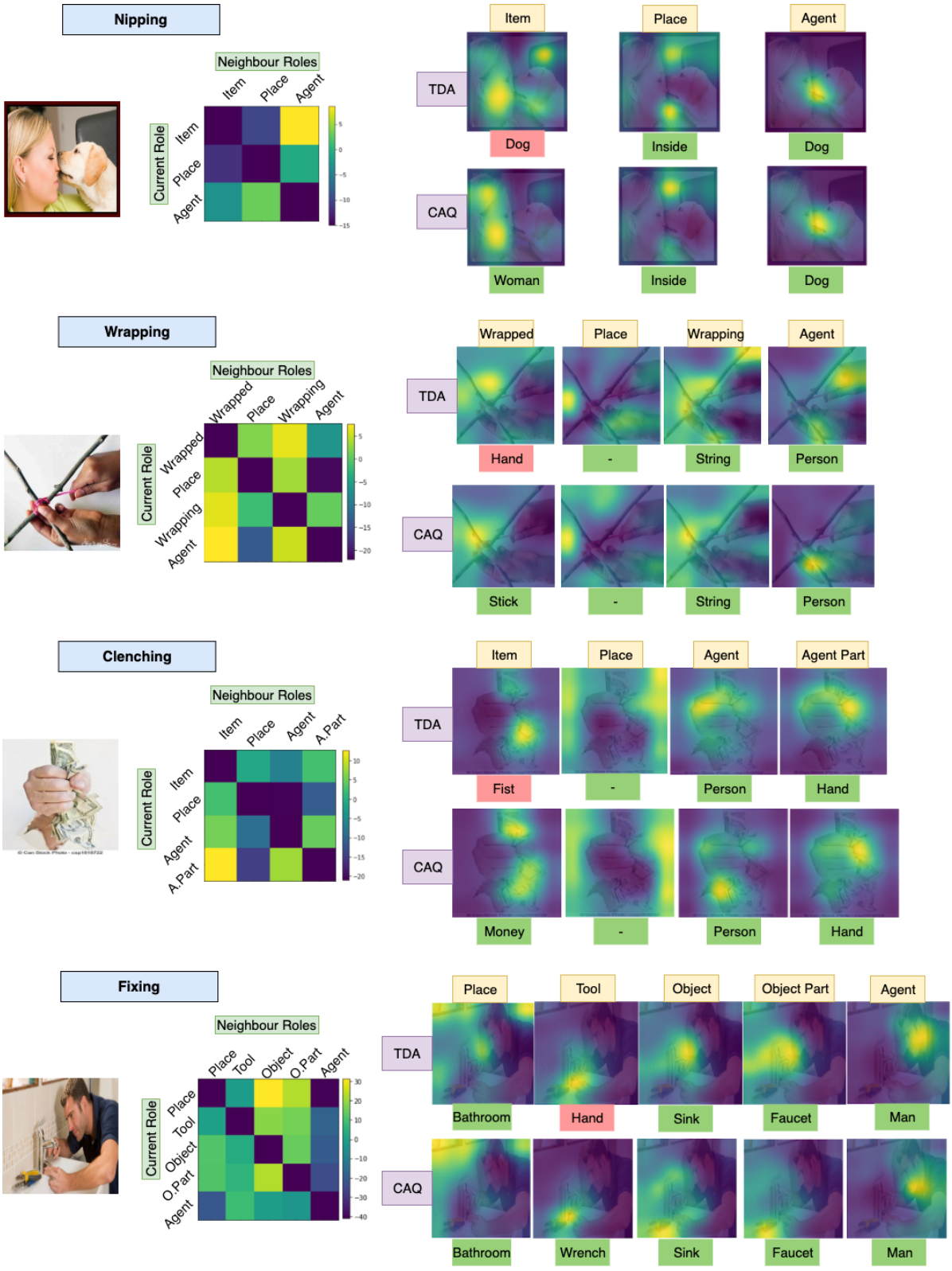


FIGURE 5.5: More qualitative analysis - Part 2

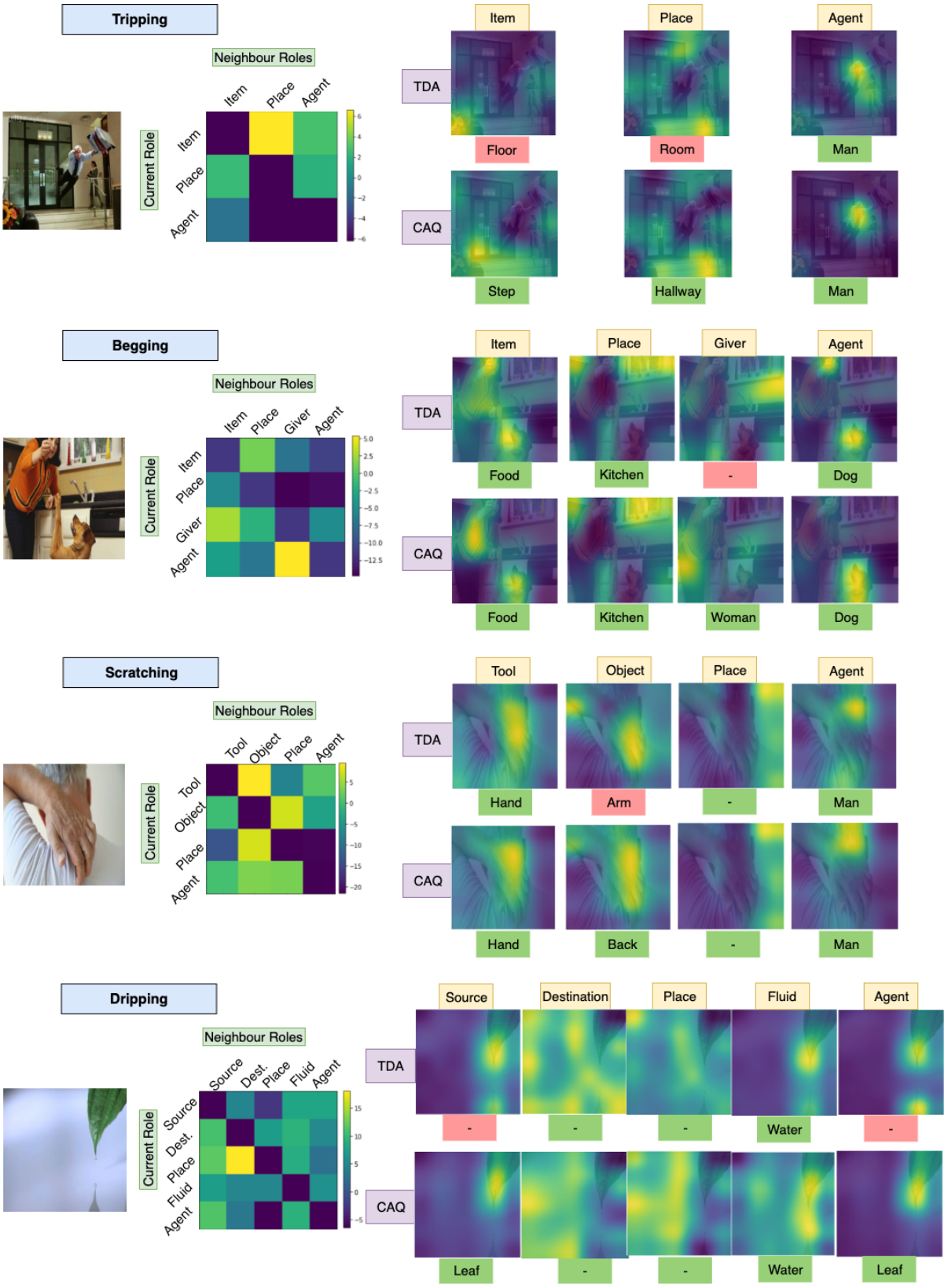


FIGURE 5.6: More qualitative analysis - Part 3

There are multiple subsets of roles that appear in many verbs. For an example the subset of roles {*Agent*, *Place*, *Item*} occur in “Opening", “Tugging" and “Carrying", while “Opening", “Applying", “Tuning" and “Spreading" share another subset of roles together which consists of {*Agent*, *Place*, *Tool*}. But do these roles get the same level of importance in every verb they appear? Do they even maintain the same correlation with their neighbour roles across the verbs they appear? This section is to highlight our observations on these matters according to the generated role dependency matrices by our proposed approach.

We observe based on our learnt role dependency matrices that, eventhough multiple verbs can have same subset of roles, the importance each role gets among its neighbours can vary based on each verb. For an example, eventhough *Item* being the role with the most impact for “Opening" and “Carrying", and *Agent* has the least impact for these verbs, *Agent* has the most impact for verb “Tugging" and *Item*’s impact is lesser. Role inter-dependency also shows a similar characteristic. For an example, *Place* is highly dependent on *Item* for “Opening". But when it comes to “Tugging" and “Carrying", *Place* has a relatively lesser dependency on *Item*.

FIGURE 5.7: **Role Dependency Matrices** of more verbs with sample images which show different senses of verbs. Role list shows the order of roles occur in the matrix whose rows indicate the *Current Role* and each column shows the *Neighbour Roles*. These samples depict how the role with the most impact and role inter-dependencies vary from verb to verb.

5.5.7 Model performance analysis after combining Context Incorporation Methods

CAQ	CAI	CAIR	Value	Value-all
Proposed approach	-	-	73.62	38.71
✓	-	✓	73.62	38.63
-	✓	✓	73.17	37.99
✓	✓	-	72.94	37.38
✓	✓	✓	73.41	38.21

TABLE 5.6: Model performance after combining Context Incorporation Methods. First row contains our final proposed CAQ only model as the reference.

We discuss our analysis on combining proposed context incorporation approaches in this section and results are reported in Table 5.6. Even-though TDA was able to benefit from CAIR according to Table 5.3, CAQ and CAI were unable to achieve improvement from combining with CAIR. This is because the generated context in these models already implicitly facilitates inter-role agreement in order to maintain the stability of predictions across the frame. Hence CAIR is just an ineffective repetition. Performance has degraded when CAQ combined with CAI. The reason for this is that, when both image and query are incorporated with context, there is no room left for individual reasoning to incorporate important information from the original image which might be particularly important for the current role. This result shows an important message on how important it is to allow models some space for independent reasoning as well without completely relying on role inter-dependency, which can cause bias for object co-occurrences in training set. However, this particular issue has been solved for a certain extend after adding CAIR to this model. This is because the $Loss_{recon}$ in Equation 5.16 pushes all predicted objects in the frame to generate an image representation closer to the original image regulating the model from biasing to training set object co-occurrences.

5.5.8 Impact of normalization layer

In our proposed models we use normalization indicated in Equation 5.6. We use normalization to reduce the magnitude of values output by element-wise multiplication operation mentioned in Equation 5.5. Element-wise multiplication can cause the magnitude of outputs vary drastically and this might cause the model to converge to local minimum (Z. Yu et al., 2018). Z. Yu et al. (2018) have used normalization to address that and in this section we empirically evaluate its impact on role and verb predictions.

From Table 5.7 we can observe that both TDA and CAQ FR models have achieved a 1% improvement when normalization layer is used. However for the verb model, the performance have reduced slightly with normalization. The reason for this could be, there are multiple queries reasoned against a single image encoding (all roles of the current frame) in the FR model. This can cause the magnitude of each of the neurons of output vector $\mathbf{h_u}$ to vary quite a lot for queries of the same image. Normalization has

TDA role model		
Feature	Value	Value-all
With normalization	72.96	37.60
Without normalization	72.47	36.85
CAQ role model		
Feature	Value	Value-all
With normalization	73.62	38.71
Without normalization	73.19	37.93
TDA verb model		
Feature	Top 1 Verb	Top 5 Verb
With normalization	34.29	61.92
Without normalization	34.83	61.87

TABLE 5.7: Impact of normalization on role and verb models.

contributed to reduce this variation for some extend. On the other hand, verb model only has one query per image and our queries are very simple compared to natural language sentences. Hence when the normalization is added, it seems to have caused the verb model to underfit a bit and lose its performance.

5.5.9 Impact of context information on TDA verb model

Table 5.8 contains results on the improvement we obtained by adding soft-query based context information to TDA verb model. Model *TDA verb with context* is our final verb model which we report results in Table 5.2 row 3. The difference it has with *TDA verb* model is that in *TDA verb*, we do not incorporate soft query based context for the reasoning process. We do not execute Equation 5.21 - 5.24 in *TDA verb* model and directly send $\mathbf{h_u}$ (output from Equation 5.5) to Equation 5.25 for verb predictions.

Model	Top 1 Verb	Top 5 Verb
TDA verb	34.83	61.87
TDA verb with context	35.70	62.19

TABLE 5.8: Performance comparison of soft-query based context incorporation to verb model.

The reason for this performance improvement is that when hidden representations of *Agent* and *Place* are used, it contains information about multiple potential role labels. Therefore even the final role label prediction was wrong causing our query $\mathbf{q\_verb}$ to be misleading, these hidden representations can contribute to correct it by incorporating secondary information which can provide clues on correct labels.

5.5.10 Performance of CAQ without attention

Table 5.9 compares the impact of attention based context generation on CAQ against TDA and CAQ which context generated without using attention (we call it CAQ without attention). CAQ without attention model does not execute Equation 5.9-5.11. It just sums up hidden representations of all neighbour roles together. CAQ without attention can improve TDA, which does not use any context adaptation. But it cannot surpass final CAQ (which uses attention) as the impact from each neighbour role to the current role differs from role to role. This can be qualitatively observed in the role dependency matrices in Figure 5.7.

Model	Value	Value-all
TDA	72.96	37.60
CAQ	73.62	38.71
CAQ without attention	73.54	38.32

TABLE 5.9: Performance comparison of CAQ for role prediction with and without attention against TDA.

5.5.11 Computational Efficiency

We compared the computational efficiency (Table 5.10) of proposed TDA and CAQ against Gated-GNN based SR model (GGNN) (R. Li et al., 2017). GGNN has the highest parameter count as it uses penultimate layer output from VGG-16 for image encoding while we use grid region features after last max-pooling layer. Although the non-CNN parameter count of GGNN is low, since GGNN is an iterative method, its computation time is high. TDA converged faster than role inter-dependency modeling approaches (GGNN and CAQ). However average running time of all models are of few seconds difference in our cluster of 1 GeForce GTX TITAN X and 1 GeForce GTX 1080 Ti.

Model	No of Total Trainable parameters	No of non-CNN parameters	Avg Training Time	Avg Evaluation Time
GGNN	148574225	10109905	15.72h	114.71s
TDA	28660369	13937233	9.87h	116.98s
CAQ	34955921	20232785	15.46h	120.18s

TABLE 5.10: Model efficiency comparison. Total trainable parameters include CNN and non-CNN parameters. CNN is image encoder, trained end-to-end with the rest of the models. “non-CNN" parameters : GNN - Parameters required for Gated-GNN, TDA - parameters used in Eq.5.2-5.7, CAQ - parameters required for all components in Section 5.3.1.

5.5.12 Comparison of CAQ to GNN with attention

Part of CAQ (Eq. 5.9-5.11) has some similarity with GNN with attention (GNN-A): both techniques try to aggregate hidden representations of neighbour nodes (indicated as *context*) to be used for updating the current node representation. However, the entire CAQ differs from GNN-A significantly; in particular, the mechanism to update the current node (current semantic role) is very different.

GNN heavily relies on inter-node agreement for final node classification as it only uses the *context* for updating nodes. If a node displays a deviation from the normal pattern (Ex: for “Brushing", in majority of samples where a person with a toothbrush, target is “teeth". But for a few, the target is “finger nails"), GNN tends to suppress it by updating the original deviated node representation using its neighbourhood. The drawback of this updating mechanism is that the model tends to get highly biased to training set object co-occurrences. In contrast, CAQ uses the *context* only to update the query in its query based reasoning approach (Eq.5.12), avoiding directly updating node representation ($\mathbf{h}$ from Eq.5.6). Since the updated query has both the original question and context, we implicitly enable the model to decide which part of the query to focus when attending the image in Eq.5.3-5.4. Therefore CAQ has the ability to decide between independent query reasoning and inter-node agreement to mitigate the drawback in GNN.

5.5.13 Error Analysis

We discuss about the main reasons which caused our FR models to make wrong predictions in this section. We consider errors made by CAQ while TDA has the correct prediction, as well as errors that both TDA and CAQ have made which caused both of them to fail in particular samples. When we call a model has failed in a sample, we mean it has been unable to predict the entire frame (measured by *Value-all* criterion) correctly.

We observed that most errors have happened because of the variety of labels *imSitu* dataset has for visually similar objects. Wrong predictions caused by object classification errors are comparatively lesser. Figure 5.8 shows examples on this. Top row consists of examples where TDA has predicted correctly, but CAQ has made some errors. We can see other than the *Agent* prediction error of verb “Arranging", all others have very similar predictions to the correct labels. However since the ground truth annotations do not have these labels included, they have marked as wrong. Same reason have caused in the bottom row also where both TDA and CAQ have failed to predict correctly. For verb “Pulling", although models have misclassified “carriage" for a “bicycle" as the pulled *Item*, the *Agent* label predictions are very reasonable. But the ground truth only contains “cyclist" and “woman", hence our predictions are marked wrong. However for the verb “Rocking", both models have not been able to clearly identify the doll in the crib. When it comes to verb “Selling", both models have not been able to deduce the *Item* should be “Milk" based on the look of the container. This is because the dataset does not have enough samples to support this information. For the verb “Pedaling", both our predictions are very relevant. But as they differ from the ground

Clinging		
Role	TDA	CAQ
Clungto	Tree	Branch
Place	Outdoors	Outdoors
Agent	Bird	Bird

Educating		
Role	TDA	CAQ
Place	Outdoors	Outdoors
Teacher	Woman	Woman
Student	Girl	Child
Subject	-	-

Crushing		
Role	TDA	CAQ
Place	Pit	Pit
Tool	Crusher	Power Shovel
Item	Rock	Rock
Agent	Power Shovel	Power Shovel

Arranging		
Role	TDA	CAQ
Item	Flower	Flower
Tool	Hand	Hand
Place	Room	-
Agent	Man	Woman

Leaking		
Role	TDA	CAQ
Source	Hose	Pipe
Destination	Land	Land
Substance	Water	Water
Place	Outdoors	Outdoors

Pulling		
Role	TDA	CAQ
Item	Bicycle	Bicycle
Tool	Bicycle	Bicycle
Place	Street	Street
Agent	Bicycle	Man

Turning		
Role	TDA	CAQ
Place	Room	-
Turned Item	Knob	Knob
Agent	Person	Person

Rocking		
Role	TDA	CAQ
Place	Room	Room
Container	Crib	Crib
Rocked	-	-
Agent	Woman	Female Child

Selling		
Role	TDA	CAQ
Buyer	Man	Man
Item	Food	Drink
Place	Street	Street
Seller	Man	Man

Pedaling		
Role	TDA	CAQ
Place	Road	Road
Agent	Man	Man
Vehicle	Bicycle	Bicycle

FIGURE 5.8: Samples where our models made wrong predictions. **Top Row** : CAQ has made errors for samples TDA has correct predictions. **Bottom Row** : Both models have made wrong predictions according to ground truth annotations. Green is used to indicate correct predictions, red otherwise.

truth, again the predictions are indicated as wrong. Even though *imSitu* has three annotations per image, it has not been able to cover all possible correct answers in some cases.

We believe grouping these vast variety of visually similar objects and narrowing down the possible answer space will be helpful in the future. Because it will allow future work to clearly separate out errors caused by models and address them.

5.6 Discussion

In this chapter, we address the task of Situation Recognition as a query-based visual reasoning problem and address SR using soft query based VQA model without using natural langauge questions. We further extend our work by proposing novel mechanisms to enable query-based visual reasoning models to handle inter-dependent queries which is a unique requirement of Situation Recognition. For the best of our knowledge, this is the first attempt in incorporating inter-dependent query handling capabilities to

query-based visual reasoning models. Our methods achieve new state-of-the-art results for Situation Recognition.

5.7 Summary

In Chapter 4 and Chapter 5, we focus about applications of graph deep learning. Instead of simply applying graph deep learning algorithms directly to a problem from other domains, this thesis aims at analyzing the challenges of adopting graph deep learning to applications of other domains. Due to the importance of many real world applications and the complex requirements, we selected Vision and Language domain applications. Among them also, situation recognition is a very advanced task due to its requirement of being a structured prediction task and the level of semantic reasoning it requires the model to learn knowledge beyond visual saliency.

We propose to expand multi-modal visual reasoning for structured prediction tasks by incorporating graph learning mechanisms like neighbourhood information propagation. However instead of completely relying on neighbourhood information and giving priority for neighbourhood agreeability, Vision and Language domain also requires the flexibility of deciding whether individual node reasoning is more important than the neighbours. To obtain this balance while applying graph learning techniques, we proposed two novel approaches. Our models obtained state-of-the-art results surpassing models which directly applied graph deep learning models such as GNNs. This proves that, when applying graph deep learning, domain adaptation is also very vital.

Chapter 6

Conclusions

In this thesis, we consider two very important directions of deep learning based approaches for graph structured data. We focus on improving deep learning algorithms for extracting meaningful information from unannotated graphs and applying graph deep learning techniques for multi-modal visual reasoning tasks. In the scope of this thesis, we first analyze how to determine what information available in a graph are most relevant for graph-level tasks and how to extract them in an unsupervised manner. Next, our attention goes on how to apply these graph learning algorithms for Situation Recognition task by addressing their inability in visual semantic reasoning and their overemphasis on co-occurring neighbourhoods.

We introduce a graph-wise common latent factor extraction based approach for unsupervised graph representation learning in Chapter 3. Based on real world graphs, we identified common global factors are used along with node specific factors for graph generation. Therefore, we hypothesized extracting common global factors could be highly beneficial for discriminative graph representations. We propose GCFX principle and deepGCFX model to address feature differentiation and proximity overemphasis limitations of VAE to enable graph-wise common latent factor extraction. With extensive experiments, we demonstrate the effectiveness of deepGCFX and its outstanding performance over existing state-of-the-art with only utilizing current sample for representation learning.

With the understanding of algorithms and how to improve them for naturally graph structured data, next we shift our focus on how to apply those techniques to solve problems in other data modalities. We observed for Vision and Language structured prediction tasks, although it requires graph like neighbourhood information propagation, more essentially it needs advanced multi-modal reasoning capabilities that current graph learning algorithms cannot provide. Hence, we need an aggregated solution which is capable of multi-modal reasoning enhanced neighbourhood information sharing.

We explore the potential of a transfer learning approach from a VQA system to incorporate multi-modal reasoning to Situation Recognition task in Chapter 4. We propose a context-aware question generation mechanism to enable neighbour semantic role information propagation (Cooray, Cheung, and Lu, 2019) using the meta data provided in our target dataset. We predict action and role labels by incorporating verb and inter-role dependency. We conduct experiments and observe that complex questions with context information shared among all elements of the semantic role structure indeed improves the task performance.

Performance of this approach significantly degrades when context aware questions are generated with predicted labels compared to ground truth labels. Unlike the source dataset which has questions composed manually by human, our automatic approach suffers from inaccurate question context as the role prediction model could make mistakes. To address this issue, We further extend our work in Chapter 5 by proposing novel mechanisms to enable query-based visual reasoning models to handle inter-dependent queries, which is a unique requirement of Situation Recognition, in a latent manner. To the best of our knowledge, this is the first attempt in incorporating inter-dependent query handling capabilities to query-based visual reasoning models. Our proposed inter-dependent query based information propagation method (Cooray, Cheung, and Lu, 2020) does not over emphasize on inter-node agreement in a neighbourhood, causing bias towards learning frequent object co-occurrence patterns ignoring rarely occurred but plausible situations like GNNs.

Future Work. Although we addressed the research questions raised in Chapter 1 throughout the course of this thesis, there are other aspects these work could be extended in the future.

- **Handling over smoothing in deep graph neural networks.** Over smoothing occurs in graph neural networks due to its neighbourhood information propagation strategy. In each layer of a GNN, it updates each node representation with the neighbour node information. This tends to make the representations of nearby nodes similar to each other. With the increase of layers in a GNN, the number of nodes with similar representations increases. Therefore at one point, it is possible that all the nodes in the graph to have the same representation ignoring the input features. This is called over smoothing (Q. Li, Z. Han, and X.-M. Wu, 2018). Although several approaches such as data augmentation (Rong et al., 2020) and normalization (L. Zhao and Akoglu, 2020) have been proposed, they are far from solving this problem. We believe that our proposed deepGCFX model can be extended to address this issue from a different perspective. Due to its capability of extracting graph-wise common features while preserving node-specific features to maintain the relation with original input (achieved via the autoencoder setup), deepGCFX always ensures that learnt node representations do not fall victim to over-smoothing. By extending this to apply layer-wise common and non-common filtering (proposed ACCUM (Sec 3.4.2 in Chapter 3)), we can further reduce duplicate information propagation among nodes.

- **Application of inter-dependent query-based reasoning for other tasks.** Apart from Situation Recognition, there are other multi-modal reasoning tasks that could benefit from inter-dependent query-based reasoning. For an example, most of the existing VQA systems (Anderson et al., 2018; J.-H. Kim, Jun, and B.-T. Zhang, 2018; Zellers et al., 2019; X. Zhang, F. Zhang, and C. Xu, 2021) are only aimed at answering each question independently (our motivation for proposing inter-dependent query reasoning). However, if we consider real life scenarios such as a navigation guidance system, users always ask multiple related questions about a single scene. Therefore, applying inter-dependent query-based reasoning for a use case like this would make VQA systems more user-friendly as the

inter-question answer agreeability would be high, avoiding the system from either providing same answer for different questions or providing contradicting answers for related questions.

Bibliography

Acosta-Mendoza, Niusvel et al. (2020). "Mining clique frequent approximate subgraphs from multi-graph collections". In: *Appl. Intell.* 50.3, pp. 878–892. DOI: 10.1007/s10489-019-01564-8. URL: https://doi.org/10.1007/s10489-019-01564-8.

Adhikari, Bijaya et al. (2018). "Sub2Vec: Feature Learning for Subgraphs". In: *Advances in Knowledge Discovery and Data Mining - 22nd Pacific-Asia Conference, PAKDD 2018, Melbourne, VIC, Australia, June 3-6, 2018, Proceedings, Part II*, pp. 170–182. DOI: 10.1007/978-3-319-93037-4_14. URL: https://doi.org/10.1007/978-3-319-93037-4%5C_14.

Anderson, Peter et al. (2018). "Bottom-up and top-down attention for image captioning and visual question answering". In: *CVPR*. Vol. 3. 5, p. 6.

Baek, Jinheon, Minki Kang, and Sung Ju Hwang (2021). "Accurate Learning of Graph Representations with Graph Multiset Pooling". In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. URL: https://openreview.net/forum?id=JHcqXGaqiGn.

Bai, Yunsheng et al. (2019). "Unsupervised Inductive Graph-Level Representation Learning via Graph-Graph Proximity". In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pp. 1988–1994. DOI: 10.24963/ijcai.2019/275. URL: https://doi.org/10.24963/ijcai.2019/275.

Baker, Collin F., Charles J. Fillmore, and John B. Lowe (1998). "The Berkeley FrameNet Project". In: *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, COLING-ACL '98, August 10-14, 1998, Université de Montréal, Montréal, Quebec, Canada. Proceedings of the Conference*. Pp. 86–90. URL: http://aclweb.org/anthology/P/P98/P98-1013.pdf.

Baldi, P. and K. Hornik (1989). "Neural networks and principal component analysis: Learning from examples without local minima". In: *Neural Networks* 2, pp. 53–58.

Bell, Michael GH and Yasunori Iida (1997). *Transportation network analysis*.

Bhatia, K. et al. (2016). *The extreme classification repository: Multi-label datasets and code*. URL: http://manikvarma.org/downloads/XC/XMLRepository.html.

Bird, James et al. (2021). "Advances in deep space exploration via simulators deep learning". In: *New Astronomy* 84, p. 101517. ISSN: 1384-1076. DOI: https://doi.org/10.1016/j.newast.2020.101517. URL: https://www.sciencedirect.com/science/article/pii/S1384107620302219.

Bochkovskiy, Alexey, Chien-Yao Wang, and Hong-Yuan Mark Liao (2020). "YOLOv4: Optimal Speed and Accuracy of Object Detection". In: *CoRR* abs/2004.10934. arXiv: 2004.10934. URL: https://arxiv.org/abs/2004.10934.

Bodra, Jay et al. (2018). "Query Processing on Large Graphs: Scalability Through Partitioning". In: *Big Data Analytics and Knowledge Discovery - 20th International Conference, DaWaK 2018, Regensburg, Germany, September 3-6, 2018, Proceedings*. Ed. by Carlos Ordonez and Ladjel Bellatreche. Vol. 11031. Lecture Notes in Computer Science. Springer, pp. 271–288. DOI: 10.1007/978-3-319-98539-8_21. URL: https://doi.org/10.1007/978-3-319-98539-8%5C_21.

Borgwardt, K. M. and H. P. Kriegel (2005). "Shortest-path kernels on graphs". In: *Fifth IEEE International Conference on Data Mining (ICDM'05)*, 8 pp.-.

Bronstein, Michael M et al. (2021a). "Geometric deep learning: Grids, groups, graphs, geodesics, and gauges". In: *arXiv preprint arXiv:2104.13478*.

Bronstein, Michael M. et al. (2021b). "Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges". In: *CoRR* abs/2104.13478. arXiv: 2104.13478. URL: https://arxiv.org/abs/2104.13478.

Brown, Nathan et al. (2019). "GuacaMol: Benchmarking Models for de Novo Molecular Design". In: *Journal of Chemical Information and Modeling* 59.3, pp. 1096–1108. DOI: 10.1021/acs.jcim.8b00839. eprint: https://doi.org/10.1021/acs.jcim.8b00839. URL: https://doi.org/10.1021/acs.jcim.8b00839.

Cadène, Rémi et al. (2019). "MUREL: Multimodal Relational Reasoning for Visual Question Answering". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 1989–1998. URL: http://openaccess.thecvf.com/content%5C_CVPR%5C_2019/html/Cadene%5C_MUREL%5C_Multimodal%5C_Relational%5C_Reasoning%5C_for%5C_Visual%5C_Question%5C_Answering%5C_CVPR%5C_2019%5C_paper.html.

Cai, Deng and Wai Lam (2020). "AMR Parsing via Graph-Sequence Iterative Inference". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. Ed. by Dan Jurafsky et al. Association for Computational Linguistics, pp. 1290–1301. DOI: 10.18653/v1/2020.acl-main.119. URL: https://doi.org/10.18653/v1/2020.acl-main.119.

Carreira, Joao et al. (June 2016). "Human Pose Estimation With Iterative Error Feedback". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Chang, Chih-Chung and Chih-Jen Lin (2011). "LIBSVM: A library for support vector machines". In: *ACM Trans. Intell. Syst. Technol.* 2.3, 27:1–27:27. DOI: 10.1145/1961189.1961199. URL: https://doi.org/10.1145/1961189.1961199.

Chanpuriya, Sudhanshu, Cameron Musco, Konstantinos Sotiropoulos, and Charalampos Tsourakakis (18–24 Jul 2021). "DeepWalking Backwards: From Embeddings Back to Graphs". In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 1473–1483. URL: http://proceedings.mlr.press/v139/chanpuriya21a.html.

Chanpuriya, Sudhanshu, Cameron Musco, Konstantinos Sotiropoulos, and Charalampos E. Tsourakakis (2020). "Node Embeddings and Exact Low-Rank Representations of Complex Networks". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle et al. URL: https://proceedings.

neurips.cc/paper/2020/hash/99503bdd3c5a4c4671ada72d6fd81433-Abstract.html.

Chen, Dexiong, Laurent Jacob, and Julien Mairal (2020). "Convolutional Kernel Networks for Graph-Structured Data". In: *CoRR* abs/2003.05189. arXiv: 2003.05189. URL: https://arxiv.org/abs/2003.05189.

Chen, Hongming et al. (2018). "The rise of deep learning in drug discovery". In: *Drug Discovery Today* 23.6, pp. 1241–1250. ISSN: 1359-6446. DOI: https://doi.org/10.1016/j.drudis.2018.01.039. URL: https://www.sciencedirect.com/science/article/pii/S1359644617303598.

Cooray, Thilini, Ngai-Man Cheung, and Wei Lu (2019). "Sometime you just need to ask: Situation Recognition via VQA". In: *Workshop on Language and Vision at CVPR*.

— (June 2020). "Attention-Based Context Aware Reasoning for Situation Recognition". In: *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Das, Abhishek, Samyak Datta, et al. (2018). "Embodied Question Answering". In: *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 1–10. DOI: 10.1109/CVPR.2018.00008. URL: http://openaccess.thecvf.com/content%5C_cvpr%5C_2018/html/Das%5C_Embodied%5C_Question%5C_Answering%5C_CVPR%5C_2018%5C_paper.html.

Das, Abhishek, Satwik Kottur, et al. (2017). "Visual Dialog". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Dauphin, Yann N. et al. (2017). "Language Modeling with Gated Convolutional Networks". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, pp. 933–941. URL: http://proceedings.mlr.press/v70/dauphin17a.html.

Defferrard, Michaël, Xavier Bresson, and Pierre Vandergheynst (2016). "Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*. Ed. by Daniel D. Lee et al., pp. 3837–3845. URL: https://proceedings.neurips.cc/paper/2016/hash/04df4d434d481c5bb723be1b6df1ee65-Abstract.html.

Deng, Jia et al. (2009). "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*. IEEE Computer Society, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848. URL: https://doi.org/10.1109/CVPR.2009.5206848.

Desjardins, Guillaume, Aaron C. Courville, and Yoshua Bengio (2012). "Disentangling Factors of Variation via Generative Entangling". In: *CoRR* abs / 1210.5474. arXiv: 1210.5474. URL: http://arxiv.org/abs/1210.5474.

Dıaz, Josep et al. (1999). "Linear Orderings of Random Geometric Graphs". In: *Graph-Theoretic Concepts in Computer Science, 25th International Workshop, WG '99, Ascona, Switzerland, June 17-19, 1999, Proceedings*. Ed. by Peter Widmayer, Gabriele Neyer, and Stephan J. Eidenbenz. Vol. 1665. Lecture Notes in Computer Science. Springer, pp. 291–302. DOI: 10.1007/3-540-46784-X_28. URL: https://doi.org/10.1007/3-540-46784-X%5C_28.

Donnat, Claire et al. (2018). "Learning Structural Node Embeddings via Diffusion Wavelets". In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pp. 1320–1329. DOI: 10.1145/3219819.3220025. URL: https://doi.org/10.1145/3219819.3220025.

Duan, Sunny et al. (2020). "Unsupervised Model Selection for Variational Disentangled Representation Learning". In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. URL: https://openreview.net/forum?id=SyxL2TNtvr.

Duvenaud, David K et al. (2015). "Convolutional Networks on Graphs for Learning Molecular Fingerprints". In: *Advances in Neural Information Processing Systems 28*. Ed. by C. Cortes et al. Curran Associates, Inc., pp. 2224–2232. URL: http://papers.nips.cc/paper/5954-convolutional-networks-on-graphs-for-learning-molecular-fingerprints.pdf.

Erdos, Paul and Alfred Renyi (1960). "On the evolution of random graphs". In: *Publ. Math. Inst. Hungary. Acad. Sci.* 5, pp. 17–61.

Fabbri, Alexander R. et al. (2021). "ConvoSumm: Conversation Summarization Benchmark and Improved Abstractive Summarization with Argument Mining". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*. Ed. by Chengqing Zong et al. Association for Computational Linguistics, pp. 6866–6880. DOI: 10.18653/v1/2021.acl-long.535. URL: https://doi.org/10.18653/v1/2021.acl-long.535.

Fey, Matthias and Jan Eric Lenssen (2019). "Fast Graph Representation Learning with PyTorch Geometric". In: *CoRR* abs/1903.02428. arXiv: 1903.02428. URL: http://arxiv.org/abs/1903.02428.

Filippidou, Ioanna and Yannis Kotidis (2015). "Online and on-demand partitioning of streaming graphs". In: *2015 IEEE International Conference on Big Data, Big Data 2015, Santa Clara, CA, USA, October 29 - November 1, 2015*. IEEE Computer Society, pp. 4–13. DOI: 10.1109/BigData.2015.7363735. URL: https://doi.org/10.1109/BigData.2015.7363735.

Flanigan, Jeffrey et al. (2014). "A Discriminative Graph-Based Parser for the Abstract Meaning Representation". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*. The Association for Computer Linguistics, pp. 1426–1436. DOI: 10.3115/v1/p14-1134. URL: https://doi.org/10.3115/v1/p14-1134.

Fornito, Alex, Andrew Zalesky, and Edward Bullmore (2016). *Fundamentals of brain network analysis*. Academic Press.

Ganie, Hilal A. and Bilal A. Chat (2019). "Bounds for the energy of weighted graphs". In: *Discret. Appl. Math.* 268, pp. 91–101. DOI: 10.1016/j.dam.2019.04.030. URL: https://doi.org/10.1016/j.dam.2019.04.030.

Gärtner, Thomas, Peter A. Flach, and Stefan Wrobel (2003). "On Graph Kernels: Hardness Results and Efficient Alternatives". In: *Computational Learning Theory and Kernel Machines, 16th Annual Conference on Computational Learning Theory and 7th Kernel Workshop, COLT/Kernel 2003, Washington, DC, USA, August 24-27, 2003, Proceedings*, pp. 129–143. DOI: 10.1007/978-3-540-45167-9_11. URL: https://doi.org/10.1007/978-3-540-45167-9%5C_11.

Gilmer, Justin et al. (2017). "Neural Message Passing for Quantum Chemistry". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pp. 1263–1272. URL: http://proceedings.mlr.press/v70/gilmer17a.html.

Gómez-Bombarelli, Rafael et al. (2018). "Automatic chemical design using a data-driven continuous representation of molecules". In: *ACS central science* 4.2, pp. 268–276.

Goyal, Yash et al. (2017a). "Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, pp. 6325–6334. DOI: 10.1109/CVPR.2017.670. URL: https://doi.org/10.1109/CVPR.2017.670.

— (2017b). "Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Graves, Alex et al. (Jan. 2006). "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural 'networks". In: vol. 2006, pp. 369–376. DOI: 10.1145/1143844.1143891.

Grill, Jean-Bastien et al. (2020). "Bootstrap Your Own Latent - A New Approach to Self-Supervised Learning". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle et al. URL: https://proceedings.neurips.cc/paper/2020/hash/f3ada80d5c4ee70142b17b8192b2958e-Abstract.html.

Grover, Aditya and Jure Leskovec (2016). "node2vec: Scalable Feature Learning for Networks". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pp. 855–864. DOI: 10.1145/2939672.2939754. URL: https://doi.org/10.1145/2939672.2939754.

Guo, Xiaojie et al. (2020). "Interpretable Deep Graph Generation with Node-edge Co-disentanglement". In: *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pp. 1697–1707. URL: https://dl.acm.org/doi/10.1145/3394486.3403221.

Guo, Yiluan and Ngai-Man Cheung (2018). "Efficient and Deep Person Re-Identification using Multi-Level Similarity". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Gupta, Abhishek et al. (2021). "Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues". In: *Array* 10, p. 100057. ISSN: 2590-0056. DOI: https://doi.org/10.1016/j.array.2021.100057. URL: https://www.sciencedirect.com/science/article/pii/S2590005621000059.

Gupta, Saurabh and Jitendra Malik (2015). "Visual Semantic Role Labeling". In: *CoRR* abs/1505.04474. arXiv: 1505.04474. URL: http://arxiv.org/abs/1505.04474.

Gurari, Danna et al. (2018). "VizWiz Grand Challenge: Answering Visual Questions From Blind People". In: *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. Computer Vision Foundation / IEEE Computer Society, pp. 3608–3617. DOI: 10.1109/CVPR.2018.00380. URL: http://openaccess.thecvf.com/content%5C_cvpr%5C_2018/html/Gurari%5C_VizWiz%5C_Grand%5C_Challenge%5C_CVPR%5C_2018%5C_paper.html.

Hamilton, William L., Zhitao Ying, and Jure Leskovec (2017). "Inductive Representation Learning on Large Graphs". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 1024–1034. URL: http://papers.nips.cc/paper/6703-inductive-representation-learning-on-large-graphs.

Hassani, Kaveh and Amir Hosein Khasahmadi (2020). "Contrastive Multi-View Representation Learning on Graphs". In: *Proceedings of International Conference on Machine Learning*, pp. 3451–3461.

He, Kaiming et al. (2016). "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, pp. 770–778. DOI: 10.1109/CVPR.2016.90. URL: https://doi.org/10.1109/CVPR.2016.90.

Higgins, Irina et al. (2017). "beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. URL: https://openreview.net/forum?id=Sy2fzU9gl.

Hinton, Geoffrey E. and Richard S. Zemel (1993). "Autoencoders, Minimum Description Length and Helmholtz Free Energy". In: *Advances in Neural Information Processing Systems 6, [7th NIPS Conference, Denver, Colorado, USA, 1993]*, pp. 3–10. URL: http://papers.nips.cc/paper/798-autoencoders-minimum-description-length-and-helmholtz-free-energy.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long Short-Term Memory". In: *Neural Computation* 9, pp. 1735–1780.

Hoskins, Jeremy et al. (2018). "Inferring Networks From Random Walk-Based Node Similarities". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2018/file/2f25f6e326adb93c5787175dda209ab6-Paper.pdf.

Hudson, Drew A and Christopher D Manning (2018). "Compositional Attention Networks for Machine Reasoning". In: *International Conference on Learning Representations (ICLR)*.

Ioffe, Sergey and Christian Szegedy (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pp. 448–456. URL: http://proceedings.mlr.press/v37/ioffe15.html.

Ji, Tao, Yuanbin Wu, and Man Lan (2019). "Graph-based Dependency Parsing with Graph Neural Networks". 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*. Ed. by Anna Korhonen, David R. Traum, and Lluıs Màrquez. Association for Computational Linguistics, pp. 2475–2485. DOI: 10.18653/v1/p19-1237. URL: https://doi.org/10.18653/v1/p19-1237.

Jin, Ming et al. (2021). "Multi-Scale Contrastive Siamese Networks for Self-Supervised Graph Representation Learning". In: *CoRR* abs/2105.05682. arXiv: 2105.05682. URL: https://arxiv.org/abs/2105.05682.

Johnson, Justin, Bharath Hariharan, et al. (2017). "CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 1988–1997. DOI: 10.1109/CVPR.2017.215. URL: https://doi.org/10.1109/CVPR.2017.215.

Johnson, Justin, Ranjay Krishna, et al. (2015). "Image retrieval using scene graphs". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3668–3678.

Kim, Jin-Hwa, Jaehyun Jun, and Byoung-Tak Zhang (2018). "Bilinear Attention Networks". In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*. Pp. 1571–1581. URL: http://papers.nips.cc/paper/7429-bilinear-attention-networks.

Kim, Jin-Hwa, Kyoung Woon On, et al. (2017). "Hadamard Product for Low-rank Bilinear Pooling". In: *The 5th International Conference on Learning Representations*.

Kingma, Diederik P. and Jimmy Ba (2015). "Adam: A Method for Stochastic Optimization". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. URL: http://arxiv.org/abs/1412.6980.

Kipf, Thomas N and Max Welling (2016a). "Semi-Supervised Classification with Graph Convolutional Networks". In: *arXiv preprint arXiv:1609.02907*.

— (2016b). "Variational Graph Auto-Encoders". In: *CoRR* abs/1611.07308. arXiv: 1611.07308. URL: http://arxiv.org/abs/1611.07308.

— (2017). "Semi-Supervised Classification with Graph Convolutional Networks". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. URL: https://openreview.net/forum?id=SJU4ayYgl.

Klein, Guillaume et al. (July 2017). "OpenNMT: Open-Source Toolkit for Neural Machine Translation". In: *Proceedings of ACL 2017, System Demonstrations*. Vancouver, Canada: Association for Computational Linguistics, pp. 67–72. URL: https://www.aclweb.org/anthology/P17-4012.

Kriege, Nils M. and Petra Mutzel (2012). "Subgraph Matching Kernels for Attributed Graphs". In: *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. URL: http://icml.cc/2012/papers/542.pdf.

Krishna, Ranjay, Michael Bernstein, and Li Fei-Fei (2019). "Information Maximizing Visual Question Generation". In: *IEEE Conference on Computer Vision and Pattern Recognition*.

Krishna, Ranjay, Yuke Zhu, et al. (2016). "Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations". In: URL: https://arxiv.org/abs/1602.07332.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*. Pp. 1106–1114. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.

Krogan, Nevan et al. (Apr. 2006). "Global landscape of protein complexes in the yeast Saccharomyces cerevisiae". In: *Nature* 440, pp. 637–43. DOI: 10.1038/nature04670.

Kuhn, Christoph and David N Beratan (1996). "Inverse strategies for molecular design". In: *The Journal of Physical Chemistry* 100.25, pp. 10595–10599.

Kulick, Seth, Ann Bies, and Justin Mott (2012). "Using Supertags and Encoded Annotation Principles for Improved Dependency to Phrase Structure Conversion". In: *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 3-8, 2012, Montréal, Canada*. The Association for Computational Linguistics, pp. 305–314. URL: https://aclanthology.org/N12-1031/.

Lecun, Yann and Yoshua Bengio (1995). "Convolutional networks for images, speech, and time-series". English (US). In: *The handbook of brain theory and neural networks*. Ed. by M.A. Arbib. MIT Press.

LeCun, Yann, Yoshua Bengio, and Geoffrey E. Hinton (2015). "Deep learning". In: *Nat.* 521.7553, pp. 436–444. DOI: 10.1038/nature14539. URL: https://doi.org/10.1038/nature14539.

Leskovec, Jure (2021). "CS224W: Machine Learning with Graphs". In: *YouTube*. URL: https://www.youtube.com/watch?v=JAB_plj2rbA&list=PLoROMvodv4rPLKxIpqhjhPgab_channel=stanfordonline.

Li, Qimai, Zhichao Han, and Xiao-Ming Wu (2018). "Deeper Insights Into Graph Convolutional Networks for Semi-Supervised Learning". In: *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*. Ed. by Sheila A. McIlraith and Kilian Q. Weinberger. AAAI Press, pp. 3538–3545. URL: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16098.

Li, Ruiyu et al. (2017). "Situation Recognition with Graph Neural Networks". In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 4183–4192. DOI: 10.1109/ICCV.2017.448. URL: http://doi.ieeecomputersociety.org/10.1109/ICCV.2017.448.

Li, Yikang et al. (2017). "Scene graph generation from objects, phrases and region captions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1261–1270.

Li, Yujia, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel (2015). "Gated graph sequence neural networks". In: *arXiv preprint arXiv:1511.05493*.

Li, Yujia, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel (2016). "Gated Graph Sequence Neural Networks". In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. URL: http://arxiv.org/abs/1511.05493.

Lim, Marcus et al. (2019). "Link Prediction in Time-Evolving Criminal Network With Deep Reinforcement Learning Technique". In: *IEEE Access* 7, pp. 184797–184807. DOI: 10.1109/ACCESS.2019.2958873. URL: https://doi.org/10.1109/ACCESS.2019.2958873.

Lin, Tsung-Yi, Priya Goyal, et al. (2017). "Focal Loss for Dense Object Detection". In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*. IEEE Computer Society, pp. 2999–3007. DOI: 10.1109/ICCV.2017.324. URL: https://doi.org/10.1109/ICCV.2017.324.

Lin, Tsung-Yi, Michael Maire, et al. (2014). "Microsoft COCO: Common Objects in Context". In: *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, pp. 740–755. DOI: 10.1007/978-3-319-10602-1_48. URL: https://doi.org/10.1007/978-3-319-10602-1%5C_48.

Linsker, Ralph (1988). "Self-Organization in a Perceptual Network". In: *computer* 21.3, pp. 105–117. DOI: 10.1109/2.36. URL: https://doi.org/10.1109/2.36.

Liu, Meng, Zhengyang Wang, and Shuiwang Ji (2020). "Non-local graph neural networks". In: *arXiv preprint arXiv:2005.14612*.

Liu, Yanbei et al. (2020). "Independence Promoted Graph Disentangled Networks". 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*, pp. 4916–4923. URL: https://aaai.org/ojs/index.php/AAAI/article/view/5929.

Luo, Rui, Shan Zhao, and Zhiping Cai (2021). "Application of Graph Neural Network in Automatic Text Summarization". In: *Theoretical Computer Science*. Ed. by Kun He et al. Singapore: Springer Singapore, pp. 123–138. ISBN: 978-981-16-1877-2.

Lyu, Chunchuan and Ivan Titov (2018). "AMR Parsing as Graph Prediction with Latent Alignment". 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*. Ed. by Iryna Gurevych and Yusuke Miyao. Association for Computational Linguistics, pp. 397–407. DOI: 10.18653/v1/P18-1037. URL: https://aclanthology.org/P18-1037/.

Ma, Jianxin et al. (2019). "Disentangled Graph Convolutional Networks". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 4212–4221. URL: http://proceedings.mlr.press/v97/ma19a.html.

Ma, Tengfei and Jie Chen (2021). "Unsupervised Learning of Graph Hierarchical Abstractions with Differentiable Coarsening and Optimal Transport". In: *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, pp. 8856–8864. URL: https://ojs.aaai.org/index.php/AAAI/article/view/17072.

Malekzadeh, Touba et al. (2017). "Aircraft Fuselage Defect Detection using Deep Neural Networks". In: *CoRR* abs/1712.09213. arXiv: 1712.09213. URL: http://arxiv.org/abs/1712.09213.

Mallya, Arun and Svetlana Lazebnik (2016). "Learning Models for Actions and Person-Object Interactions with Transfer to Question Answering". In: *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part I*, pp. 414–428. DOI: 10.1007/978-3-319-46448-0_25. URL: https://doi.org/10.1007/978-3-319-46448-0%5C_25.

— (2017). "Recurrent Models for Situation Recognition". In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 455–463. DOI: 10.1109/ICCV.2017.57. URL: https://doi.org/10.1109/ICCV.2017.57.

McGregor, Andrew (2014). "Graph stream algorithms: a survey". In: *SIGMOD Rec.* 43.1, pp. 9–20. DOI: 10.1145/2627692.2627694. URL: https://doi.org/10.1145/2627692.2627694.

Meng, Fanyu et al. (Nov. 2020). "A structure-enhanced graph convolutional network for sentiment analysis". In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, pp. 586–595. DOI: 10.18653/v1/2020.findings-emnlp.52. URL: https://aclanthology.org/2020.findings-emnlp.52.

Merkwirth, Christian and Thomas Lengauer (2005). "Automatic generation of complementary descriptors with molecular graph networks". In: *Journal of chemical information and modeling* 45.5, pp. 1159–1168.

Mikolov, Tomás et al. (2013). "Efficient Estimation of Word Representations in Vector Space". In: *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. URL: http://arxiv.org/abs/1301.3781.

Monti, Federico et al. (2017). "Geometric Deep Learning on Graphs and Manifolds Using Mixture Model CNNs". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, pp. 5425–5434. DOI: 10.1109/CVPR.2017.576. URL: https://doi.org/10.1109/CVPR.2017.576.

Moon, Seunghyeon et al. (2016). "Parallel community detection on large graphs with MapReduce and GraphChi". In: *Data Knowledge Engineering* 104, pp. 17–31. ISSN: 0169-023X. DOI: https://doi.org/10.1016/j.datak.2015.05.001. URL: https://www.sciencedirect.com/science/article/pii/S0169023X15000208.

Nam, Hyeonseob, Jung-Woo Ha, and Jeonghee Kim (2017). "Dual Attention Networks for Multimodal Reasoning and Matching". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 2156–

2164. DOI: 10.1109/CVPR.2017.232. URL: https://doi.org/10.1109/CVPR.2017.232.

Narayanan, Annamalai et al. (2017). "graph2vec: Learning Distributed Representations of Graphs". In: *CoRR* abs/1707.05005. arXiv: 1707.05005. URL: http://arxiv.org/abs/1707.05005.

Nasution, Mahyuddin KM (2016). "Social network mining (SNM): A definition of relation between the resources and SNA". In: *International Journal on Advanced Science, Engineering and Information Technology* 6.6, pp. 975–981.

Newman, M. E. J. and M. Girvan (Feb. 2004). "Finding and evaluating community structure in networks". In: *Phys. Rev. E* 69 (2), p. 026113. DOI: 10.1103/PhysRevE.69.026113. URL: https://link.aps.org/doi/10.1103/PhysRevE.69.026113.

Nivre, Joakim and Ryan T. McDonald (2008). "Integrating Graph-Based and Transition-Based Dependency Parsers". In: *ACL 2008, Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, June 15-20, 2008, Columbus, Ohio, USA*. Ed. by Kathleen R. McKeown et al. The Association for Computer Linguistics, pp. 950–958. URL: https://aclanthology.org/P08-1108/.

Noldus, Rogier and Piet Van Mieghem (2015). "Assortativity in complex networks". In: *J. Complex Networks* 3.4, pp. 507–542. DOI: 10.1093/comnet/cnv005. URL: https://doi.org/10.1093/comnet/cnv005.

Pan, Shirui et al. (2018). "Adversarially Regularized Graph Autoencoder for Graph Embedding". In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pp. 2609–2615. DOI: 10.24963/ijcai.2018/362. URL: https://doi.org/10.24963/ijcai.2018/362.

Park, Jiwoong et al. (2019). "Symmetric Graph Convolutional Autoencoder for Unsupervised Graph Representation Learning". In: *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pp. 6518–6527. DOI: 10.1109/ICCV.2019.00662. URL: https://doi.org/10.1109/ICCV.2019.00662.

Paszke, Adam, Sam Gross, Soumith Chintala, et al. (2017). "Automatic differentiation in PyTorch". In: *NIPS-W*.

Paszke, Adam, Sam Gross, Francisco Massa, et al. (2019). "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Pei, Hongbin et al. (2020). "Geom-GCN: Geometric Graph Convolutional Networks". In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. URL: https://openreview.net/forum?id=S1e2agrFvS.

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning (2014). "Glove: Global Vectors for Word Representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*. Ed. by Alessandro

Moschitti, Bo Pang, and Walter Daelemans. ACL, pp. 1532–1543. DOI: 10.3115/v1/d14-1162. URL: https://doi.org/10.3115/v1/d14-1162.

Perozzi, Bryan, Rami Al-Rfou, and Steven Skiena (2014). "DeepWalk: online learning of social representations". In: *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*. Ed. by Sofus A. Macskassy et al. ACM, pp. 701–710. DOI: 10.1145/2623330.2623732. URL: https://doi.org/10.1145/2623330.2623732.

Pistilli, Francesca et al. (2020). "Learning Graph-Convolutional Representations for Point Cloud Denoising". In: *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XX*. Ed. by Andrea Vedaldi et al. Vol. 12365. Lecture Notes in Computer Science. Springer, pp. 103–118. DOI: 10.1007/978-3-030-58565-5_7. URL: https://doi.org/10.1007/978-3-030-58565-5%5C_7.

Plummer, Bryan A et al. (2015). "Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models". In: *Computer Vision (ICCV), 2015 IEEE International Conference on*. IEEE, pp. 2641–2649.

Qiu, Jiezhong et al. (2020). "GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training". In: *arXiv preprint arXiv:2006.09963*.

Quarterman, John S (1990). *The matrix: Computer networks and conferencing systems worldwide*. Digital Press.

Quiterio, Thaise M. and Ana Carolina Lorena (2018). "Using complexity measures to determine the structure of directed acyclic graphs in multiclass classification". In: *Appl. Soft Comput.* 65, pp. 428–442. DOI: 10.1016/j.asoc.2018.01.013. URL: https://doi.org/10.1016/j.asoc.2018.01.013.

Ren, Shaoqing et al. (2015). "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 91–99. URL: http://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.

Ridge, Justin T. et al. (2020). "Deep learning for coastal resource conservation: automating detection of shellfish reefs". In: *Remote Sensing in Ecology and Conservation* 6.4, pp. 431–440. DOI: https://doi.org/10.1002/rse2.134. eprint: https://zslpublications.onlinelibrary.wiley.com/doi/pdf/10.1002/rse2.134. URL: https://zslpublications.onlinelibrary.wiley.com/doi/abs/10.1002/rse2.134.

Robertson, Brett W. et al. (2019). "Using a combination of human insights and 'deep learning' for real-time disaster communication". In: *Progress in Disaster Science* 2, p. 100030. ISSN: 2590-0617. DOI: https://doi.org/10.1016/j.pdisas.2019.100030. URL: https://www.sciencedirect.com/science/article/pii/S2590061719300304.

Rong, Yu et al. (2020). "DropEdge: Towards Deep Graph Convolutional Networks on Node Classification". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=Hkx1qkrKPr.

Rozemberczki, Benedek, Carl Allen, and Rik Sarkar (2021). "Multi-Scale attributed node embedding". In: *J. Complex Networks* 9.2. DOI: 10.1093/comnet/cnab014. URL: https://doi.org/10.1093/comnet/cnab014.

Sanchez-Lengeling, Benjamin and Alán Aspuru-Guzik (2018). "Inverse molecular design using machine learning: Generative models for matter engineering". In: *Science* 361.6400, pp. 360–365. ISSN: 0036-8075. DOI: 10.1126/science.aat2663. eprint: https://science.sciencemag.org/content/361/6400/360.full.pdf. URL: https://science.sciencemag.org/content/361/6400/360.

Sanfeliu, A. and K. Fu (1983). "A distance measure between attributed relational graphs for pattern recognition". In: *IEEE Transactions on Systems, Man, and Cybernetics* SMC-13.3, pp. 353–362. DOI: 10.1109/TSMC.1983.6313167.

Sarrafzadeh, Bahareh, Olga Vechtomova, and Vlado Jokic (2014). "Exploring knowledge graphs for exploratory search". In: *Fifth Information Interaction in Context Symposium, IIiX '14, Regensburg, Germany, August 26-29, 2014*. Ed. by David Elsweiler et al. ACM, pp. 135–144. DOI: 10.1145/2637002.2637019. URL: https://doi.org/10.1145/2637002.2637019.

Schneider, Gisbert (2013). *De novo molecular design*. John Wiley & Sons.

Schneider, Gisbert and Uli Fechner (2005). "Computer-based de novo design of drug-like molecules". In: *Nature Reviews Drug Discovery* 4.8, pp. 649–663.

Schuster, M. and K.K. Paliwal (1997). "Bidirectional recurrent neural networks". In: *IEEE Transactions on Signal Processing* 45.11, pp. 2673–2681. DOI: 10.1109/78.650093.

Sharma, Gaurav, Frédéric Jurie, and Cordelia Schmid (2013). "Expanded Parts Model for Human Attribute and Action Recognition in Still Images". In: *2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, June 23-28, 2013*, pp. 652–659. DOI: 10.1109/CVPR.2013.90. URL: https://doi.org/10.1109/CVPR.2013.90.

Shervashidze, Nino, Pascal Schweitzer, et al. (2011). "Weisfeiler-Lehman Graph Kernels". In: *J. Mach. Learn. Res.* 12, pp. 2539–2561. URL: http://dl.acm.org/citation.cfm?id=2078187.

Shervashidze, Nino, S. V. N. Vishwanathan, et al. (2009). "Efficient graphlet kernels for large graph comparison". In: *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, AISTATS 2009, Clearwater Beach, Florida, USA, April 16-18, 2009*, pp. 488–495. URL: http://proceedings.mlr.press/v5/shervashidze09a.html.

Silberer, Carina and Manfred Pinkal (2018). "Grounding Semantic Roles in Images". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pp. 2616–2626. URL: https://www.aclweb.org/anthology/D18-1282/.

Simonyan, K. and A. Zisserman (2015). "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *International Conference on Learning Representations*.

Song, Sibo et al. (2018). "Deep Adaptive Temporal Pooling for Activity Recognition". In: *ACM Multimedia*.

Srivastava, Nitish et al. (2014). "Dropout: a simple way to prevent neural networks from overfitting". In: *J. Mach. Learn. Res.* 15.1, pp. 1929–1958. URL: http://dl.acm.org/citation.cfm?id=2670313.

Suhonen, Jukka et al. (2012). *Low-Power Wireless Sensor Networks - Protocols, Services and Applications*. Springer Briefs in Electrical and Computer Engineering. Springer. ISBN: 978-1-4614-2172-6. DOI: 10.1007/978-1-4614-2173-3. URL: https://doi.org/10.1007/978-1-4614-2173-3.

Sun, Fan-Yun et al. (2020). "InfoGraph: Unsupervised and Semi-supervised Graph-Level Representation Learning via Mutual Information Maximization". In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. URL: https://openreview.net/forum?id=r1lfF2NYvH.

Sun, Yizhou and Jiawei Han (2012). "Mining heterogeneous information networks: a structural analysis approach". In: *SIGKDD Explor.* 14.2, pp. 20–28. DOI: 10.1145/2481244.2481248. URL: https://doi.org/10.1145/2481244.2481248.

Tan, Jiwei, Xiaojun Wan, and Jianguo Xiao (2017). "Abstractive Document Summarization with a Graph-Based Attentional Neural Model". 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*. Ed. by Regina Barzilay and Min-Yen Kan. Association for Computational Linguistics, pp. 1171–1181. DOI: 10.18653/v1/P17-1108. URL: https://doi.org/10.18653/v1/P17-1108.

Tang, Jie et al. (2009). "Social influence analysis in large-scale networks". In: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28 - July 1, 2009*. Ed. by John F. Elder IV et al. ACM, pp. 807–816. DOI: 10.1145/1557019.1557108. URL: https://doi.org/10.1145/1557019.1557108.

Teney, Damien, Peter Anderson, et al. (2017). "Tips and tricks for visual question answering: Learnings from the 2017 challenge". In: *arXiv preprint arXiv:1708.02711*.

Teney, Damien, Lingqiao Liu, and Anton van den Hengel (2017). "Graph-Structured Representations for Visual Question Answering". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, pp. 3233–3241. DOI: 10.1109/CVPR.2017.344. URL: https://doi.org/10.1109/CVPR.2017.344.

Tian, Yifei et al. (2021). "DGCB-Net: Dynamic Graph Convolutional Broad Network for 3D Object Recognition in Point Cloud". In: *Remote. Sens.* 13.1, p. 66. DOI: 10.3390/rs13010066. URL: https://doi.org/10.3390/rs13010066.

Tian, Yonglong, Dilip Krishnan, and Phillip Isola (2020). "Contrastive Multiview Coding". In: *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XI*. Ed. by Andrea Vedaldi et al. Vol. 12356. Lecture Notes in Computer Science. Springer, pp. 776–794. DOI: 10.1007/978-3-030-58621-8_45. URL: https://doi.org/10.1007/978-3-030-58621-8%5C_45.

Vaswani, Ashish et al. (2017). "Attention is All you Need". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 6000–6010. URL: http://papers.nips.cc/paper/7181-attention-is-all-you-need.

Velickovic, Petar, Guillem Cucurull, et al. (2018). "Graph Attention Networks". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. URL: https://openreview.net/forum?id=rJXMpikCZ.

Velickovic, Petar, William Fedus, et al. (2019). "Deep Graph Infomax". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. URL: https://openreview.net/forum?id=rklz9iAcKQ.

Veličković, Petar et al. (2018). "Graph Attention Networks". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=rJXMpikCZ.

Veyseh, Amir Pouran Ben et al. (2020). "Improving Aspect-based Sentiment Analysis with Gated Graph Convolutional Networks and Syntax-based Regulation". In: *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*. Ed. by Trevor Cohn, Yulan He, and Yang Liu. Vol. EMNLP 2020. Findings of ACL. Association for Computational Linguistics, pp. 4543–4548. DOI: 10.18653/v1/2020.findings-emnlp.407. URL: https://doi.org/10.18653/v1/2020.findings-emnlp.407.

Vinyals, Oriol, Samy Bengio, and Manjunath Kudlur (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*. Ed. by Yoshua Bengio and Yann LeCun. URL: http://arxiv.org/abs/1511.06391.

Wang, Jian et al. (2020). "Graph-PCNN: Two Stage Human Pose Estimation with Graph Pose Refinement". In: *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XI*. Ed. by Andrea Vedaldi et al. Vol. 12356. Lecture Notes in Computer Science. Springer, pp. 492–508. DOI: 10.1007/978-3-030-58621-8_29. URL: https://doi.org/10.1007/978-3-030-58621-8%5C_29.

Wang, Li et al. (2011). "Predicting Thread Discourse Structure over Technical Web Forums". In: *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*. ACL, pp. 13–25. URL: https://aclanthology.org/D11-1002/.

Wang, Yue et al. (2019). "Dynamic Graph CNN for Learning on Point Clouds". In: *ACM Trans. Graph.* 38.5, 146:1–146:12. DOI: 10.1145/3326362. URL: https://doi.org/10.1145/3326362.

Weisfeiler, Boris and Andrei Leman (1968). "The reduction of a graph to canonical form and the algebra which appears therein". In: *NTI, Series* 2.9, pp. 12–16.

Wieder, Oliver et al. (2020). "A compact review of molecular property prediction with graph neural networks". In: *Drug Discovery Today: Technologies*. ISSN: 1740-6749. DOI: https://doi.org/10.1016/j.ddtec.2020.11.009. URL: https://www.sciencedirect.com/science/article/pii/S1740674920300305.

Xu, Danfei, Yuke Zhu, Christopher B Choy, et al. (2017a). "Scene graph generation by iterative message passing". In: *arXiv preprint arXiv:1701.02426*.

— (2017b). "Scene Graph Generation by Iterative Message Passing". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 3097–3106. DOI: 10.1109/CVPR.2017.330. URL: https://doi.org/10.1109/CVPR.2017.330.

Xu, Guangluan et al. (2019). "Edge-Nodes Representation Neural Machine for Link Prediction". In: *Algorithms* 12.1, p. 12. DOI: 10.3390/a12010012. URL: https://doi.org/10.3390/a12010012.

Xu, Huan et al. (2013). "Node Classification in Social Network via a Factor Graph Model". In: *Advances in Knowledge Discovery and Data Mining, 17th Pacific-Asia Conference, PAKDD 2013, Gold Coast, Australia, April 14-17, 2013, Proceedings, Part I*. Ed. by Jian Pei et al. Vol. 7818. Lecture Notes in Computer Science. Springer, pp. 213–224. DOI: 10.1007/978-3-642-37453-1_18. URL: https://doi.org/10.1007/978-3-642-37453-1%5C_18.

Xu, Keyulu et al. (2019). "How Powerful are Graph Neural Networks?" In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. URL: https://openreview.net/forum?id=ryGs6iA5Km.

Xu, Minghao et al. (2021). "Self-supervised Graph-level Representation Learning with Local and Global Structure". In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 11548–11558. URL: http://proceedings.mlr.press/v139/xu21g.html.

Yanardag, Pinar and S. V. N. Vishwanathan (2015). "Deep Graph Kernels". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pp. 1365–1374. DOI: 10.1145/2783258.2783417. URL: https://doi.org/10.1145/2783258.2783417.

Yang, Kevin et al. (2019). "Analyzing learned molecular representations for property prediction". In: *Journal of chemical information and modeling* 59.8, pp. 3370–3388.

Yang, Shaohua et al. (2016). "Grounded Semantic Role Labeling". In: *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pp. 149–159. URL: https://www.aclweb.org/anthology/N16-1019/.

Yang, Xu et al. (2019). "Auto-Encoding Scene Graphs for Image Captioning". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, pp. 10685–10694. DOI: 10.1109/CVPR.2019.01094. URL: http://openaccess.thecvf.com/content%5C_CVPR%5C_2019/html/Yang%5C_Auto-Encoding%5C_Scene%5C_Graphs%5C_for%5C_Image%5C_Captioning%5C_CVPR%5C_2019%5C_paper.html.

Yang, Yiding et al. (2020). "Factorizable Graph Convolutional Networks". In: *CoRR* abs/2010.05421. arXiv: 2010.05421. URL: https://arxiv.org/abs/2010.05421.

Yang, Zhilin, William W. Cohen, and Ruslan Salakhutdinov (2016). "Revisiting Semi-Supervised Learning with Graph Embeddings". In: *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pp. 40–48. URL: http://proceedings.mlr.press/v48/yanga16.html.

Yasunaga, Michihiro et al. (Aug. 2017). "Graph-based Neural Multi-Document Summarization". In: *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*. Vancouver, Canada: Association for Computational Linguistics, pp. 452–462. DOI: 10.18653/v1/K17-1045. URL: https://aclanthology.org/K17-1045.

Yatskar, Mark, Vicente Ordonez, et al. (2017). "Commonly uncommon: Semantic sparsity in situation recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yatskar, Mark, Luke Zettlemoyer, and Ali Farhadi (2016). "Situation Recognition: Visual Semantic Role Labeling for Image Understanding". In: *Conference on Computer Vision and Pattern Recognition*.

Yeh, Peter Z. and Adwait Ratnaparkhi (2014). "Mining Large-Scale Knowledge Graphs to Discover Inference Paths for Query Expansion in NLIDB". In: *2014 AAAI Fall Symposia, Arlington, Virginia, USA, November 13-15, 2014*. AAAI Press. URL: http://www.aaai.org/ocs/index.php/FSS/FSS14/paper/view/9135.

Ying, Zhitao et al. (2018). "Hierarchical Graph Representation Learning with Differentiable Pooling". In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. Ed. by Samy Bengio et al., pp. 4805–4815. URL: https://proceedings.neurips.cc/paper/2018/hash/e77dbaf6759253c7c6d0efc5690369c7-Abstract.html.

You, Yuning, Tianlong Chen, Yang Shen, et al. (2021). "Graph Contrastive Learning Automated". In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 12121–12132. URL: http://proceedings.mlr.press/v139/you21a.html.

You, Yuning, Tianlong Chen, Yongduo Sui, et al. (2020). "Graph Contrastive Learning with Augmentations". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle et al. URL: https://proceedings.neurips.cc/paper/2020/hash/3fe230348e9a12c13120749e3f9fa4cd-Abstract.html.

Yu, Weijiang et al. (2019). "Heterogeneous Graph Learning for Visual Commonsense Reasoning". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by Hanna M. Wallach et al., pp. 2765–2775. URL: https://proceedings.neurips.cc/paper/2019/hash/8f19793b2671094e63a15ab883Abstract.html.

Yu, Zhou et al. (2018). "Beyond Bilinear: Generalized Multimodal Factorized High-Order Pooling for Visual Question Answering". In: *IEEE Transactions on Neural Networks and Learning Systems* 29.12, pp. 5947–5959.

Zaman, Sameia et al. (2019). "A Recurrent Neural Network Approach to Image Captioning in Braille for Blind-Deaf People". In: *2019 IEEE International Conference on Signal Processing, Information, Communication Systems (SPICSCON)*, pp. 49–53. DOI: 10.1109/SPICSCON48833.2019.9065144.

Zellers, Rowan et al. (June 2019). "From Recognition to Cognition: Visual Commonsense Reasoning". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zeng, Jiaqi and Pengtao Xie (2021). "Contrastive Self-supervised Learning for Graph Classification". In: *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021,*

The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021. AAAI Press, pp. 10824–10832. URL: https://ojs.aaai.org/index.php/AAAI/article/view/17293.

Zhang, Muhan et al. (2018). "An End-to-End Deep Learning Architecture for Graph Classification". In: *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*, pp. 4438–4445. URL: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17146.

Zhang, Sheng et al. (July 2019). "AMR Parsing as Sequence-to-Graph Transduction". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 80–94. DOI: 10.18653/v1/P19-1009. URL: https://aclanthology.org/P19-1009.

Zhang, Shuai et al. (2019). "Deep Learning Based Recommender System: A Survey and New Perspectives". In: *ACM Comput. Surv.* 52.1, 5:1–5:38. DOI: 10.1145/3285029. URL: https://doi.org/10.1145/3285029.

Zhang, Weishan et al. (2016). "Distributed embedded deep learning based real-time video processing". In: *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 001945–001950. DOI: 10.1109/SMC.2016.7844524.

Zhang, Xi, Feifei Zhang, and Changsheng Xu (2021). "Multi-Level Counterfactual Contrast for Visual Commonsense Reasoning". In: *MM '21: ACM Multimedia Conference, Virtual Event, China, October 20 - 24, 2021*. Ed. by Heng Tao Shen et al. ACM, pp. 1793–1802. DOI: 10.1145/3474085.3475328. URL: https://doi.org/10.1145/3474085.3475328.

Zhang, Zhiwang et al. (2021). "Dense Video Captioning Using Graph-Based Sentence Summarization". In: *IEEE Transactions on Multimedia* 23, pp. 1799–1810. DOI: 10.1109/TMM.2020.3003592.

Zhang, Ziwei, Peng Cui, and Wenwu Zhu (2020). "Deep learning on graphs: A survey". In: *IEEE Transactions on Knowledge and Data Engineering*.

Zhao, Lingxiao and Leman Akoglu (2020). "PairNorm: Tackling Oversmoothing in {GNN}s". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=rkecl1rtwB.

Zheng, Yue, Yali Li, and Shengjin Wang (2018). "Intention Oriented Image Captions with Guiding Objects". In: *CoRR* abs/1811.07662. arXiv: 1811.07662. URL: http://arxiv.org/abs/1811.07662.

Zhong, Yiwu et al. (2020). "Comprehensive Image Captioning via Scene Graph Decomposition". In: *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XIV*. Ed. by Andrea Vedaldi et al. Vol. 12359. Lecture Notes in Computer Science. Springer, pp. 211–229. DOI: 10.1007/978-3-030-58568-6_13. URL: https://doi.org/10.1007/978-3-030-58568-6%5C_13.

Zunger, Alex (2018). "Inverse design in search of materials with target functionalities". In: *Nature Reviews Chemistry* 2.4, pp. 1–16.