Learning Better Representation Using Auxiliary Knowledge

by

Saed Rezayi

(Under the Direction of Sheng Li)

Abstract

Representation learning has become a vital part of machine learning in recent years, as it can automatically learn useful representations of complex data. While representation learning has shown impressive results in various domains, there is still a need to further improve its performance. One way to enhance the performance of representation learning is to incorporate auxiliary knowledge sources, such as text, knowledge graphs, and similar datasets.

This dissertation focuses on exploring the use of auxiliary knowledge sources to improve representation learning. We will discuss the advantages of utilizing text, knowledge graphs, and similar datasets to augment the representation learning process. Specifically, we will explore how transfer learning from text data can significantly improve the performance of language models in natural language processing tasks. We will also investigate the use of knowledge graphs to capture relationships between entities in a structured way, which can be leveraged to improve the accuracy of machine learning models in various tasks such as recommendation and fraud detection. Additionally, we will examine the benefits of incorporating similar datasets into representation learning, as it can provide additional data to help reduce overfitting and improve generalization performance.

Furthermore, we will discuss the necessary conditions for auxiliary knowledge sources to be useful in representation learning. The quality and relevance

of auxiliary knowledge sources play an important role in the success of these techniques. Therefore, we will explore various techniques for selecting and integrating auxiliary knowledge sources into representation learning models. Overall, in this dissertation, we will discuss various implementations of using auxiliary knowledge to improve representation learning. These techniques have shown significant potential for enhancing the performance of machine learning models across various domains, and their continued exploration is crucial for further advancements in representation learning.

INDEX WORDS: Natural Language Processing, Knowledge Graphs, Auxiliary Knowledge, Representation Learning

Learning Better Representation Using Auxiliary Knowledge

by

Saed Rezayi

M.S., University of Oregon, 2017

A Dissertation Submitted to the Graduate Faculty of the
University of Georgia in Partial Fulfillment of the Requirements for the
Degree.

Doctor of Philosophy

Athens, Georgia

2023

Learning Better Representation Using Auxiliary Knowledge

by

Saed Rezayi

Major Professor:	Professor Sheng Li
Committee:	Professor Sheng Li
	Professor Khaled Rasheed
	Professor Pengsheng Ji

Electronic Version Approved:

Ron Walcott
Dean of the Graduate School
The University of Georgia
May 2023

Acknowledgments

I am deeply grateful to all those who have contributed to my academic journey and supported me in countless ways. Without their guidance and encouragement, this achievement would not have been possible.

First and foremost, I would like to express my sincere gratitude to my advisor, Dr. Sheng Li, for his invaluable guidance, support, and encouragement throughout my academic journey. His expertise, insights, and mentorship have been instrumental in shaping my research and professional development. I am also thankful to my other committee members, Dr. Khaled Rasheed and Dr. Pengsheng Ji, for their valuable feedback and constructive criticism.

I am grateful to all the students in the Rise lab at University of Georgia, especially Ronghang Zhu, for their friendship, camaraderie, and support. Their insights and feedback have been critical in shaping my research ideas and methods. I would also like to thank all my collaborators and mentors in various research projects, including Handong Zhao, Nedim Lipka, Ryan Rossi, and many others, for their valuable guidance, support, and insights.

Finally, I would like to express my heartfelt appreciation to my family, including my parents, siblings, wife, and nephew, Parsa, for their unconditional love, encouragement, and support. They have been my rock throughout this journey, providing me with the strength and inspiration to overcome challenges and achieve my goals.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Representation learning has become a crucial tool in machine learning as it allows algorithms to learn meaningful representations of the data that can be used for various downstream tasks. Auxiliary knowledge, such as domain knowledge or external information, has been shown to be useful in representation learning. One way that auxiliary knowledge can be helpful is by providing sparsity constraints on the learned representations. Sparsity can be thought of as a way to reduce the dimensionality of the representation, while maintaining its expressiveness. This can be especially useful when dealing with high-dimensional data. For example, in natural language processing, the use of sparse features has been shown to improve performance on various tasks Reisinger and Mooney, 2010.

Another way that auxiliary knowledge can be helpful in representation learning is by leveraging unlabeled data. Unlabeled data is often abundant and cheap to acquire, making it an attractive source of information. However, learning from unlabeled data can be challenging, as there is no explicit supervision signal. Auxiliary knowledge can be used to provide an implicit supervision signal that can help guide the learning process. For example, in image recognition, unsupervised pre-training has been shown to improve performance on downstream tasks Erhan et al., 2010.

Some forms of auxiliary knowledge, such as knowledge graphs, are carefully curated and contain useful domain information that can be leveraged in representation learning. Knowledge graphs are structured representations of domain knowledge that encode relationships between entities. They have been shown to be useful in a variety of applications, such as entity linking and question answering Bordes et al., 2013a. In representation learning, knowledge graphs can be used to provide additional constraints on the learned representations, such as enforcing that related entities have similar representations Q. Wang et al., 2017.

Different modalities have also been used to improve the learned representation. For example, in multimodal learning, information from multiple modalities, such as text, image, and audio, is combined to learn a joint representation Ngiam et al., 2011. This can be useful in applications such as video captioning, where the goal is to generate a natural language description of a video. In transfer learning, knowledge learned from one task can be transferred to another task S. J. Pan and Yang, 2010. For example, a model trained on a large text corpus can be used as a pre-training step for natural language understanding tasks. In reinforcement learning, auxiliary tasks can be used to improve the learning of the primary task Jaderberg et al., 2017. For example, in the game of Go, auxiliary tasks such as predicting the outcome of the game or the next move can be used to improve the performance of the primary task, which is to win the game.

1.2 Text as Auxiliary Knowledge

Text is a valuable source of knowledge due to its ubiquity and diversity. It is one of the most common forms of communication and can be found in a wide variety of formats, including books, articles, social media posts, and chat logs. Text can be used to convey information about many different types of objects and concepts, making it a rich source of domain-specific knowledge. In addition, the structure of text, such as the way sentences are constructed and the relationships between words, can provide valuable information for learning representations.

Text has been used in many different ways to improve representation learning. One common approach is to use pre-trained language models, such as BERT Devlin et al., 2019, as a starting point for learning representations. These models are trained on large amounts of text and can be fine-tuned for specific tasks, such as sentiment analysis or named entity recognition, to improve performance. Another approach is to use text as a source of supervision for other modalities, such as images or audio. For example, in the task of image captioning, text can be used as a source of supervision for learning to generate natural language descriptions of images Karpathy and Fei-Fei, 2015.

In order for text to be a useful source of knowledge for representation learning, it should meet certain conditions. One important condition is that the text should be relevant to the task at hand. This means that the text should contain information that is related to the objects or concepts being represented. In addition, the text should be diverse and representative of the different types of objects and concepts in the domain. This can help ensure that the learned representations are robust and generalize well to new examples. Finally, the text should be of high quality and free of errors, as errors in the text can lead to errors in the learned representations. Chapter 2 provides a successful example of using external text to improve the task of knowledge graph embedding.

1.3 Knowledge Graphs as Auxiliary Knowledge

Knowledge graphs are a useful source of knowledge because they provide a structured representation of information that can be used to enhance various machine learning tasks. A knowledge graph is a directed graph that represents knowledge as a set of entities and their relationships. Knowledge graphs can be constructed from a variety of sources, including structured data, unstructured text, and ontologies. By representing knowledge in a structured form, knowledge graphs can be used to enhance tasks such as question answering, recommendation systems, and natural language processing.

Knowledge graphs have shown to be useful for representation learning in various fields, such as recommendation systems and fraud detection. In rec-

ommendation systems, knowledge graphs have been used to improve the accuracy of personalized recommendations. For example, (R. Wang et al., 2019) proposed a knowledge graph convolutional network (KGCN) that takes into account both user-item interactions and the knowledge graph structure to make recommendations. KGCN achieved state-of-the-art performance on several benchmark datasets. In another study, (Q. Guo et al., 2020) used a knowledge graph to improve link prediction in a recommendation system. In fraud detection, knowledge graphs have been used to represent relationships between entities, such as accounts and transactions, to detect fraudulent behavior. For example, (Qin et al., 2018) proposed a graph convolutional network (GCN) that uses a knowledge graph to model the relationship between accounts and transactions. GCN achieved better performance than traditional fraud detection methods on several benchmark datasets.

In addition, knowledge graphs have been used in natural language processing (NLP) tasks such as entity recognition and relation extraction. For example, (Sung et al., 2022) used a knowledge graph to improve named entity recognition (NER) performance in the biomedical domain. Their approach, called multi-level adaptive attention, integrates a knowledge graph and a concept ontology to improve the identification of named entities in text. Overall, knowledge graphs provide a powerful tool for representation learning by enabling the integration of structured and unstructured data, as well as providing a structured representation of relationships between entities. The examples discussed above demonstrate how knowledge graphs can be used to improve the accuracy of machine learning models in various tasks.

To be a useful source of knowledge, a knowledge graph should have several properties. First, it should be comprehensive, meaning that it should contain a large number of entities and relationships. Second, it should be accurate, meaning that the entities and relationships in the knowledge graph should be correctly labeled and annotated. Third, it should be up-to-date, meaning that the knowledge graph should be regularly updated to reflect changes in the domain. Fourth, it should be easily accessible, meaning that the knowledge graph should be publicly available and easy to query. Overall, knowledge graphs are a useful

source of knowledge that can be used to enhance various machine learning tasks. By providing a structured representation of knowledge, knowledge graphs can help machine learning models better understand the world and make more accurate predictions and decisions. In the forthcoming chapters, we will delve into two distinct approaches for utilizing knowledge graphs to enhance representation learning. Chapter 3 and 4 will each cover a different implementation of this technique.

1.4 Similar Datasets as Auxiliary Knowledge

Similar datasets can be a useful source of knowledge for representation learning because they provide additional data that can help to improve the accuracy of machine learning models. When training a model, having a larger and more diverse dataset can help to reduce overfitting and improve generalization performance. Similar datasets can also provide additional context or perspective on the problem being tackled, which can help to uncover patterns or relationships that may not be apparent from a single dataset alone.

There are many examples of how similar datasets can be useful for representation learning. One example is in natural language processing, where multiple datasets with similar tasks have been used to improve the performance of language models. For instance, (Howard & Ruder, 2018) used transfer learning from a similar dataset to improve the accuracy of a text classification model. Another example is in computer vision, where similar datasets have been used to improve the performance of image recognition models. For example, (Kornblith et al., 2019) used transfer learning from similar datasets to improve the accuracy of an image recognition model.

To be a useful source of knowledge, similar datasets should have some degree of overlap in the target variable or task. The datasets should also be diverse enough to provide additional context or perspective on the problem being tackled, but not so diverse that they are fundamentally different from the original dataset. Additionally, the datasets should be of high quality and representative of the domain of interest. Finally, it is worth noting that similar datasets can

also be useful for improving the interpretability of machine learning models. By providing additional context or perspective on the problem being tackled, similar datasets can help to uncover relationships or patterns that may not be apparent from a single dataset alone. This can be particularly useful in domains where model interpretability is important, such as healthcare or finance. In Chapter 5, we will explore the usage of similar datasets to enhance representation learning. We will discuss various approaches and techniques for leveraging similar datasets to improve the performance of machine learning models.

1.5 Summary

Representation learning has become an increasingly popular area of research due to its ability to automatically learn effective representations of complex data. One of the key factors that can help to improve the performance of representation learning models is the incorporation of auxiliary knowledge sources, such as text, knowledge graphs, and similar datasets. These knowledge sources can provide additional context or perspective on the problem being tackled, which can help to uncover patterns or relationships that may not be apparent from a single dataset alone.

Text data is one of the most widely available sources of auxiliary knowledge and can provide valuable information for representation learning. Text can be used to provide additional context or perspective on the problem being tackled, as well as to improve the interpretability of machine learning models. Recent research has shown that transfer learning from text data can significantly improve the performance of language models in natural language processing tasks. Similarly, knowledge graphs are a useful source of knowledge that can provide valuable domain-specific information for representation learning. Knowledge graphs can be used to capture relationships between entities in a structured way, which can be leveraged to improve the performance of machine learning models in various tasks such as recommendation and fraud detection. Similar datasets can also be a useful source of knowledge for representation learning, as they can provide additional data that can help to reduce overfitting and improve

generalization performance. Transfer learning from similar datasets has been shown to improve the accuracy of machine learning models in various domains, including natural language processing and healthcare.

In conclusion, incorporating auxiliary knowledge sources can significantly improve the performance of representation learning models. However, it is important to carefully consider the quality and relevance of these knowledge sources to ensure that they are truly useful for the task at hand. The development of novel techniques for incorporating auxiliary knowledge into representation learning models is an important area of research, and future work in this area is likely to yield significant improvements in the performance of machine learning models across a wide range of domains.

Chapter 2

Text as Auxiliary Knowledge

Knowledge graphs suffer from sparsity which degrades the quality of representations generated by various methods. While there is an abundance of textual information throughout the web and many existing knowledge bases, aligning information across these diverse data sources remains a challenge in the literature. Previous work has partially addressed this issue by enriching knowledge graph entities based on *"hard"* co-occurrence of words present in the entities of the knowledge graphs and external text, while we achieve *"soft"* augmentation by proposing a knowledge graph enrichment and embedding framework named Edge. Given an original knowledge graph, we first generate a rich but noisy augmented graph using external texts in semantic and structural level. To distill the relevant knowledge and suppress the introduced noise, we design a graph alignment term in a shared embedding space between the original and augmented graph. To enhance the embedding learning on the augmented graph, we further regularize the locality relationship of target entity based on negative sampling. Experimental results on four benchmark datasets demonstrate the robustness and effectiveness of Edge in link prediction and node classification.

2.1 Introduction

Knowledge Graph (KG)[1] embedding has been an emerging research topic in natural language processing, which aims to learn a low dimensional latent vector for every node.

One major challenge is sparsity. Knowledge graphs are often incomplete, and it is a challenge to generate low-dimensional representations from a graph with many missing edges. To mitigate this issue, auxiliary texts that are easily accessible have been popularly exploited for enhancing the KG (as illustrated in Figure 2.1). More specifically, given that KG entities contain textual features, we can link them to an auxiliary source of knowledge, e.g., WordNet, and therefore enhance the existing feature space. With notable exceptions, the use of external textual properties for KG embedding has not been extensively explored before. Recently, Kartsaklis et al., 2018 used entities of the KG to query BabelNet (Navigli & Ponzetto, 2012), added new nodes to the original KG based on co-occurrence of entities, and produced more meaningful embeddings using the enriched graph. However, this hard-coded, co-occurrence based KG enrichment strategy fails to make connections to other semantically related entities. As motivated in Figure 2.1, the newly added entities "*wound*", "*arthropod*" and "*protective body*", are semantically close to some input KG entity nodes (marked in red). However, they cannot be directly retrieved from BabelNet using co-occurrence matching.

In this chapter, we aim to address the sparsity issue by integrating a learning component into the process. We propose a novel framework, EDGE, for KG enrichment and embedding. EDGE first constructs a graph using the external text based on similarity and aligns the enriched graph with the original KG in the same embedding space. It infuses learning in the knowledge distillation process by graph alignment, ensuring that similar entities remain close, and dissimilar entities get as far from each other. Consuming information from an auxiliary textual source helps improve the quality of final products, i.e., low dimensional embeddings, by introducing new features. This new feature space is effective

[1] Knowledge graph usually represents a heterogeneous multigraph whose nodes and relations can have different types. However in the work, we follow (Kartsaklis et al., 2018), consider knowledge graph enrichment problem where only one relation type (connected or not) appears.

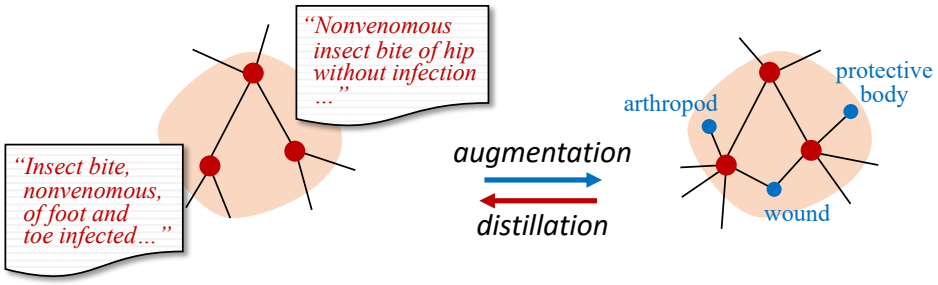


Figure 2.1: An example illustrating the original (left) and augmented knowledge graphs (right). Red nodes are knowledge graph entities and small blue nodes are textual nodes obtained from the external text. In augmentation process, a new set of keywords are discovered and attached to the original entities. To keep the augmented graph semantically close to the original graph, a backward pass of knowledge distillation is achieved by the proposed graph alignment.

because it is obtained from a distinct knowledge source and established based on affinity captured by the learning component of our model.

More specifically, our framework takes $\mathcal{KG}$, and an external source of texts, $\mathcal{T}$, as inputs, and generates an augmented knowledge graph, $a\mathcal{KG}$. in generating $a\mathcal{KG}$ we are mindful of semantic and structural similarities among $\mathcal{KG}$ entities, and we make sure it contains all the original entities of $\mathcal{KG}$. This ensures that there are common nodes in two graphs which facilitates the alignment process. To align $\mathcal{KG}$ and $a\mathcal{KG}$ in the embedding space, a novel multi-criteria objective function is devised. In particular, we design a cost function that minimizes the distance between the embeddings of the two graphs. As a result, textual nodes (e.g., blue nodes in Figure 2.1) related to each target entity are rewarded while unrelated ones get penalized in a negative sampling setting.

Extensive experimental results on four benchmark datasets demonstrate that EDGE outperforms state-of-the-art models in different tasks and scenarios, including link prediction and node classification. Evaluation results also confirm the generalizability of our model. We summarize our contributions as follows: (i) We propose EDGE, a general framework to enrich knowledge graphs and node embeddings by exploiting auxiliary knowledge sources. (ii) We introduce a procedure to generate an augmented knowledge graph from exter-

nal texts, which is linked with the original knowledge graph. (iii) We propose a novel knowledge graph embedding approach that optimizes a multi-criteria objective function in an end-to-end fashion and aligns two knowledge graphs in a joint embedding space. (iv) We demonstrate the effectiveness and generalizability of EDGE by evaluating it on two tasks, namely link prediction and node classification, on four graph datasets.

2.2 Related Works

Knowledge graph embedding has been studied extensively in the literature (Bordes et al., 2013b; Sheu & Li, 2020; Z. Sun et al., 2019a; Z. Wang et al., 2014; Xian et al., 2020; Yan et al., 2020; B. Yang et al., 2015; W. Zhang et al., 2019). A large number of them deal with the heterogeneous knowledge graph, where it appears different types of edges. While in this work we consider the type of knowledge graph with only one type (i.e. connected or not) of relation, and only focus on entity embedding learning. Our work is related to graph neural networks, such as the graph convolutional networks (GCN) (Kipf & Welling, 2017) and its variants (X. Jiang et al., 2019; X. Jiang, Zhu, Ji, et al., 2020; Z. Wu et al., 2020), which learn node embeddings by feature propagation. In the following, we mainly review the most relevant works in two aspects, i.e., graph embedding learning with external text and knowledge graph construction.

2.2.1 Knowledge Graph Embedding

Learning low dimensional embeddings for entities of knowledge graphs has been studied extensively in the literature(Bordes et al., 2013b; L. Cai & Wang, 2018; Y. Lin et al., 2015; Z. Sun et al., 2019a; Z. Wang et al., 2014; W. Zhang et al., 2019). While a large number of them deal with the heterogeneous knowledge graph, where it appears different types of edges. Translate-based models such as TransE (Bordes et al., 2013b), TransH (Z. Wang et al., 2014), etc. have been successfully employed for this task. These methods also provide embeddings for relations in the knowledge graph. While in this work we only focus on entity embedding. Hence we would compare our model with similar techniques such

as GAE (Kipf & Welling, 2016b), LoNGAE (Tran, 2018), ARVGAE (S. Pan et al., 2018), and SCAT (Zou & Lerman, 2019).

Recently, B. Yang et al., 2015 introduced DistMult in which each relation is represented by a diagonal rather than a full matrix. L. Cai and Wang, 2018 proposed to use adversarial learning to iteratively learn more useful negative samples (previously negative sampling was implemented randomly). Z. Sun et al., 2019a suggested to represent entities as complex vectors and model relations as rotation in complex plane. W. Zhang et al., 2019 proposed IterE that combines embedding and rule learning for the task of graph completion. However, none of the above considered auxiliary text to improve the embeddings. In this work, we show that incorporating external text enhances the quality of embeddings.

2.2.2 Graph Embedding with External Text

The most similar line of work to ours is where an external textual source is considered to enrich the graph and learn low dimensional graph embeddings using the enriched version of the knowledge graph. For instance, (Z. Wang & Li, 2016) annotates the KG entities in text, creates a network based on entity-word co-occurrences, and then learns the enhanced KG. Similarly, (Kartsaklis et al., 2018) adds an edge (e, t) to KG per entity e based on co-occurrence and finds graph embeddings using random walks. However, there is no learning component in these approaches in constructing the new knowledge graph. And the enrichment procedure is solely based on occurrences ("hard" matching) of entities in the external text.

For graph completion task, (Malaviya et al., 2020) uses pre-trained language models to improve the representations and for Question Answering task, (H. Sun et al., 2018) extracts a sub-graph $\mathcal{G}_q$ from KG and Wikipedia, which contains the answer to the question with a high probability and apply GCN on $\mathcal{G}_q$ which is limited to a specific task. We emphasize that the main difference between our model and previous work is that we first create an augmented knowledge graph from an external source, and improve the quality of node representation by jointly mapping two graphs to an embedding space. To the best

of our knowledge, this is the first time that a learning component is incorporated to enriching knowledge graphs.

2.2.3 Knowledge Graph Construction

Knowledge graph construction methods are broadly classified into two main groups: 1) Curated approaches where facts are generated manually by experts, e.g., WordNet (Fellbaum, 1998) and UMLS (Bodenreider, 2004), or volunteers such as Wikipedia, and 2) Automated approaches where facts are extracted from semi-structured text like DBpedia (Auer et al., 2007), or unstructured text (Carlson et al., 2010). The latter approach can be defined as extracting structured information from unstructured text.

One important component of information extraction is relation extraction in which a relation is extracted for a pair of entities from an entity annotated sentence. Mintz et al., 2009 introduced a distant supervised method for relation extraction, which assumes if two entities have a relationship, then all sentences mentioning the entities express the same relation. Thus, it can generate large amounts of training instances for a supervised task. Following (Mintz et al., 2009), PCNN was introduced in (Zeng et al., 2015) that employs convolutional neural networks to learn representations of instances. Attention mechanism and side information are used by (Y. Lin et al., 2016) and (Vashishth et al., 2018) respectively to further improve the representations.

In this work, we do not intend to construct a knowledge base from scratch, instead we aim to generate an augmented knowledge graph using side information. Hence, we employ existing tools to acquire a set of new facts from external text and link them to an existing KG.

2.3 Methodology

2.3.1 Problem Statement

We formulate the knowledge graph enrichment and embedding problem as follows: given a knowledge graph $\mathcal{KG} = (\mathcal{E}, \mathcal{R}, X)$ with $|\mathcal{E}|$ nodes (or entities),

Table 2.1: We employ representation learning algorithms to find a set of semantically and structurally similar entities to each target entity (column 2). We then find a set of keywords, K, that are representative of the target entity (column 3) and use them to query an external text and obtain a set of sentences, S (column 4). Finally, we extract textual entities (column 5), and connect them to the target entity.

Target Entity		semantically and structurally Similar Entities	Most definitive keywords	Sentences obtained from auxiliary text	Entities obtained from information extraction
Nonvenomous insect bite of hip without infection	semantic	1. Nonvenomous insect bite of foot with infection 2. Crushing injury of hip and/or thigh 3. Superficial injury of lip with infection 4. Infected insect bite of hand	1. bite 2. insect 3. nonvenomous 4. infect	1. a wound resulting from biting by an animal or a person 2. small air-breathing arthropod 3. not producing or resulting from poison 4. contaminate with a disease or microorganism	1. wound 2. arthropod 3. poison 4. microorganism
	structural	1. Insect bite, nonvenomous, of back 2. Tick bite 3. Animal bite of calf 4. Inset bite, nonvenomous, of foot and toe			
Insect bite, nonvenomous, of foot and toe infected	semantic	1. Insect bite, nonvenomous, of lower limb, infected 2. Infected insect bite of hand 3. Insect bite, nonvenomous, of hip 4. Insect bite granuloma	1. bite 2. insect 3. lower 4. skin	1. a wound resulting from biting by an animal or a person 2. small air-breathing arthropod 3. move something or somebody to a lower position 4. a natural protective body	1. wound 2. arthropod 3. position 4. protective body
	structural	1. Nonvenomous insect bite of hip without infection 2. Insect bite, nonvenomous, of back 3. Recurrent infection of skin 4. Skin structure of lower leg			

$|\mathcal{R}|$ edges (or relations) and $X \in \mathbb{R}^{|\mathcal{E}| \times D}$ as feature matrix, where D is the number of features per entity, also given an external textual source, $\mathcal{T}$, the goal is to generate an augmented knowledge graph and jointly learn d ($d << |\mathcal{E}|$) dimensional embeddings for knowledge graph entities, which preserve structural and semantic properties of the knowledge graph. The learned representations are then used for the tasks of link prediction and node classification. Link prediction is defined as a binary classification whose goal is to predict whether or not an edge exists in KG, and node classification is the task of determining node labels in labelled graphs.

To address the problem of knowledge graph enrichment and embedding, we propose EDGE, a framework that contains two major components, i.e., augmented knowledge graph construction, and knowledge graph alignment in a joint embedding space.

2.3.2 Augmented Knowledge Graph Construction

Given the entities of $\mathcal{KG}$ and an external source of textual data, $\mathcal{T}$, we aim to generate an augmented graph, $a\mathcal{KG}$, which is a supergraph of $\mathcal{KG}$ (i.e., $\mathcal{KG}$ is a subgraph of $a\mathcal{KG}$). Augmentation is the process of adding new entities to $\mathcal{KG}$. These newly added entities are called *textual entities* or *textual nodes*. A crucial

property of $a\mathcal{KG}$ is that it contains entities of $\mathcal{KG}$. The presence of these entities establishes a relationship between the two graphs, and such a relationship will be leveraged to learn the shared graph embeddings. To construct $a\mathcal{KG}$, we need to find a set of keywords to query an external source, To obtain high quality keywords and acquire new textual entities, we design the following procedure per target entity e_t (For every step of this process refer to Table 2.1 for a real example from SNOMED dataset).

First, we find a set of semantically and structurally similar entities to e_t denoted by $\mathcal{E}_{e_t}$. This set creates a textual context around e_t which we use to find keywords to query an external text, e.g., WordNet or Wikipedia. Here by *query* we mean using the API of the external text to find related sentences, S (for instance for a given keyword "bite" we can capture several sentences from the wikipedia page for the entry "biting" or find several Synsets[2] from WordNet when we search for "bite").

[2] Synset is the fundamental building block of WordNet which is accompanied by a definition, example(s), etc.

Finally, we extract entities from S and attach them to e_t. We call these new entities, *textual entities* or *textual features*. By connecting these newly found textual entities to the e_t, we enhance $\mathcal{KG}$ and generate the augmented knowledge graph, $a\mathcal{KG}$. We observed that the new textual entities are different from our initial feature space. Also, it is possible that two different target entities share one or more textual nodes, hence the distance between them in $a\mathcal{KG}$ would decrease. The implementation details of this process is provided in Supplementary materials.

Querying an external text allows us to extend the feature space beyond the context around e_t. By finding other entities in $\mathcal{KG}$ that are similar to the target entity and extracting keywords from the collection of them to query the external text, distant entities that are related but not connected would become closer to each other owing to the shared keywords.

Figure 2.1 illustrates a subset of SNOMED graph and its augmented counterpart by following the above procedure. As this figure reveals, the structure of $a\mathcal{KG}$ is different from $\mathcal{KG}$, and as a result of added textual nodes, distant but similar entities would become closer. Therefore, augmenting knowledge

graphs would alleviate the KG sparsity issue. Although we may introduce noise by adding new entities but later in the alignment process we address this issue.

Remarks. In the above procedure, we need to obtain similar entities before looking for textual entities, and the rationality of such a strategy is discussed as follows. One naive approach is to simply use keywords included in the target entity to find new textual features. In this way, we would end up with textual features that are related to that target entity, but we cannot extend the feature space to capture similarity (i.e., dependency) among entities.

2.3.3 Knowledge Graph Alignment in Joint Embedding Space

With the help of augmented knowledge graph $a\mathcal{KG}$, we aim to enrich the graph embeddings of $\mathcal{KG}$. However, inevitably, a portion of newly added entities are noisy, and even potentially wrong. To mitigate this issue, we are inspired by Hinton et al. (Hinton et al., 2015), and propose a graph alignment process for knowledge distillation. In fact, $a\mathcal{KG}$ and $\mathcal{KG}$ share some common entities, which makes it possible to map two knowledge graphs into a joint embedding space. In particular, we propose to extract low-dimensional node embeddings of two knowledge graphs using graph auto-encoders (Kipf & Welling, 2016b), and design novel constraints to align two graphs in the embedding space. The architecture of our approach is illustrated in Figure 5.1.

Let $\mathbf{A}_K$ and $\mathbf{A}_T$ denote the adjacency matrices of $\mathcal{KG}$ and $a\mathcal{KG}$, respectively. The loss functions of graph auto-encoders that reconstruct knowledge graphs are defined as:

$$\mathcal{L}_K = \min_{\mathbf{Z}_K} ||\mathbf{A}_K - \hat{\mathbf{A}}_K||_2, \tag{2.1}$$

$$\mathcal{L}_T = \min_{\mathbf{Z}_T} ||\mathbf{A}_T - \hat{\mathbf{A}}_T||_2, \tag{2.2}$$

where $\hat{\mathbf{A}}_K = \sigma(\mathbf{Z}_K \mathbf{Z}_K^\top)$ is the reconstructed graph using node embeddings $\mathbf{Z}_K$. And $\mathbf{Z}_K$ is the output of graph encoder that is implemented by a two-layer GCN Kipf and Welling, 2016b:

$$\mathbf{Z}_K = \mathrm{GCN}(\mathbf{A}_K, \mathbf{X}_K) = \tilde{\mathbf{A}_K} \tanh(\tilde{\mathbf{A}}_K \mathbf{X}_K \mathbf{W}_0)\mathbf{W}_1, \tag{2.3}$$

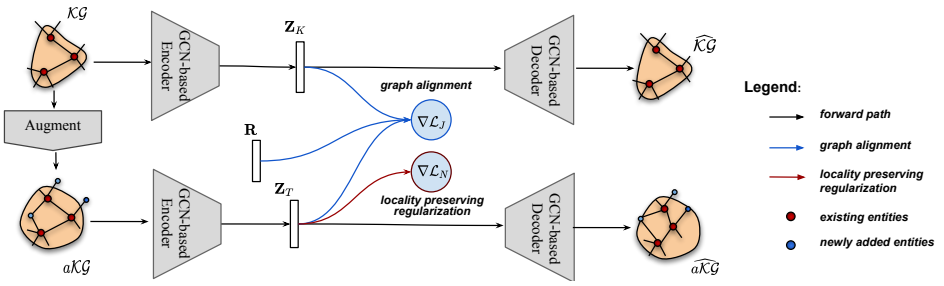


Figure 2.2: Our proposed framework for aligning two graphs in the embedding space. The graph alignment component, $\mathcal{L}_J$, requires an additional matrix, $\mathbf{R}$, that selects embeddings of $\mathcal{KG}$ entities from $\mathbf{Z}_T$, so the resulting matrix, $\mathbf{RZ}_T$, would have the same dimension as $\mathbf{Z}_K$. Furthermore, $\mathcal{L}_N$ penalizes additional entities that are unrelated to the target entity, while rewards the related ones. We omit the graph reconstruction loss for simplicity.

where $\tilde{\mathbf{A}}_K = \mathbf{D}_K^{-\frac{1}{2}}\mathbf{A}_K\mathbf{D}_K^{-\frac{1}{2}}$. $\mathbf{D}_K$ is the degree matrix, $\tanh(.)$ is the Hyperbolic Tangent function that acts as the activation function of the neurons, $\mathbf{W}_i$ are the model parameters, and $\mathbf{X}_K$ is the feature matrix.[3] Similarly, $\hat{\mathbf{A}}_T = \sigma(\mathbf{Z}_T\mathbf{Z}_T^\top)$, and $\mathbf{Z}_T$ is learned by another two-layer GCN. Equations (2.1) and (2.2) are l_2-*norm* based loss functions that aim to minimize the distance between original graphs and the reconstructed graphs.

[3] In case of a featureless graph, an identity matrix, $\mathbf{I}$, replaces $\mathbf{X}_K$.

Furthermore, to map $\mathcal{KG}$ and $a\mathcal{KG}$ to a joint embedding space and align their embeddings through common entities, we define the following graph alignment loss function:

$$\mathcal{L}_J = ||\mathbf{Z}_K - \mathbf{RZ}_T||_2, \tag{2.4}$$

where $\mathbf{R}$ is a transform matrix that selects common entities that exist in $\mathcal{KG}$ and $a\mathcal{KG}$. Note that the two terms $\mathbf{Z}_K$ and $\mathbf{RZ}_T$ should be of the same size in the L_2 norm equation. Our motivation is to align the embeddings of common entities across two knowledge graphs. By using $\mathbf{R}$, the node embeddings of common entities can be selected from $\mathbf{Z}_T$. Note that $\mathbf{Z}_T$ is always larger than $\mathbf{Z}_K$, as $\mathcal{KG}$ is a subgraph of $a\mathcal{KG}$. Equation (2.4) also helps preserve local structures of the original knowledge graph $\mathcal{KG}$ in the graph embedding space. In other

Algorithm 1 Training process of EDGE

Input: $\mathbf{A}_K, \mathbf{X}_K, \mathbf{A}_T, \mathbf{X}_T$, POS, NEG,
Input: $\mathbf{R} \in \mathbb{R}^{|\mathcal{E}_K| \times (|\mathcal{E}_T| - |\mathcal{E}_K|)}$

for each epoch **do**
  $\hat{\mathbf{A}}_K = \sigma(\mathbf{Z}_K \mathbf{Z}_K^\top)$
  $\mathbf{Z}_K = \tilde{\mathbf{A}}_K \tanh(\tilde{\mathbf{A}}_K \mathbf{X}_K \mathbf{W}_0^K) \mathbf{W}_1^K$
  $\hat{\mathbf{A}}_T = \sigma(\mathbf{Z}_T \mathbf{Z}_T^\top)$
  $\mathbf{Z}_T = \tilde{\mathbf{A}}_T \tanh(\tilde{\mathbf{A}}_T \mathbf{X}_T \mathbf{W}_0^T) \mathbf{W}_1^T$
  Calculate $\mathcal{L}_K$ and $\mathcal{L}_T$ using Equations (2.1) and (2.2).
  Compute $\mathcal{L}_J$ using Equation (2.4)
  Find negative and positive samples and calculate $\mathcal{L}_N$ using Equation (2.5)
  Sum up all losses with their corresponding ratios using Equation (2.6)
  Run Adam optimizer to minimize $\mathcal{L}$
  Update model parameters $\mathbf{W}_i^K$ and $\mathbf{W}_i^T$
end for

Output: Z_K

words, nodes that are close to each other in the original knowledge graph will be neighbors in the augmented graph as well.

Moreover, we notice that the proposed augmented knowledge graph $a\mathcal{KG}$ involves more complicated structures than the original knowledge graph $\mathcal{KG}$, due to the newly added textual nodes for each target entity in $\mathcal{KG}$. In $a\mathcal{KG}$, one target entity is closely connected to its textual nodes, and their embeddings should be very close to each other in the graph embedding space. However, such local structures might be distorted in the graph embedding space. Without proper constraints, it is possible that one target entity is close to textual entities of other target entities in the embedding space, which is undesired for downstream applications. To address this issue, we design a margin-based loss function with negative sampling to preserve the locality relationship as follows:

$$\mathcal{L}_N = -\log(\sigma(\mathbf{z}_e^\top \mathbf{z}_t)) - \log(\sigma(-\mathbf{z}_e^\top \mathbf{z}_{t'})), \tag{2.5}$$

where $\mathbf{z}_t$ are the embeddings of the related textual nodes, $\mathbf{z}'_t$ are the embeddings of textual nodes that are not related to the target entity, and σ is the *sigmoid* function.

Table 2.2: Link prediction results for SNOMED and three citation networks. Numbers for SNOMED are obtained from rerunning their code on the dataset. The rest of the results are reported from corresponding papers.

Model	SNOMED		Cora		Citeseer		PubMed	
	AUC	AP	AUC	AP	AUC	AP	AUC	AP
GAE Kipf and Welling, 2016b	0.773	0.844	0.914	0.926	0.908	0.920	0.964	0.965
LoNGAE Tran, 2018	0.890	0.910	0.954	0.963	0.953	0.961	0.960	0.963
ARVGE S. Pan et al., 2018	0.805	0.864	0.924	0.926	0.924	0.930	0.968	0.971
SCAT Zou and Lerman, 2019	0.902	0.918	0.945	0.946	0.973	**0.976**	**0.975**	**0.972**
GIC Mavromatis and Karypis, 2020	-	-	0.935	0.933	0.970	0.968	0.937	0.935
EDGE (This work)	**0.916**	**0.944**	**0.973**	**0.975**	**0.974**	**0.976**	0.969	0.968

Finally, the overall loss function is defined as:

$$\mathcal{L} = \min_{\mathbf{Z}_K, \mathbf{Z}_T} \underbrace{\mathcal{L}_K + \alpha \mathcal{L}_T}_{\substack{\text{reconstruction} \\ \text{loss}}} + \underbrace{\beta \mathcal{L}_J}_{\substack{\text{graph} \\ \text{alignment}}} + \underbrace{\gamma \mathcal{L}_N}_{\substack{\text{locality} \\ \text{preserving}}}, \tag{2.6}$$

where α, β, and γ are hyper-parameters. We perform full-batch gradient descent using the Adam optimizer to learn all the model parameters in an end-to-end fashion. The whole training process of our approach is summarized in Algorithm 1.

The learned low-dimensional node embeddings $\mathbf{Z}_K$ could benefit a number of unsupervised and supervised downstream applications, such as link prediction and node classification. Link prediction is the task of inferring missing links in a graph, and node classification is the task of predicting labels to vertices of a (partially) labeled graph. Extensive evaluations on both tasks will be provided in the experiment section.

2.3.4 Model Discussions

We have proposed a general framework for graph enrichment and embedding by exploiting auxiliary knowledge sources. What we consider as a source of knowledge is a textual knowledge base that can provide additional information about the entities of the original knowledge graph. It is a secondary source of

knowledge that supplies new sets of features outside of the existing feature space, which improves the quality of representations.

The proposed graph alignment approach can fully exploit augmented knowledge graph and thus improve the graph embeddings. Although $a\mathcal{KG}$ is a supergraph of $\mathcal{KG}$, its connectivity pattern is different. With the help of our customized loss function for graph alignment, both graphs contribute in the quality of derived embeddings. We will also demonstrate the superiority of our joint embedding approach over the independent graph embedding approach (with only $a\mathcal{KG}$) in the experiments, and we investigate which component of our model contributes more in the final performance in the ablation study in Subsection 2.4.4.

2.4 Experiments

We design our experiments to investigate effectiveness of different components of EDGE as well as its overall performance. To this end, we aim to answer the following three questions.

- $Q1$ How well does EDGE perform compared to state-of-the-art in the task of link prediction? (Section 2.4.1)
- $Q2$ How is the quality of embeddings generated by EDGE compared to similar methods? (Sections 2.4.2 and 2.4.3)
- $Q3$ What is the contribution of each component (augmentation and alignment) in the overall performance? (Section 2.4.4)

2.4.1 Task 1: Link Prediction

To investigate Q1 we perform link prediction on four benchmark datasets, and compare the performance of our model with five relevant baselines. For this task we consider SNOMED and three citation networks. For SNOMED, similar to (Kartsaklis et al., 2018), we select 21K medical concepts from the original dataset. Each entity in SNOMED is a text description of a medical concept, e.g., *Nonvenomous insect bite of hip without infection*. According to the procedure

explained in subsection 2.3.2, we construct an augmented knowledge graph, $a\mathcal{KG}$. Additionally, we consider three other datasets, namely Cora, Citeseer, and PubMed, which are citation networks consisting of 2,708, 3,312, and 19,717 papers, respectively. In all three datasets, a short text accompanies each node which is extracted from the title or abstract of the paper. For these networks, *relation* is defined as citation and the textual content of the nodes enables us to obtain $a\mathcal{KG}$. Cora and Citeseer datasets come with a set of default features. We defer the detailed description of datasets in the supplementary.

In this experiment, for each dataset, we train the model on 85% of the input graph. Other 15% of the data is split into 5% validation set and 10% as part of the test set (positive samples only). An additional set of edges are produced, equal to the number of positive samples, which does not exist in the graph, as negative samples. The union of positive and negative samples are used as the test set. In all baselines, we test the model on $\mathcal{KG}$. We obtain the following values for loss ratios after hyper-parameter tuning: $\alpha = 0.001$, $\beta = 10$, $\gamma = 1$. We discuss parameter tuning and explain the small value of α in Section 2.4.5.

We provide comparison against VGAE (Kipf & Welling, 2016b) and its adversarial variant ARVGE (S. Pan et al., 2018). Also we consider LoNGAE (Tran, 2018), SCAT (Zou & Lerman, 2019) and GIC (Mavromatis & Karypis, 2020) which are designed for link prediction task on graphs, hence they make strong baselines. Table 2.2 presents the Area Under the ROC Curve (AUC) and average precision (AP) scores for five baselines and our methods across all datasets. We observe that EDGE outperforms all baselines in three out of four datasets and produces comparable results for PubMed dataset.

2.4.2 Task 2: Node Classification on Citation Networks

To evaluate the quality of embeddings (Q2) we design a node classification task based on the final product of our model. For this task, we use Cora, Citeseer and PubMed datasets, and follow the same procedure explained in 2.3.2 to generate $a\mathcal{KG}$ and jointly map the two graphs into an embedding space. All the settings are identical to Task 1. To perform node classification, we use the final product of our model, which is a 160 dimensional vector per node. We train a linear

Table 2.3: Node classification results in terms of accuracy for citation networks. TR stands for training ratio and *un.* and *semi.* are short for unsupervised and semi-supervised.

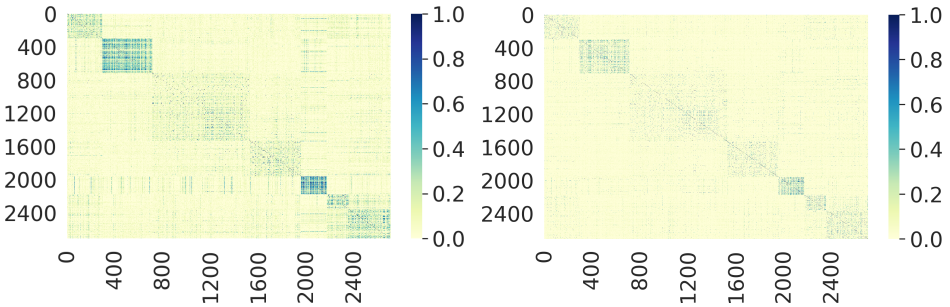
Model	Approach	Cora TR=0.5	Citeseer TR=0.03	PubMed TR=0.003
DeepWalk	*un.*	0.67	0.43	0.65
GCN	*semi.*	0.81	0.70	0.79
GAT	*semi.*	**0.83**	**0.72**	0.79
LoNGAE	*semi.*	0.78	0.71	0.79
MixHop	*semi.*	0.82	0.71	**0.81**
EDGE	*un.*	0.81	0.66	0.76

SVM classifier and obtain the accuracy measure to compare the performance of our model with state-of-the-art methods. Training ratio varies across different datasets, and we consider several baselines to compare our results against.

We compare our approach with state-of-the-art semi-supervised models for node classification, including GCN (Kipf & Welling, 2017), GAT (Veličković et al., 2018a), LoNGAE (Tran, 2018), and MixHop (Abu-El-Haija et al., 2019). These models are semi-supervised, thus they were exposed to node labels during training while our approach is completely unsupervised. We also include DeepWalk, an unsupervised approach, to have a more complete view for our comparison. Table 2.3 reveals that our model achieves reasonable performance compared with semi-supervised models in two out of three datasets. Since EDGE is fully unsupervised, it is fair to declare that its performance is comparable as other methods are exposed to more information (i.e., node labels).

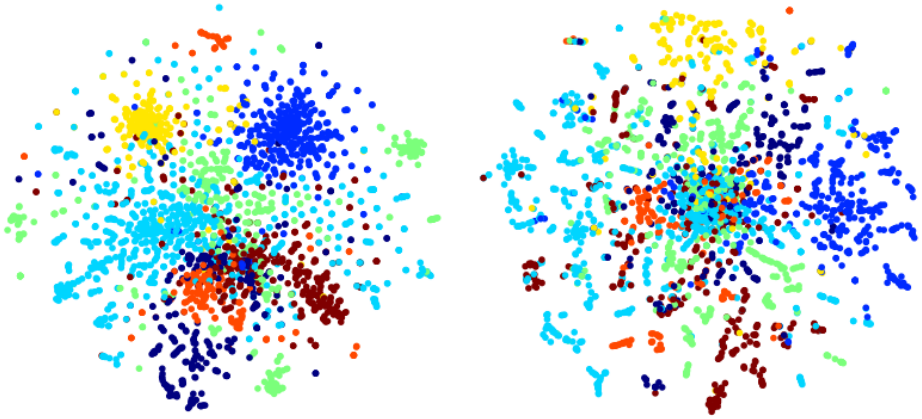
2.4.3 Embedding Effectiveness

Further, to measure the quality of embeddings produced by our model and compare it against the baseline, we visualize the similarity matrix of node embeddings for two scenarios on the Cora dataset: 1) GAE on $\mathcal{KG}$, and 2) EDGE on $\mathcal{KG}$ and $a\mathcal{KG}$. The results are illustrated in Figure 2.3. In this heatmap, elements are pair-wise similarity values sorted by different labels (7 classes). We



(a) EDGE (b) GAE

Figure 2.3: Pair-wise similarity comparison between GAE and EDGE.



(a) feature-based (b) featureless

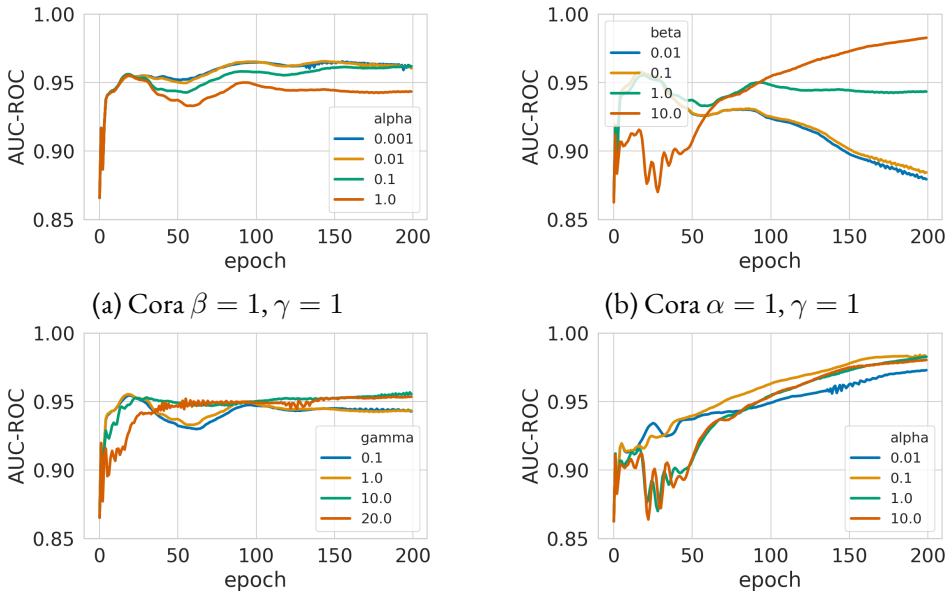
Figure 2.4: Visualization of embedding vectors on Cora: a) with and b) without features to study the effect of features on quality of embeddings in node classification.

can observe that the block-diagonal structure learned by our approach is clearer than that of GAE, indicating enhanced separability between different classes.

Next, we examine our model in more details and study how different parameters affect its performance.

2.4.4 Ablation Study

To investigate the effectiveness of different modules of our model (Q3), we consider two scenarios. First we use a single graph to train our model. Note that when we use a single graph, the graph alignment and locality preserving losses



(a) Cora $\beta = 1, \gamma = 1$

(b) Cora $\alpha = 1, \gamma = 1$

(c) Cora $\alpha = 1, \beta = 1$

(d) Cora $\beta = 10, \gamma = 1$

Figure 2.5: Effect of parameterization on link prediction performance

are discarded and our model is reduced to GAE. In single graph scenario we consider two versions of augmented graph, $a\mathcal{KG}$ that was explained in subsection 2.3.2 and $a\mathcal{KG}^*$ that was created based on co-occurrence proposed by (Kartsaklis et al., 2018). In the second scenario, we use two graphs to jointly train EDGE, and we feed our model with $\mathcal{KG} + a\mathcal{KG}^*$ and $\mathcal{KG} + a\mathcal{KG}$ to show the effect of augmentation.

For link prediction we only consider SNOMED dataset which is the largest dataset, and as Table 2.4 presents we observe that our augmentation process is slightly more effective than co-occurrence based augmentation. More importantly, by comparing second two rows with first two rows we realize that alignment module improves the performance more than augmentation process which highlights the importance of our proposed joint learning method. Moreover, we repeat this exercise for node classification (see Table 2.5) which results in a similar trend across all datasets.

Finally, we plot the t-SNE visualization of embedding vectors of our model with and without features. Figure 2.4 clearly illustrates the distinction between

Table 2.4: Link prediction results for SNOMED dataset. In this table $a\mathcal{KG}^*$ is the augmented knowledge graph generated using the method explained in (Kartsaklis et al., 2018)

Input	Model	AUC	AP
$\mathcal{KG}$	GAE (Kipf & Welling, 2016b)	0.77	0.84
$a\mathcal{KG}^*$	GAE (Kipf & Welling, 2016b)	0.85	0.88
$a\mathcal{KG}$	GAE (Kipf & Welling, 2016b)	0.86	0.90
$\mathcal{KG} + a\mathcal{KG}^*$	EDGE (This work)	0.90	0.93
$\mathcal{KG} + a\mathcal{KG}$	EDGE (This work)	**0.91**	**0.94**

quality of the clusters for the two approaches. This implies that knowledge graph text carries useful information. When the text is incorporated into the model, it can help improve the model performance.

2.4.5 Parameter Sensitivity

We evaluate the parameterization of EDGE, and specifically we examine how changes to hyper parameters of our loss function (i.e., α, β and γ) could affect the model performance in the task of link prediction on Cora dataset. In each analysis, we fix the values of two out of three parameters and study the effect of the variation of the third parameter on evaluating AUC scores across 200 epochs. The detailed results are shown in Figure 2.5.

Figure 2.5a shows the effect of varying α, when $\beta = 1$ and $\gamma = 1$ are fixed. We observe a somewhat consistent trend across performance for different values of α. It is evident that decreasing α improves the performance. α is the coefficient of $\mathcal{L}_T$ (see Equation 2.2). This examination suggests that the effect of this loss function is less significant, because we re-address it in the $\mathcal{L}_N$ part of the loss function, where we consider the same graph ($a\mathcal{KG}$) and try to optimize distance between its nodes but with more constraints.

Figure 2.5b illustrates the effect of varying β, while $\alpha = 1$ and $\gamma = 1$ are fixed. Tuning β results in more radical changes in the model performance, which is again consistent between the two datasets. Small values for β degrades

Table 2.5: Node classification results in terms of accuracy for citation networks. TR stands for training ratio and $a\mathcal{KG}^*$ is an augmented knowledge graph produced by the method proposed in (Kartsaklis et al., 2018)

Input	Model	Cora TR=0.5	Citeseer TR=0.03	PubMed TR=0.003
$\mathcal{KG}$	GAE	0.62	0.51	0.60
$a\mathcal{KG}^*$	GAE	0.70	0.57	0.65
$a\mathcal{KG}$	GAE	0.75	0.61	0.67
$\mathcal{KG} + a\mathcal{KG}^*$	EDGE	0.80	0.64	0.73
$\mathcal{KG} + a\mathcal{KG}$	EDGE	**0.81**	**0.66**	**0.76**

performance remarkably, and we observe a much more improved AUC score for larger values of β. This implies the dominant effect of the joint loss function, $\mathcal{L}_J$, which is defined as the distance between corresponding entities of $\mathcal{KG}$ and $a\mathcal{KG}$.

Next, we fix $\alpha = 1$ and $\beta = 1$ and tweak γ from 0.1 to 10. As Figure 2.5c reveals, the variation in performance is very small. Finally, as we obtained the best results when $\beta = 10$, we set $\gamma = 1$ and once again tune α. Figure 2.5d shows the results for this updated setting. These experiments confirm the insignificance of parameter α. In practice, we obtained the best results by setting α to 0.001.

2.5 Summary

Sparsity is a major challenge in KG embedding, and many studies failed to properly address this issue. We proposed EDGE, a novel framework to enrich KG and align the enriched version with the original one with the help of auxiliary text. Using external source of information introduces new sets of features that enhance the quality of embeddings. We applied our model on three citation networks and one large scale medical knowledge graph. Experimental results show that our approach outperforms existing graph embedding methods on link prediction and node classification.

We demonstrated the effectiveness of incorporating augmented knowledge graph in the learning process by comparing pair-wise distance among entities before and after adding textual nodes, and by visualizing and comparing the pair-wise similarity of embeddings of our model and GAE. The results of the two analysis suggest that including additional information in the augmented knowledge graph helps learn better embeddings for the KG.

Our model is also flexible such that many other graph embedding models could be employed to generate low dimensional embeddings. In this work, we used graph convolutional networks to produce embeddings because of its recent success and widespread implementations. One issue with GCN is that it loads the whole graph into the physical memory and therefore is not memory efficient and cannot handle very large graphs. We will explore the issue of efficiency in our future work.

Chapter 3

Knowledge Graph as External Knowledge I

Search engines for domain-specific media collections often rely on rich metadata being available for the content items. The annotations may not be complete or rich enough to support an adequate retrieval effectiveness. As a result, some search queries receive only a small result set (low recall) and others might suffer from reduced relevance (low precision). To alleviate this, we present a framework that exploits external knowledge to provide entity-oriented reformulation suggestions for queries that contain entities. We propose that queries be added as *surrogate nodes* to an external Knowledge Graph (KG) via the use of state-of-the-art entity linking algorithms. Embedding methods are invoked on the augmented graph, which contains additional edges between surrogate nodes and KG entities. We introduce a new evaluation setting to evaluate the quality of these embeddings. Experimental results on seven datasets confirm the effectiveness of the approach.

3.1 Introduction

Knowledge Graphs (KGs) organize information and objects into a graph structure in which entities (i.e., nodes) can be traversed through relations (i.e., edges), and every traversal represents a fact which consists of two entities (head and tail)

and a relation between them (i.e., a `(head, relation, tail)` triple). In this manner, a KG is a formal representation of abstract concepts and real world objects. For example the word "bass" can refer to a type of fish or describe tones of low in music. However, they are two different entities in the KG and have different local structure around them which might help disambiguate different senses of the word. The construction of a knowledge graph is a well-developed research area in its own right, and the availability of large and general purpose KGs have fuelled their popularity across a range of application settings (Ji et al., 2021).

Knowledge graphs are sparse data structures, meaning that the number of facts per entity is very small. For example, one of the largest real world knowledge graphs, Freebase, has a fact-to-entity ratio of 16 (Pujara et al., 2017). The graph structure of a KG enables reasoning over the individual *facts*. One such task is that of Link Prediction (LP) (Benson et al., 2018; Duan et al., 2017; Kazemi & Poole, 2018)—inferring new relations among entities given a snapshot of a graph. LP has many applications from friendship suggestion in social networks (Ahmad et al., 2010; Aiello et al., 2012; Al Hasan & Zaki, 2011) to predicting associations between molecules in biological networks (J. Jiang et al., 2020).

In this chapter, we consider the problem of *query suggestions* within an information retrieval setting and frame it as an application of link prediction. Here, the user has provided the search engine with a short keyword query. The user has examined the returned results and our objective is to provide reformulation assistance so that the refined query is more likely to return relevant results ranked high. Typical algorithms for this task rely on historical data, where associations between queries and their constituent words can be mined, which might work for common queries for which there is plentiful data.

We propose a knowledge-aware query suggestion method that leverages an existing KG to surface entities related to those present in the query. In an entity-oriented search, we expect to rank related entities (nodes in the KG) using a link prediction model where the head is an entity present in the query and the tail entities are candidates from the KG. This ranking can further be restricted to

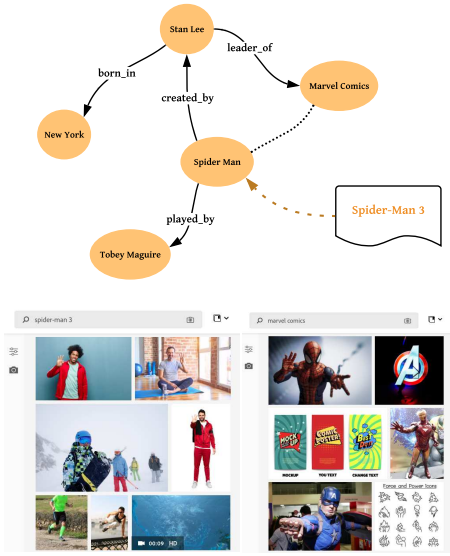


Figure 3.1: An example illustrating the proposed pipeline. In this scenario, the search engine (Adobe Stock) yields small number of results for the query "Spider-Man 3". Our model suggests a new query based on an EL/LP pipeline. The suggested query, "Marvel Comics", provides search results with higher recall.

selected relation types. This process requires an Entity Linking (EL) method that allows the linking of mentions in the query to the entities of the underlying knowledge graph.

Despite advances in entity linking methods, there are inherent problems in the use of these algorithms in the current setting. Search queries tend to be short keyword based phrases, allowing little or no context for the EL algorithm to leverage. Given the earlier example, it is almost impossible to distinguish between different instances of "bass" if there is no context around them. As a result there is a significant chance of error in the entity linking which can propagate through the link prediction phase and produce bad candidates. Knowledge graphs can be useful in addressing this challenge. The information within the graph structure is leveraged by KG embedding methods (Q. Wang et al., 2017) that we expect would be useful in disambiguating alternate interpretations of words/entities in the query.

In this chapter, we propose a framework that leverages state-of-the-art entity linking and KG embedding methods to help identify query reformulation candidates. We introduce the notion of *surrogate nodes* which are nodes corresponding to queries added into the KG. For each mention in the query (an entity), we add a new node to the KG and connect it to its related entities (output of the EL algorithm) already present in the KG. The inclusion of surrogate nodes improves the semantics of the KG by adding different senses of a query. It also offers a mechanism to grow/refine the KG based on emerging/evolving set of entities present in user search queries.

We then perform LP on the enhanced KG (based on similarity in embedding space, see Algorithm 2), thereby providing related entities that are interpreted as query suggestions. We evaluate this new problem setting, an EL-LP pipeline, by the construction of a new benchmark and design of relevant metrics to measure the performance of our approach against the baseline. Additionally, we undertake a query suggestion study which demonstrates how providing additional knowledge to the KG can disambiguate the user intent. In Figure 3.1, the user may not know that in order to get information for "Spider-Man 3", the keyword "Marvel Comics" can be used. In this sense our goal is to assist the user more effectively benefit from the search experience.

We summarize our contributions as follows:

1. We propose the framing of knowledge-aware query suggestion as a link prediction task. The key idea of our framework is to add surrogate nodes which represent query entities to an existing knowledge graph.

2. We propose a framework that tolerates imperfect entity linking but still identifies relevant related entities to those present in the query.

3. We conduct a thorough evaluation of this new problem setting, including a range of metrics, to illustrate the utility of our method.

4. We conduct a case study to qualitatively demonstrate the effectiveness of our approach.

3.2 Related Work

3.2.1 Query Reformulation

The problem of query suggestion has been widely studied recently J.-Y. Jiang and Wang, 2018; L. Li et al., 2017; R. Li et al., 2019; Zhong et al., 2020. Most of the work on query suggestion leverages user log data such as their clicks L. Li et al., 2017; R. Li et al., 2019; Zhong et al., 2020. Most of these works focus on using only user log data **empty citation** as opposed to knowledge graph embeddings as done in our work. In contrast, our work focuses on leveraging an external knowledge graph embedding. and studies for query reformulation. Recently, there have been a few works that leverage knowledge graph embeddings (KGE) for tasks related to query suggestion X. Huang et al., 2019; Rosset et al., 2020; C. Shi et al., 2020. In particular, Recent work has also studied the problem of suggesting graph-queries for exploring knowledge graphs Lissandrini et al., 2020. However, these works all solve a different problem than the one we study in this chapter.

While there are some recent work on query reformulation Das et al., 2020; Hirsch et al., 2020; S.-C. Lin et al., 2020; Verma et al., 2020; X. Wang et al., 2020, none of them study the same problem as the one we study in this work. In particular, a recent paper by Hirsch et al. Hirsch et al., 2020 conducts a large-scale study investigating query reformulations by users. Another work by Wang et al. X. Wang et al., 2020 proposed a reinforcement learning approach that uses a seq2seq model trained with user query log data to perform query reformulations. None of these works leverage a knowledge graph or its embedding to improve query reformulation task.

3.2.2 Link Prediction

Link Prediction has been extensively studied in social networks (Aiello et al., 2012), web graphs (M. Zhang, Cui, et al., 2018), biological networks (J. Jiang et al., 2020), information networks (Gesese et al., 2020), and KG (Pachev & Webb, 2018). Methods have been proposed for predicting links in different

types of graphs including bipartite graphs (Kunegis et al., 2010), homogeneous graphs (Benson et al., 2018), and knowledge graphs (Kazemi & Poole, 2018; Tay, Luu, & Hui, 2017). Methods for link prediction are all essentially based on either the notion of proximity in the graph (A. Kumar et al., 2020; Shao et al., 2019; Sharma et al., 2020; M. Zhang, Wang, et al., 2018; M. Zhang & Chen, 2018) or the notion of structural similarity/roles (Ahmed et al., 2020). There have been some work on using motifs for link prediction (Benson et al., 2018; R. A. Rossi et al., 2020). Complex link prediction methods have been developed recently that leverage embeddings (Tay, Luu, & Hui, 2017) derived from graph autoencoders (Salha et al., 2020; Salha et al., 2019), graph neural networks (M. Zhang & Chen, 2018), spectral methods (Pachev & Webb, 2018; Sharma et al., 2020), among many others (Daza et al., 2020; Hao et al., 2020; R. A. Rossi et al., 2018). Other work has focused on the evaluation of different link prediction methods (Y. Yang et al., 2015). While most work has focused on transductive (within-network) link prediction (Salha et al., 2019; M. Zhang & Chen, 2018), there are some recent inductive (across-network) link prediction methods (Daza et al., 2020; Hao et al., 2020; R. A. Rossi et al., 2018).

More recently, there has been a lot of work on link prediction in knowledge graphs (Kazemi & Poole, 2018; A. Rossi et al., 2020; Rosso et al., 2020; M. Zhang, Wang, et al., 2018). Most link prediction methods for knowledge graph are based on proximity/distance in the graph, and leverage paths (M. Zhang, Wang, et al., 2018), tensor factorization (Balazevic et al., 2019), random walks (Tay, Luu, Hui, & Brauer, 2017), and other local proximity-based mechanisms (Rosso et al., 2020). There have also been some recent work that enriches the graph to improve link prediction using multilingual textual descriptions (Gesese et al., 2020). The proposed framework in this work is agnostic to the link prediction method, and can naturally leverage any state-of-the-art approach. Unlike previous methods, our proposed framework can choose from a variety of EL and LP algorithms in a plug-in manner. In addition, our method can be used when user data is sparse. It searches for queries that exist in the underlying KG, and hence does not need behavioral and retrieval data.

3.3 Proposed Method

In this section we first define the terminology used in this paper, formally, then describe the problem we are trying to solve, and finally explain how we address this problem.

3.3.1 Preliminary Definition

Knowledge Graph: Let $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$ be the knowledge graph, where $\mathcal{E}$ is a set of textual entities, $\mathcal{R}$ is a set of textual relations, and $\mathcal{T}$ is a set of triplets in the form of (h, r, t), where $h, t \in \mathcal{E}$ are *head* and *tail* entities and $r \in \mathcal{R}$ is a relation between two entities.

Entity Linking: Let $Q = \{q_1, \cdots, q_n\}$ be a set of textual queries. Every query q is defined as a sequence of words $q = (w_1 \cdots w_v)$. Every subsequence of words in q that represents an entity e is called entity mention and denoted by m if $e \in \mathcal{E}$. The process of mapping mention m to entity e is called Entity Linking, EL : $m \mapsto e$.

3.3.2 Problem Definition

Knowledge-Derived Query Suggestion: The standard query suggestion task is considered as a ranking problem where a set of candidate suggested queries are ranked and provided to the user given an initial query, q, and a scoring function. Existing methods take q_i and predict the next query $P(q_{i+1}|q_i)$ using a sequence-to-sequence model. These methods cannot handle out-of-vocabulary words and have unpredictable behavior for rare combinations. We redefine the task of query suggestion such that it would not require user data (historical or behavioral). The central idea is that KG entities are meaningful suggestions because they are curated objects whose coverage are not limited based on word popularity. To incorporate the KG we use an entity linking algorithm which assigns a set of KG entities to the query and use these linked entities to find relevant suggestions for the initial query.

In summary, given the above setting, a knowledge graph $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$, and a query $q \in Q$, the goal is to return a ranked list of *relevant* entities, $\langle e_1, e_2, \cdots \rangle$, where $e_i \in \mathcal{E}$ and relevance is inferred based on the distance in the embedding space.

3.3.3 Proposed Model

Our model incorporates a state-of-the-art EL algorithm which allows us to map queries to entities in the underlying KG. Knowledge-derived query suggestion relies on properly identifying the entities in the query. With a perfect EL system, we can annotate the mentions in the query with known entities and the task is concluded. However, since queries are short with little or no context, EL algorithms fall short in this disambiguation task. This makes the task of query reformulation challenging and motivates our method. In what follows we explain how imperfect EL can address these shortcomings.

First, we employ existing EL algorithms to map mentions in queries to multiple KG entities. We consider several linked entities because the EL method may make incorrect predictions and considering multiple entities improves the recall of our model while maintaining good precision. Further, the output of most entity linking methods are accompanied by confidence scores, and we can use these scores to weight edges when we later connect them to the knowledge graph entities. Next, for each mention m in the query set, we add a new node (i.e., an entity) e_m to the KG, which we refer to as a surrogate node. We connect surrogate nodes to linked entities if they are present in the KG, thus the new links have the form of $\langle e_m, e_l \rangle$, where e_l is the linked entity. Incorporating surrogate nodes into the KG changes the structure of the underlying KG. These textual entities introduce new semantics which we exploit in our similarity-based link prediction module.

Once we construct the augmented KG, we use a KG embedding algorithm to compute low dimensional embeddings for its entities. Given the vector representation of an entity, we propose a LP model which ranks entities based on their relevance to the surrogate nodes. We define LP formally as follows: given a head entity e_l, the goal is to infer a tail entity $\hat{t}$ that completes a link $\langle e_l, \hat{t} \rangle$.

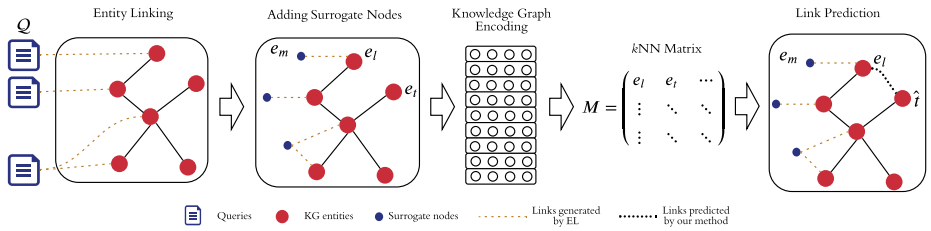


Figure 3.2: Using existing EL algorithms, we link the mentions (m) in the input queries to KG entities: linked entities (e_l). Then we create a node per mention: surrogate nodes (e_m). Next we obtain node embeddings and use the vector representations to calculate the kNN matrix (M). In this example e_t is the nearest neighbor to e_l so we call it the predicted tail ($\hat{t}$). $\hat{t}$ is used as a suggested query. The pair $\langle m, \hat{t} \rangle$ is then evaluated.

$$\hat{t} = \underset{e_i \in \mathcal{E}}{\operatorname{argmax}} f(\mathbf{e_l}, \mathbf{e_i}) \tag{3.1}$$

In this equation lower case notations in bold refer to embeddings and $f(.)$ is a score function that minimizes the distance between the two entities.

Given the embedding of a linked entity, $\mathbf{e_l} \in \mathbb{R}^d$ and the set of embeddings $\{\mathbf{e_i}\}$ where $0 < i < |\mathcal{E}|$ and $\mathbf{e_i} \in \mathbb{R}^d$, we search for top k most similar entities in the embedding space.

$$M = \underset{0<i<|\mathcal{E}|}{k\text{-argmin}} \|\mathbf{e_l} - \mathbf{e_i}\| \tag{3.2}$$

In other words, for each surrogate node, e_m, we find k nearest neighbors (kNN) of its linked entities and predict links in the form of $(e_m, \hat{t})$, where $\hat{t}$ is an entity belonging to kNN of e_l. This process is presented in Figure 5.1 and Algorithm 2.

3.4 Evaluation

One approach to assess the quality of the outcome of a search engine is to measure how satisfied the users are with the results, and the user satisfaction is quan-

Algorithm 2 Link Prediction Process

Input: KG $\mathcal{G}$ and the set of queries $\mathcal{Q}$
Output: Predicted links L

for each $q \in Q$ **do**
  $m \leftarrow$ mention in q
  $S[q] = \{e_m |\ q : m \mapsto e_l \in \mathcal{E}\}$
  for each $\langle e_m \rangle \in S[q]$ **do**
    Add (e_m, e_l) to $\mathcal{G}$
  end for
end for
Calculate embeddings for all the nodes in enhanced $\mathcal{G}$
Calculate M which is the kNN matrix for all of the embedding vectors.
$L = \{\}$ ▷ predicted links/tails per query
for each $q \in Q$ **do**
  for each $e_m \in S[q]$ **do**
    for each $\hat{t} \in M[e_m]$ **do**
      $L[q].append(\langle m, \hat{t} \rangle)$
    end for
  end for
end for

tified by several methods in the Information Retrieval community, such as relevance of the results to the query or quantifying the click information, etc.

Evaluating a system where ground truth information regarding the relevance of the returned documents to the target query is available is standard. However, in our problem setting we have a set of queries with unsatisfactory search results and the ground truth is not obtainable. Furthermore, the link prediction algorithm provides a set of entities as suggestions for reformulating the original query. Since our approach adds surrogate nodes that are not present in the ground truth, it is not possible to use the standard link prediction setting to evaluate our framework. Hence, we propose a rank-based evaluation technique that measures how well the distance metric ranks the entities.

3.4.1 Rank-based Evaluation

A surrogate node is a query in the KG which is connected to n entities obtained by one or multiple EL algorithms, and we predict k other entities per linked

entity. Thus we have at most $n \times k$ predicted entities. To sort these entities based on importance we consider their euclidean distance from their associated linked entity in the embedding space. To evaluate this sorted list of predicted links we count how many of them are present in the KG if we consider the top 10 and 50 linked entities. We normalize these numbers by the total number of predicted links to obtain Hits@10 and Hits@50.

$$Hits@k = \sum_{i=1}^{|L|} 1 \text{ if } rank_{(h,r,t)_i} \leq k$$

Additionally, we borrow two other metrics from information retrieval community: mean Average Precision (mAP) which measures the percentage of relevant suggested entities, and Normalized Discounted Cumulative Gain (NDCG) to give more weight to highly relevant suggested entities compared to moderately relevant ones.

Given a ranked list of predicted links per query we mark them as relevant if they exist in the KG and calculate AP:

$$\text{AP} = \frac{1}{n_r} \sum_{i=1}^{n} (P(i) \times \text{rel}(k))$$

where n_r is the number of relevant links, $\text{rel}(k) \in \{0, 1\}$ indicates if the link is relevant or not, and $P(i)$ is the precision at i in the ranked list. Once we obtain AP for each query we can average across all queries to find MAP:

$$\text{MAP} = \frac{1}{N} \sum_{q=1}^{N} \text{AP(q)}$$

where N is the number of all queries.

$$NDCG = \frac{DCG}{iDCG}$$

where *DCG* is Discounted Cumulative Gain which calculates the sum of all the relevance scores in a suggested query set and *iDCG* is the same measure for the sorted set of suggested queries:

$$DCG = \sum_{i=1}^{N} \frac{rel_i}{\log_2(i+1)}$$

where logarithm in the denominator takes the position of the suggested query into consideration.

3.4.2 Similarity-based Evaluation

In addition to rank-based metrics we can capture the relatedness of the suggested queries to the intended query by measuring the similarity between the two. The similarity can be defined in text space or in the embedding space. Hasibi et al., (Hasibi et al., 2017) proposed lexical similarity as a feature to measure relevance. For this metric we use Jaro edit distance to capture spelling mismatches.

$$\text{sim}_{lex} = \max_{\hat{t} \in L}(1 - dist(\hat{t}, q))$$

where $\hat{t}$ is the predicted tail entity and q is the target query.

To further measure the quality of suggested queries and following the idea proposed by Dehghani et al., 2017, we use a word embedding algorithm to obtain the vector representations of the target query and the suggested query and calculate the cosine similarity of the two vectors and report it as a performance measure. This is only possible if the click information of the users is available. To this end we choose a dataset which provides session based query log information from AOL (more details in the next section). Given two embedding vectors, $\mathbf{e}_1$ and $\mathbf{e}_2$, cosine similarity is defined as follows:

$$\text{sim}_{emb} = \frac{\mathbf{e}_1 \cdot \mathbf{e}_2}{\|\mathbf{e}_1\|\|\mathbf{e}_2\|}$$

3.4.3 Baseline

In this new problem setting, we define a strong baseline. We compare our model with the case where we predict links in the form of $\langle \text{EL}(m), \hat{t} \rangle$, where $\text{EL}(m)$ is the top linked entity for the mention m. Additionally, we establish a *gold*

standard upper bound in which we know the true entity for each mention and predict links in the form of $\langle s, \hat{t} \rangle$ where s is the true linked entity for mention m which means we have an error-free EL oracle. Please note that we cannot compare our model with conventional methods because our problem setting is different. In fact our model can be built on top of any query suggestion framework and improve their results.

3.5 Experiments

This section describes the datasets and evaluates our model based on the evaluation setting explained in previous section.

3.5.1 Datasets

We consider several benchmark datasets to evaluate our framework. These datasets are designed for tasks other than query suggestion, such as entity linking, named entity recognition, etc.

- **AIDA-YAGO2**: This dataset consists of hand annotated Reuters news articles and contains assignments of entities to the mentions of named entities (Hoffart et al., 2011). Mentions in this dataset can be mapped to YAGO, Wikidata, and Freebase entities.

- **ACE2004**: ACE is a subset of ACE co-reference dataset annotated by Amazon's MTurk (Ratinov et al., 2011), Originally, The Automatic Content Extraction (ACE) program presented this dataset for information extraction-related tasks such as entity recognition, relation recognition, and event extraction (Doddington et al., 2004).

- **AQUAINT**: This dataset is a news corpus consists of text data in English, drawn from three sources: the Xinhua News Service, the New York Times News Service, and the Associated Press (Milne & Witten, 2008). It has been used for evaluating entity disambiguation and entity linking tasks (De Cao et al., 2020; L. Wu et al., 2020a).

Table 3.1: Statistics of the benchmark datasets. We used BLINK L. Wu et al., 2020a to obtain accuracy measures in the last column which is a large scale entity linking algorithm.

Dataset	#documents	#mentions	EL accuracy
AIDA-YAGO2	946	18,448	80.27%
ACE2004	36	257	86.89%
AQUAINT	50	727	85.88%
MSNBC	20	656	85.09%
WNED-CWEB	320	11,154	68.25%
WNED-WIKI	320	6,821	80.67%
Yahoo!	980	2,114	60.07%

- **MSNBC**: This dataset contains top two stories in the ten MSNBC news categories (Cucerzan, 2007).
- **WNED-CWEB** and **WNED-WIKI**: These datasets were introduced in (Z. Guo & Barbosa, 2018) for the task of entity disambiguation and entity linking. WNED-CWEB and WNED-WIKI are obtained from large web corpora, namely Clueweb12 (Gabrilovich et al., 2013), and Wikipedia, respectively.

Moreover, we consider a dataset from information retrieval: Yahoo data search query log is part of the Yahoo Webscope program (Yahoo, 2013) that contains queries obtained from Yahoo web search. The basic statistics of these datasets are presented in Table 4.2.

For all the listed datasets in Table 4.2, true entity and the context around the mention are provided. True entity is the label that links mentions to Wikidata entities (Vrandečić & Krötzsch, 2014), and we use these labels to measure the performance of the entity linking and investigate how this performance affects the performance of our final model. State-of-the-art entity linking accuracy on these datasets is also provided in Table 4.2. For these datasets we report rank-based metrics, as we can establish baseline and the upper bound as explained in previous section.

To evaluate our work using similarity-based metrics, we use **AOL** dataset which is a collection of 20M web queries collected from 650k users from 01 March, 2006 to 31 May, 2006. Time stamp data and click through information are available and if the user clicked on a search result, the rank of the item on which they clicked is also listed (Pass et al., 2006). The presence of click information enables us to perform similarity-based evaluation. We split queries into sessions, every 30 minutes is considered a session (Jansen et al., 2007). The queries in each session form a context for the target query (the last query in a session is the target query if the user clicks on the search result). We perform basic preprocessing (e.g., removing punctuation and converting to lower case), and select 10,000 sessions at random for the experiment.

For our case study, we use a set of queries issued to Adobe Stock. This dataset only provides query information, and the ground truth and click information are not provided. Hence none of the evaluations discussed in previous section could be performed on this dataset. As a result, we picked 50 queries and asked 20 annotators to annotate the search result from two search engines (Google Image and Adobe Stock). Given the annotated pairs of queries and search results, we can study the quality of suggested queries.

As an underlying KG we use **FB15k-237** which contains 15K entities and 237 relations (Toutanova et al., 2015). It is based on Freebase and is a subset of FB15k (Bordes et al., 2013a) where redundant relations have been removed. This KG has been widely used in the literature (Bamler et al., 2020; Z. Sun et al., 2019b; R. Wang et al., 2019) for the task of graph completion. We use it as an external source of knowledge to suggest alternative queries. We employ the existing mapping between the entities of FB15k-237, denoted by *mid*s, and Wikidata entities. This mapping is required as the EL algorithm we use maps the mentions to Wikidata entities. Besides, we construct a reduced version of the FB15k-237 to only consider the intersection of the linked entities of the query log mentions and the knowledge graph entities.

Table 3.2: Rank based results for all datasets. The entity linking for Yahoo dataset is missing as the ground truth is not available for this dataset/task.

		Baseline				Our Approach				Gold Standard			
Dataset	EL Accuracy	Hits@10	Hits@50	mAP	NDCG	Hits@10	Hits@50	mAP	NDCG	Hits@10	Hits@50	mAP	NDCG
ACE2004	84.43	1.81	12.14	18.78	14.55	2.17	13.41	22.40	17.20	14.50	72.46	97.00	56.88
AIDA-YAGO2	79.51	1.65	7.16	33.72	44.93	4.21	10.60	61.78	54.34	11.25	38.25	73.36	63.79
AQUAINT	86.62	1.45	3.26	15.27	16.77	5.47	13.14	88.23	34.41	10.87	16.30	92.43	62.98
MSNBC	84.28	1.00	6.52	15.37	19.64	10.94	22.61	57.32	42.55	13.91	51.09	93.65	53.95
WNED-CWEB	67.47	3.43	11.38	43.84	43.20	6.47	16.29	68.10	59.01	10.40	34.58	77.48	61.12
WNED-WIKI	79.76	2.66	13.45	38.17	26.82	5.06	16.12	66.50	40.40	11.42	42.46	85.35	54.45
Yahoo!	-	2.28	22.57	32.14	16.71	14.49	25.88	92.22	47.90	23.81	55.69	97.66	57.80

3.5.2 Rank-based results

We compare the predicted links from our approach with links predicted by the baseline and gold standard. We report Hits@10, Hits@50, mAP and NDCG. Table 3.2 presents the performance measures for this analysis. Our approach lies between the performance of the baseline and gold standard, outperforming the baseline across all datasets.

Consider the case when the accuracy of EL is low (e.g., 67.47% for WNED-CWEB). In this case, we observe a considerable increase in performance over the baseline—a 100% increase in Hits@10. This is likely due the top linked entity being identical to the correct entity and is consistent with our underlying hypothesis. This suggests that our approach is most helpful when EL fails. We hypothesize that short queries are most likely to encounter low EL accuracy, and these are the focus of our case study (Section 3.6).

We further study the cumulative precision with regard to the distance used to rank the suggested queries. To do this, we first sort the predicted links based on distance, then calculate precision for each threshold considering all links with distance between 0 and that threshold. This is illustrated in Figure 3.3 for WNED-CWEB dataset. In Figure 3.3, precision at threshold 1 is the precision when considering all the predicted links, i.e., nothing is filtered out based on distance. Ideally, we should observe a sigmoid shape with the center at 0.5 (similar to the gold standard). Our model follows a similar trend. The precision of the baseline varies only up to about 0.4 and is not monotonic.

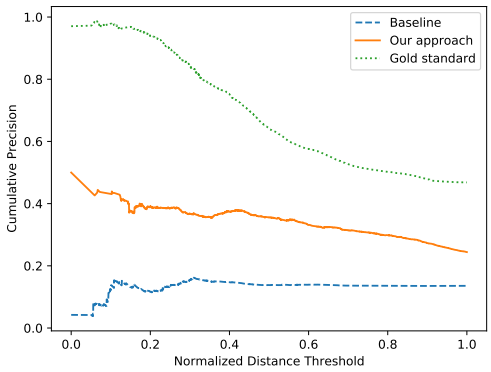


Figure 3.3: Cumulative precision at distance-based thresholds.

3.5.3 Similarity-based results

We report similarity-based results on AOL dataset. For this task we perform entity linking on the context of each session and predict tails using baseline and our approach. As described in previous section we use lexical and semantic similarity metrics to measure the relevance. For lexical similarity we obtain 59.3% similarity between the predicted tail and the target query (we retrieve this based on click information that is available in AOL dataset) for our approach, while achieve 48.2% similarity when using the baseline.

We then compute the embedding of the top predicted tail in each approach (using a pretrained BERT model (Wolf et al., 2020)), also compute the embedding of the target query and calculate the cosine similarity between the two embeddings per session. We obtain 90.6% for cosine similarity averaged across all sessions when using our approach compared to 86.3% for baseline.

3.6 Case Study

We investigate the quality of the suggested query candidates in a case study based on 210 queries and 50 MTurk participants. Participants judged the relevance of results obtained from Google Images and from Adobe Stock. We collected

10 ratings per query, search engine, and approach triple. We randomly sampled from real-world queries issued on Adobe Stock where the results were reported as unsatisfactory by users. We consider basic variants of using suggested entities as search queries that are produced by the baseline and our method: (*exclusive*) uses exclusively suggested entities as query, and, (*expanded*) uses the original query in conjunction with the suggested entities.

Table 3.3: (left) Results of case study broken down by configuration. Shown are percentage of participants rating result pages to be relevant and Krippendorff's α. (right) Results of filtered queries with lower rating variance in order to achieve a Krippendorff's $\alpha > 0.7$

	Baseline		**Our Approach**		
	(exclusive)	(expanded)	(exclusive)	(expanded)	
Google Image	52.5% (.408)	54.6% (.210)	60.0% (.492)	60.7% (.534)	**60.4%** (.414)
Adobe Stock	40.0% (.289)	53.8% (.629)	56.6% (.741)	62.5% (.561)	**51.6%** (.623)
	51.1% (.484)		**62.0%** (.619)		

	Baseline	**Our Approach**
Google Image	**74.2%** (.337)	**80.5%** (.370)
Adobe Stock	**40.0%** (.698)	**75.4%** (.756)

Overall—with a rather low overall Krippendorff's $\alpha = .55$—employing Knowledge-Derived Query Suggestion techniques such as the introduced baseline or our proposed approach lead on average to 57% relevant retrieval results. Our approach is outperforming the baseline in all experiment configurations. While Google's image search produces more relevant results compared to Adobe Stock due to its larger repository, our approach has the largest impact on Adobe Stock relative to the baseline. Table 3.3 (left) shows a breakdown of the experiment configurations including the corresponding Krippendorff's α scores. Our approach tends to produce larger inter-rater reliability.

A Krippendorff's α of .7 which allows tentative conclusions according to Krippendorff, 2018 can be achieved with the removal of 26% of queries that have the highest rating variance. Independent of the setup, in 64.8% participants found the results relevant. Table 3.3 (right) shows the corresponding results for several experiment configurations.

3.7 Summary

This chapter proposed a query suggestion framework that exploits an external source of knowledge. Using state-of-the-art entity linking, we added queries represented as surrogate nodes to an external KG and showed how the inclusion

of different senses of a query boosts the retrieval effectiveness. We devised a link prediction mechanism that returns a ranked list of queries similar to the linked entities and we proposed metrics to evaluate the list of suggested queries. We performed extensive experiments on seven benchmark datasets to show the superiority of our model over the baseline. We also carried out a case study to assess the qualitative effectiveness of our model.

As a future direction, we hope to focus on the problem of producing improved query suggestions. Currently our model suggests an alternative query but a hierarchical encoding scheme can enable users to have the ability to choose from generalization, i.e., integrating suggested entities into a higher-level entity, e.g., *student* and *faculty* into *university member*, or specialization, i.e., identifying sub-groups of the target query, e.g., *employee* to *developer* and *engineer*.

Chapter 4

Knowledge Graph as External Knowledge II

Pretraining domain-specific language models remains an important challenge which limits their applicability in various areas such as agriculture. This chapter investigates the effectiveness of leveraging food related text corpora (e.g., food and agricultural literature) in pretraining transformer-based language models. We evaluate our trained language model, called AgriBERT, on the task of semantic matching, i.e., establishing mapping between food descriptions and nutrition data, which is a long-standing challenge in the agricultural domain. In particular, we formulate the task as an answer selection problem, fine-tune the trained language model with the help of an external source of knowledge (e.g., FoodOn ontology), and establish a baseline for this task. The experimental results reveal that our language model substantially outperforms other language models and baselines in the task of matching food description and nutrition.

4.1 Introduction

United States Department of Agriculture (USDA) maintains a database called Food and Nutrient Database for Dietary Studies (FNDDS) which provides the nutrient values for foods and beverages reported in what is eaten in the US. Additionally, household and retail scanner data on grocery purchases, such as the

Nielsen data available through the Kilts Center for Marketing, have been extensively used in food policy research[4]. Mapping these two databases, i.e., food description found in retail scanner data, to nutritional information database is of utmost importance. This linkage can capture the relationship between the retail food purchase and community health and also the difference between poor and non-poor diets across the whole diet spectrum, and thus it can impact future funding policies that can provide healthy food for low-income households.

In this work, we aim to develop and employ Natural Language Processing (NLP) techniques to find the best linkage between the two databases. A common approach to tackle this kind of problems is semantic matching, which is the task of determining whether two or more elements have similar meaning. Bi-encoders are the most common techniques for semantic matching. A bi-encoder inputs two strings and encodes them in the embedding space and in the final layer calculates the similarity in a supervised fashion. That is why word embedding techniques are a good candidate for this purpose. Word embeddings have been utilized extensively for the task of semantic matching Kenter and De Rijke, 2015, but recent advancements in contextual word embeddings, in which each word is assigned a vector representation based on the context, have resulted in significant improvements in many NLP tasks, including semantic matching.

Transformer-based language models, e.g., BERT Devlin et al., 2019, have been widely used in research and practice to study computational linguistics and they have shown superior performance in variety of applications including text classification Jin et al., 2020, question answering W. Yang et al., 2019, and many more. However, these models are not generalizable to every domain when used with their default objectives, i.e., pretrained on generic corpora such as Wikipedia. To address this issue, previous work has attempted to incorporate domain-specific knowledge into the language model by different strategies. One of the prominent approaches is in biomedical domain where a BERT-based language model is pretrained on a large corpus of biomedical literature called BioBERT Lee et al., 2020. Motivated by the impressive performance of BioBERT, we use a large corpus of agricultural literature to train a language

[4] Researcher(s)' own analyses calculated (or derived) based in part on data from Nielsen Consumer LLC and marketing databases provided through the NielsenIQ Datasets at the Kilts Center for Marketing Data Center at The University of Chicago Booth School of Business. The conclusions drawn from the NielsenIQ data are those of the researcher(s) and do not reflect the views of NielsenIQ. NielsenIQ is not responsible for, had no role in, and was not involved in analyzing and preparing the results reported herein.

model for agricultural applications from scratch. The trained model will be further fine-tuned by the downstream tasks.

Another method to incorporate domain knowledge into the language model is to use an external source of knowledge such as a knowledge graph (KG). Knowledge graphs are rich sources of information that are carefully curated around objects called entities and their relations. A basic building block of a KG is called a triplet which consists of two entities and a relation between the two. Previous work has attempted to inject triples into the sentences Liu et al., 2020, however, injecting triples can introduce noise to the sentences which will mislead the underlying text encoder. To address this issue, we propose to add n entities from an external knowledge source (i.e., a knowledge graph) based on similarity that can be obtained by various methods such as entity linking. This augments the semantic space but keeps the vocabularies within the domain. We show how changing n can affect the performance of the downstream task, quantitatively and qualitatively.

Moreover, we propose to formulate the task of mapping retail scanner data (also known as Nielsen) to USDA description as an answer selection problem. Given a question and a set of candidate answers, answer selection is the task of identifying which of the candidates answers the question correctly. An answer selection model inputs a pair of question and answer and outputs a binary label (true or false), so it is a binary classification problem. Similarly, we can consider Nielsen product descriptions as the set of all questions and USDA descriptions as the set of all answers, and the goal is to find the best answer for each question. The difference is that in the original answer selection task, the number of answers is limited and usually unique to each question, however, in this setting there is a shared set of answers and its size is much larger. We use our pre-trained language model as the backbone for the answer selection component and we augment both questions and answers during fine-tuning using external knowledge to boost the performance. In summary, we make the following contributions in this chapter.

- We collect a large-scale corpus of agricultural literature with more than 300 million tokens. This domain corpus has been instrumental to fine-tune generic BERT into AgriBERT.
- We propose a knowledge graph guided approach to augment the dataset for the answer selection component. We inject related entities to the sentences before the fine-tuning step.
- AgriBERT substantially outperforms existing language models on USDA datasets in the task answer selection. We plan to release our datasets and language models to the community upon publication.

4.2 Related Work

4.2.1 Pre-trained Language Models

In NLP, Pre-trained language models learn from large text corpora and build representations beneficial for downstream tasks. In recent years, there are two successive generations of languages models. Earlier models, such as Skip-Gram Mikolov et al., 2013 and GloVe Pennington et al., 2014, primarily focus on learning word embeddings from statistical patterns, semantic similarities and syntactic relationships at the word level. With this first group of language embedding methods, polysemous words are mapped to the same representation, irregardless of word contexts. For example, the word "bear" in "I see a bear" and "Rising car sales bear witness to population increase in this area" will not be distinguishable in the vector space.

A later group of models, however, recognizes the importance of textual contexts and aims to learn context-dependent representations at the sentence level or higher. For example, CoVe McCann et al., 2017 utilizes a LSTM model trained for machine translation to encode contextualized word vectors. Another popular model, Bidirectional Encoder Representations from Transformers (BERT) Devlin et al., 2019 is based on bidirectional transformers and pre-trained with Masked Language Modeling (MLM) and Next Sentence Prediction (NSP) tasks, both ideal training tasks for learning effective contextual rep-

Table 4.1: An example of how we propose to extend the dataset.

Product Description	USDA Description	Label
domino white sugar granulated 1lb	salsa, red, commercially-prepared	False
domino white sugar granulated 1lb	cookie-crisp	False
domino white sugar granulated 1lb	sugar, white, granulated or lump	True

resentations from unlabelled data. It delivers exceptional performance and can be easily fine-tuned for downstream tasks.

BERT has enjoyed wide acceptance from the NLP community and practitioners from other domains. In particular, domain experts can build domain-specific BERT models that cater to specific environments and task scenarios.

4.2.2 Domain Specific Language Models

BERT has become a fundamental building block for training task specific models. It can be further extended with domain specific pre-training to achieve additional gains over general domain models.

Prior work has shown that language models perform better when the source and target domains are highly relevant Y. Gu et al., 2021; Lee et al., 2020. In other words, pre-training BERT models with in-domain corpora can significantly improve overall performance on a wide variety of downstream tasks Y. Gu et al., 2021.

There is also a correlation between a model's performance and the extent of domain specific training Y. Gu et al., 2021. In particular, Gu et al. Y. Gu et al., 2021 note that training models from scratch (i.e., not importing pre-trained weights from the original BERT model Devlin et al., 2019 or any other existing BERT-based models) is more effective than simply fine-tuning an existing BERT model with domain specific data.

In this chapter, agricultural text such as food-related research papers are considered in-domain while other sources such as Wikipedia and news corpus are regarded as out-domain or general domain. Our primary approach is in line with training-from-scratch with in-domain data.

4.2.3 Augmenting Pre-trained Language Models

Data augmentation refers to the practice of increasing training data size and diversity without collecting new data Feng et al., 2021. Data augmentation aims to address practical data challenges related to model training. It is applicable to scenarios such training with low-resource languages Xia et al., 2019, rectifying class imbalance Chawla et al., 2002, mitigating gender bias J. Zhao et al., 2018, and few-shot learning Wei et al., 2021.

Some data augmentation methods incorporate knowledge infusion. For example, Feng et al. Feng et al., 2020 used WordNet Miller, 1995 as the knowledge base to replace words with synonyms, hyponyms and hypernyms. Another study Grundkiewicz et al., 2019 extracts confusion sets from the Aspell spellchecker to perform synthetic data generation in an effort to enhance the training data, which consists of erroneous sentences used for training a neural grammar correction model.

However, there is limited research on the efficacy of applying data augmentation to large pre-trained language models Feng et al., 2021. In fact, some data augmentation methods have been found to have limited benefit for large language models Feng et al., 2021; Longpre et al., 2020. For example, EDA Wei and Zou, 2019, which consists of 4 operations (synonym replacement, random insertion, random swap and random deletion), provides minimal performance enhancement for BERT Devlin et al., 2019 and RoBERTa.

Nonetheless, researchers Feng et al., 2021 advocate for more work to explore scenarios in which data augmentation is effective for large pre-trained language models, because some studies H. Shi et al., 2021 demonstrate results contrary to the claims of Longpre et al., 2020.

In this study, we investigate the effectiveness of data augmentation with knowledge infusion and apply our method to the Answer Selection task scenario. We find that our method significantly improves semantic matching performance.

4.2.4 Answer Selection

Answer Selection refers to the task of finding the correct answer among a set of candidate answers for a specific question. For example, given the question "What is the capital of France?", a solution to this task is required to select the correct answer among the following choices:

- A) Paris is the capital of France.
- B) Paris is the most populous city in France.
- C) London and Paris are financial hubs in Europe.

In this case, the first answer should be selected. It is clear that matching words or phrases is not sufficient for this task.

A common approach is to formulate this problem as a ranking problem such that candidate answers are assigned ranking scores based on their relevance to the question. Earlier work primarily relies on feature engineering and linguistic information Yih et al., 2013. However, the advance of deep learning introduces powerful models Laskar et al., 2020; Rücklé et al., 2019 that outperform traditional methods without the need of manual efforts or feature engineering.

In this study, our goal is to establish valid mapping between food descriptions and nutrition data. We formulate this task as an Answer Selection problem and demonstrates the superiority of our method over baselines.

4.3 Methodology

4.3.1 Domain Specific Language Model

Training language models is a powerful tool for a variety of NLP applications, and when it comes to a particular task in a specific domain, it becomes more effective if the language model is trained on a corpora that contain large amount of text in that specific domain. Such practices exist in the literature in various domains, for instance BioBERT and FinBERT are successful examples of training a domain-specific language models in biomedical and financial domains,

respectively. Building upon previous research and motivated by the lack of existing corpora or a pre-trained model in agricultural domain and because we are interested in a model that produces vocabulary and word embeddings better suited for this domain than the original BERT, we collect 46,446 articles related to food and agricultural that contain more than 300 million tokens and use it to train a BERT model from scratch. This trained model can be used for various NLP applications in agricultural field. We adopt the standard procedure in training a language model which is masked language modeling. In Masked Language Modelling, a certain fraction of words in a given sentence are masked, and the model is expected to predict those masked words based on other words in that sentence. In this process it learns meaningful representation for each sentence.

4.3.2 Answer Selection Problem Definition

To evaluate the trained language model we require a downstream task in this specific domain. For instance most biomedical language models are evaluated on named entity recognition tasks on medical datasets. Due to the lack of benchmark NLP dataset in this domain, and since the semantic matching problem has practical values, we evaluate our model on this task. Semantic matching is a technique to determine whether two sentences have similar meaning. We express the task at hand as an answer selection problem, we could assign all USDA descriptions (answers) to each Nielsen product description (question) and select S random incorrect USDA description per product description as negative samples where $S << D$. In this case the size of extended dataset is $S \times D$. Table 4.1 provides an example of how we extend the dataset when $S = 2$.

4.3.3 Knowledge Infused Finetuning

There have been studies that successfully inject related information from an external source of knowledge to enhance the performance of the downstream task. For instance incorporating facts (i.e., a curated triple extracted from a knowledge graph in the form of `(entity,relation,entity)`) from knowledge

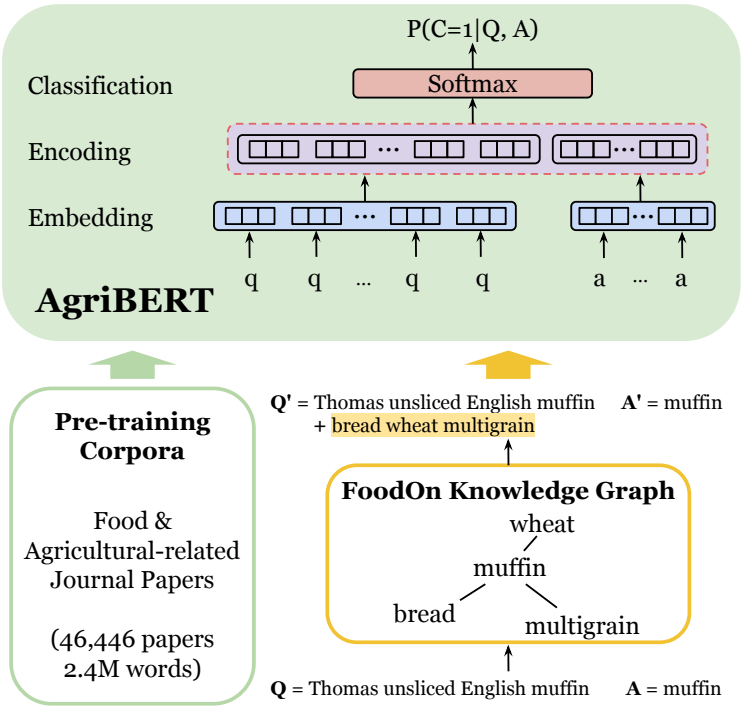


Figure 4.1: The overall framework of AgriBERT which is trained on agriculture literature from scratch. AgriBERT is evaluated on answer selection task. The answer selection component has two inputs: a question and an answer, and before we input them to the framework, we add new entities to them from an external source of knowledge such as Wikidata or FoodOn. The output of the framework is a score (a probability) which is used for ranking the answers.

graphs Liu et al., 2020, or injecting refined entities extracted from text to a knowledge graph Rezayi, Zhao, et al., 2021. In our setting, since we are dealing with answer selection and the size of training set is small, we propose to append external knowledge to both questions and answers to enhance the performance of the answer selection module.

Finding relevant external knowledge can be fulfilled via different mechanisms such as entity linking, querying, calculating similarity, etc. In this chapter we suggest to use entity linking and querying. In entity linking, all the named entities in a text are recognized and then linked to the entities of a knowledge graph which is an ideal solution for our case. However the downside of this approach

Table 4.2: Basic statistics of our dataset compared with two benchmark datasets in the standard language modeling field.

Dataset	Articles	Tokens	Words	Size
Penn Treebank	-	887,521	10,000	10MB
WikiText-103	28,475	103,227,021	267,735	0.5GB
Agriculture corpus	46,446	311,101,592	2,394,343	4.0GB

is that both entity recognition and entity linking algorithms are commonly trained on general text corpora such as Wikipedia and in our case it is not of practical value if we use these tools without reconfiguring them for our purposes. Hence, we consider a domain-specific knowledge graph, e.g., FoodOn Dooley and Griffiths, 2018 and we obtain new knowledge by querying it using the keywords in the text of question answer pairs. We simply append new entities to the end of questions or answers. Figure 4.1 illustrates our proposed framework.

4.4 Experiments

4.4.1 Datasets

We employ several datasets for training the language models, evaluating the trained language model, and augmenting the downstream dataset. In this section we briefly explain these datasets.

Language Training Datasets

Our main dataset is a collection of 46,446 food- and agricultural-related journal papers. We downloaded all published articles from 26 journals and converted the pdf files to text format for use in the masked language modeling task. We also cleaned the dataset by removing URLs, emails, references, and non-ASCII characters. In order to compare the contributions of different components of our model, we consider a secondary dataset for training (WikiText-103) that contains all articles extracted from Wikipedia. This datasets also retains numbers,

case, and punctuation, which is similar to our dataset described above. The statistics of the two datasets is provided in Table 4.2. We also include Penn Treebank dataset Marcus et al., 1994, which is another common dataset for the task of language modeling, as a reference.

Answer Selection Dataset

We use two different data sources for this part. First, we use the consumer panel product from the Nielsen Homescan data. Nielsen provides very granular data on the food purchases from the stores at the product barcode or Universal Product Code (UPC) with detailed attributes for each UPC, including UPC description. While scanner data come with some nutrition-related product attribute variables, this information is not sufficient to examine the nutritional quality. To address this issue, we link product level data from the Nielsen with the USDA Food Acquisition and Purchase Survey (FoodAPS) that supplements scanner data with the detailed nutritional information. The survey contains detailed information about the food purchased or otherwise acquired for consumption during a seven-day period by a nationally representative sample of 4826 US households. The FoodAPS matched 32000+ barcodes with the Food and Nutrient Database for Dietary Studies (FNDDS) food codes of high quality. The linked data set has UPC description for each product, and the corresponding FNDDS food code. In addition, the final data set has full information needed to construct diet quality indexes to evaluate the healthfulness of overall purchases.

External Source of Knowledge

We use FoodOn knowledge graph for the augmentation purposes. FoodOn is formatted in the form of OWL ontology. The OWL ontology provides a globally unique identifier (URI) for each term which is used for lookup services and facilitates the query processing system. Much of FoodOn's core vocabulary comes from transforming LanguaL, a mature and popular food indexing thesaurus Dooley and Griffiths, 2018. That is why FoodOn is a unique and valuable resource for enhancing our language model.

4.4.2 Metrics

Since the output of the evaluation task on the answer selection dataset is a ranked list of answers per question, we require metrics that take into account the order of results. That is why we propose to use precision at 1 (P@1) and Mean Average Precision (MAP).

Precision@1 or P@1. We sort the selected answers based on final similarity score and we count how many times the top answer is correctly selected.

$$P@1 = \sum_{i=1}^{|N|} 1 \text{ if } rank_{a_i} == 1$$

where N is the set of all questions.

Mean Average Precision (MAP). MAP measures the percentage of relevant selected answers. Given a ranked list of selected answers per question we mark them as relevant if they are correctly selected and calculate AP as follows:

$$\text{AP} = \frac{1}{n}\sum_{i=1}^{n}(P(i) \times \text{rel}(k)),$$

where n is the set of all selected answers, $\text{rel}(k) \in \{0, 1\}$ indicates if the answer is relevant or not, and $P(i)$ is the precision at i in the ranked list. Once we obtain AP for each question we can average across all questions to find MAP:

$$\text{MAP} = \frac{1}{|N|}\sum_{q=1}^{|N|}\text{AP(q)},$$

where N is the set of all questions.

4.4.3 Baselines

To perform ablation study and make sure that our dataset improves the performance of the downstream task and not using a pretrained model nor training from scratch, we consider following scenarios:

- *k*NN: we compute the embeddings[5] of the Nielsen product descriptions and USDA descriptions and for each vector belonging to the product description embedding space we find the most similar vector from the USDA description embedding space. This naive approach is effective if the number of unique USDA descriptions is small. However, this does not hold in our case.

- We use BERT without any modification as the underlying language model and use an existing answer selection technique to further fine-tune the model on the answer selection dataset.

- We consider BERT as the base language model and further train it with our own corpus. In this case the vocabularies of the final language model is the union of Wikipedia and our corpus. We employ the fine-tuned language model as the backbone of the answer selection tool and apply it on the unmodified answer selection dataset.

- We train a BERT model from scratch using masked language modeling technique on WikiText-103. We employ the fine-tuned language model as the backbone of the answer selection tool and apply it on the unmodified answer selection dataset.

- We train a BERT model from scratch to train a new language model using masked language modeling technique on our own dataset. We employ the fine-tuned language model as the backbone of the answer selection tool and apply it on the unmodified answer selection dataset.

[5] We use sentence-transformer library for this task: https://github.com/UKPLab/sentence-transformers

For the trained language model we also consider a scenario where we use entity linking algorithm to find related entities from Wikidata and append them to question and answers. As discussed this approach introduces noise to the text and may harm the performance but we include it as a baseline for the sake of comparison with the case where we query the FoodOn knowledge graph to augment the text of questions and answers.

Table 4.3: Examples to demonstrate the quality of added entities to the text of product descriptions. For Wikidata we use entity linking and we present here the top linked entity (highest confidence score). For FoodOn we use SPARQL to query the ontology and the first outcome is listed here.

Sentence	Wikidata entity	FoodOn entity
nestle nido powder infant formula	nestle	rice powder
aunt jemima frozen french toast breakfast entree	aunt jemima	frozen dairy dessert
woodys hickory barbecue cooking sauce	woody's chicago style	hickory nut
sour punch sour watermelon fruit chew straw	sour punch	sour milk beverage
philly steak frozen beef sandwich steak	philly steaks	wagyu steak
yoplait original rfg harvest peach yogurt low fat	yoplait	creamy salad dressing

4.4.4 Experimental Settings

Once the extended dataset is generated we can apply any answer selection method on the dataset. There are a number of studies in the literature on this topic, including COALA Rücklé et al., 2019, CETE Laskar et al., 2020, MTQA Deng et al., 2019, and many more, among which CETE is considered state-of-the-art in the answer selection task by the ACL community[6]. CETE implements a transformer-based encoder (e.g., BERT) to encode the question and answer pair into a single vector and calculates the probability that a pair of question/answer should match or not[7].

For the entity linking process we use the implementation proposed by L. Wu et al., 2020b, called BLINK. BLINK is an entity linking python library that uses Wikipedia as the target knowledge base. Moreover to send in efficient SPARQL queries to FoodOn knowledge graph which is in the format of OWL ontology we use ROBOT which is a tool for working with Open Biomedical Ontologies[8]. Additionally, we try to simulate a setting where the number of labeled training samples is small, thus we use 20% of the dataset for training and the remaining 80% for the test. We believe this is a more realistic scenarios in real world applications.

[6] Reported here: https://aclweb.org/aclwiki/Question_Answering_(State_of_the_art)

[7] The code for this study is open source and available for public use: https://github.com/tahmedge/CETE-LREC

[8] Find it here: https://github.com/ontodev/robot

Table 4.4: Test performances of all models trained on all datasets for the task of answer selection. for kNN model we use sentence-transformers to compute embeddings. EL stands for Entity Linking and bold numbers indicate the best performance. $BERT^p$ is a pre-trained BERT and $BERT^s$ means training a BERT model from scratch.

#	Training Dataset	Model	MAP	P@1
1	-	kNN	26.70	14.49
2	-	$BERT^p$	27,77	10.88
3	WikiText-103	$BERT^p$	28.03	11.12
4	WikiText-103	$BERT^s$+EL (Wikidata)	27.36	10.09
5	WikiText-103	$BERT^s$+FoodOn (n=1)	28.78	24.83
6	Agricultural Corpus	$BERT^p$	29.72	12.71
7	Agricultural Corpus	$BERT^s$	**44.21**	22.72
8	Agricultural Corpus	$BERT^s$+EL (Wikidata)	42.33	21.52
9	Agricultural Corpus	$BERT^s$+FoodOn (n=1)	31.54	47.89
10	Agricultural Corpus	$BERT^s$+FoodOn (n=3)	30.65	49.80
11	Agricultural Corpus	$BERT^s$+FoodOn (n=5)	29.91	**49.98**

4.4.5 Results

As Table 4.4 presents, not surprisingly, the best performance is obtained when the language model is trained on the agricultural corpus. We summarize the main observations as follows: First the *k*NN performs surprisingly well compared to other complicated methods, in fact in terms of P@1 it surpasses methods that are trained with BERT and the answers are selected using state-of-the-art answer selection framework. Next, the best MAP score (MAP= 44.21%) is obtained when the language model is trained with agricultural corpus from scratch, and any additional augmentation hurts the performance in terms of MAP. On the other hand, the model performance improves in terms of P@1 when incorporating external knowledge. More specifically, P@1= 49.98% when 5 new entities are added. This implies that by adding related external knowledge we can find the correct match roughly 50% of the times but if the correct match is not at the top position they ranked very low, and hence the low MAP score.

Moreover, by comparing lines 5 and 9 we can see that the inclusion of external knowledge alone cannot improve the quality of language model in a domain specific task. We also investigate the number of external entities that we include in the text of questions and answers and as lines 9-10 of Table 4.4 demonstrates, increasing n decreases the MAP score and increases the P@1. This suggests that incorporating related entities from a relevant knowledge source helps to find the correct match in 50% of the times but it mislead the answer selection module and rank the correct match lower that it drops the MAP score. Table 4.3 provides some examples of augmented sentences by Wikidata and FoodOn knowledge sources. As this table presents, linking the food description to Wikidata entities can easily go wrong, first three rows for instance, where the food descriptions are linked to brand names[9]. However this does not happens in querying method, as the entities are purely food related. Additionally, FoodOn entities contain food-related adjectives such as frozen, creamy, etc. that help in matching the food descriptions to nutrition data.

[9] https://en.wikipedia.org/wiki/Aunt_Jemima

4.5 Summary

In this chapter we trained a language model called AgriBERT that will facilitate the NLP tasks in the food and agricultural domain. AgriBERT is a BERT model trained from scratch with a large corpus of academic journal in the field. To evaluate our language model we propose to solve the problem of semantic matching which aims at matching two databases of food description (Nielsen database and USDA database). We reformulate the problem as an answer selection task and used our language model as a backbone of a generic answer selection module to find the best match. Before feeding the pairs of questions and answers to the model we augmented them with external entities obtained from FoodOn knowledge graph, a domain-specific ontology in the field of food. We showed that inclusion of external knowledge can help boost the performance in terms of the more strict P@1 measure but it lowers the performance in terms of mean average precision. As a future direction, we plan to investigate more sophis-

ticated approaches for incorporating external knowledge such as refining the knowledge before including it in the text.

Chapter 5

Similar Datasets as External Knowledge

Short text clustering is a challenging unsupervised learning task which requires a complex representation of each document to effectively model the semantics and syntactic structure of the text. Existing works have attempted to tackle this challenging task by incorporating additional information to the model, such as number of clusters, number of datapoints in each clusters, the distribution of the input data, and more. Unlike previous approaches, we propose to exploit an auxiliary dataset that is fully labeled to augment the quality of the learned representations. We also define the problem as cross domain clustering (XDC), which leverages adversarial learning to train an adaptive clustering model across text domains. Specifically, XDC jointly exploits a labeled source domain and an unlabeled target domain during model training. Owing to domain adversarial learning, the distribution shift across source and target domains could be mitigated. Moreover, XDC is implemented as a linkage-based clustering approach using graphs, which is agnostic of the number of clusters. We evaluate our XDC framework on three text datasets, and results show that it outperforms the state-of-the-art text clustering methods in most cases. Ablation studies and qualitative analysis also demonstrate the effectiveness of our framework.

5.1 Introduction

Clustering, in general, is challenging as it is an unsupervised tasks and labeling information is not available during training. To alleviate this challenge, existing clustering schemes make prior assumptions about type or shape of the data distribution Lloyd, 1982, number of clusters Van Gansbeke et al., 2020, or number of samples in each clusters J. Shi and Malik, 2000. More specifically, clustering short text has its own distinct challenges. On one hand, there is not enough semantic information in a short sequence of words, which makes it difficult to obtain useful representations for each document. On the other hand, due to the origin of the data (e.g., micro-blogs and news outlets), the topic space is large and evolving. The latter, particularly, is the reason traditional models fail to perform well in clustering short documents, as they require some prior knowledge about the data distribution.

Inspired by the remarkable advances in transfer learning, one promising solution to addressing the challenges in short text clustering is to exploit auxiliary data sources Rezayi, Zhao, et al., 2021, such as labeled text datasets. Our motivation is simple yet effective. We argue that the incorporation of auxiliary texts datasets (i.e., source domain) could help model the underlying distribution of target domain and therefore facilitate the discovery of short text clusters. A relevant problem setting in literature is unsupervised domain adaptation (UDA) R. Zhu and Li, 2022, although it is mainly applied to classification tasks Rezayi, Soleymani, et al., 2021; R. Zhu, Jiang, et al., 2021b. In UDA, we have a labeled source domain ($\mathcal{D}^s$) and an unlabeled target domain ($\mathcal{D}^t$). The goal is to train a model that can generalize to $\mathcal{D}^t$. Recent UDA methods adopt an adversarial learning pipeline, by adversarially training a data classifier and a domain classifier Ganin et al., 2016. Consequently, a domain-invariant feature space could be learned, and the classifier can generalize to the target domain $\mathcal{D}^t$.

In this chapter, we present a new problem setting, i.e., cross-domain short text clustering, and propose a adaptive cross-domain clustering (XDC) framework based on domain adversarial learning. First, a model is trained by jointly exploiting the labeled source domain $\mathcal{D}^s$ and unlabeled target domain $\mathcal{D}^t$ using

both a classification loss and an adversarial loss. Second, a separate process is adopted to find the clusters in $\mathcal{D}^t$ using a pseudo label propagation algorithm. The most similar work to our model is a face clustering approach Z. Wang et al., 2019 that trains a model on a large-scale labeled face dataset and then uses it to obtain cluster labels for the unlabeled dataset. A major shortcoming of this approach is that it only relies on source dataset to train a clustering model, and therefore it cannot mitigate the distribution shift between source and target datasets. Additionally, the feature extraction module in this work is fixed, and thus the model would not be well adapted to target domains.

Moreover, our XDC model does not make any prior assumption regarding the target dataset, hence it is useful for clustering short text corpora in which the label space is broad and short-lived (e.g., Twitter). We list our contributions as follows:

- We set up the problem setting, cross domain short text clustering, in which we use a labeled auxiliary dataset to train a classifier and then leverage it to find linkage scores between data points in an unlabeled short text dataset.
- We propose a cross-domain clustering (XDC) framework, which incorporates domain adversarial learning to reduce the domain shift between source dataset and target dataset. XDC is also agnostic of number of clusters and in general it makes no prior assumptions about the data.
- We conduct comprehensive experiments on benchmark datasets and show the effectiveness of XDC by examining it on three short text-based datasets. Experimental results show the superiority of our proposed model over the existing baselines.

5.2 Related Work

In this section we discuss related works in three categories. First, we introduce recent approaches in similarity-based clustering, Next we introduce graph-based methods for clustering, and finally we address how domain adaptation comes

into play. We also give explanation as to how our model is different from previous work.

5.2.1 Similarity-based Clustering

The notion of closeness has been employed widely in the literature of clustering in general, as in clustering it is desired to find the distance/similarity between different objects and decide how to assign them to different clusters Zhan et al., 2018. More specifically, C. Zhu et al., 2011 proposed Rank-Order distance which measures the dissimilarity between two samples using their neighboring information in the dataset and performs hierarchical clustering based on this metric, W.-A. Lin et al., 2018 considered the local structure of deep features to learn an improved similarity measure, and Z. Wang et al., 2019 found that using context of the graph associated to input samples is beneficial in learning meaningful clusters.

To address the scalability issue in the aforementioned methods, Otto et al., 2017 proposed to use an approximate version of Rank-Order distance. The computational complexity of the original Rank-Order approach was $\mathcal{O}(n^2)$, while using the approximation in calculating the k-NN graphs can reduce the run-time to $\mathcal{O}(kn)$ J. Chen et al., 2009; Silpa-Anan and Hartley, 2008. We use the same approximation in in calculating k-NN in our proposed method but our method is graph-based and it is different such that it also considers the context of the nodes in the graph that is representative of each input image.

5.2.2 GNN-based Clustering

Graph Neural Networks (GNN) X. Jiang, Zhu, Li, et al., 2020; R. Zhu, Tao, et al., 2021 are a branch of deep neural networks, which have demonstrated successful predictive performance in a wide range of applications involving graph-structured data M. Zhang and Chen, 2018, texts Yao et al., 2019 and images Mou et al., 2020. To benefit from GNN in clustering, we can represent the input samples as a graph, $G = (V, E)$ where V is the set of vertices and E is the set of edges which represents similarity/distance between pairs of samples.

L. Yang et al., 2019 was the first work that proposed to use Graph Convolutional Networks (GCN) Kipf and Welling, 2016a for clustering. They built an affinity graph based on latent representation of the input samples and used cosine similarity to find k nearest neighbors for each sample, and ran GCN on the resulting graph to refine the clusters. Following this idea, Z. Wang et al., 2019 proposed to construct a subgraph (called Instance Pivot Subgraph or IPS) per input sample and reformulated clustering as a link prediction problem where a link exists between two nodes when their identity labels are identical. They train the model on a large dataset and use the trained model on a smaller scale dataset to find the cluster assignments. This is similar to our work in a sense that we also use graph convolutional networks for classifying constructed graph, but our work is different as we expose the model to the target datasets during training, we update the features during training, and we perform domain adaptation to reduce the shift between the two datasets.

5.2.3 Domain Adaptive Clustering

Due to real world constraints, specially in short text dataset, domain shift is inevitable. To address the domain shift issue, numerous techniques have been proposed and successfully reduced the domain mismatch in different scenarios R. Cai et al., 2019; F. Yuan et al., 2019; R. Zhu, Jiang, et al., 2021a. The question here is that how an unlabeled test/target dataset can help train a better model. Based on the definitions and categorization of different transfer learning methods given in S. J. Pan and Yang, 2009, this setting is called transductive transfer learning where the source and target domains are different but a related and unlabeled target dataset is available during training.

Less studied is the problem of cross domain clustering. For instance Samanta et al., 2013 proposed to clusters data points in target domain using means in source domain and the assumption is that both domains have the same number of clusters which is not true in many real life applications. More recently Abavisani and Patel, 2018 proposed the use of domain adaptive clustering, but it is limited to subspace clustering and has no support for practical domain shift. We propose to use adversarial domain adaptation Ganin et al., 2016 that aims

at learning domain invariant features based on two deep classifiers on a labeled source dataset and an unlabeled target dataset. Additionally, there are some works in image domain that address this problem S. Li and Fu, 2016; Rezayi et al., 2022.

It is worth mentioning that different studies mentioned above consider various datasets to measure the quality of the clusters generated by their proposed model, and cannot be generalized to every dataset. This implies that there is an unknown parameter that lies in the data shape and distribution that has not been tackled in the previous work and was simply ignored.

5.3 Preliminaries

In this section we first introduce the notations used in the chapter, and set up the problem definition, and then discuss how our methodology is different from previous cross domain problems.

5.3.1 Notation and Problem Definition

We define cross domain clustering as follows: Let $\mathcal{D}^s$ be the source domain consisting of a feature space $\mathcal{X}^s$, a label space $\mathcal{Y}^s$, and a conditional distribution $P(Y^s|X^s)$. Every sample drawn from this distribution can be shown as $\{x_i^s, y_i^s\}$ where $x_i^s \in \mathcal{X}^s$ and $y_i^s \in \mathcal{Y}^s$. Similarly, we assume $\mathcal{D}^t$ is the target domain consisting of feature space $\mathcal{X}^t$ and the data distribution $P(X^t)$ which indicates label space is not available for target dataset. We also define domain label, d_i, as a binary variable for i-th sample, which indicates whether x_i is drawn from $\mathcal{X}^s$ or $\mathcal{X}^t$. Cross domain clustering, in general, aims to learn $P(Y|X)$ on labeled source dataset and assign a pseudo label $\hat{y}_i^t$ to each unlabeled target data point, x_i^t.

5.3.2 Discussion

The most similar problem setting to ours in terms of methodology is (adversarial) unsupervised domain adaptation (UDA) in which labeled source data, $\mathcal{D}^s$,

is used to train a classifier in an adversarial fashion and the model learns features that are discriminative for the main learning task on $\mathcal{D}^s$ and indiscriminate with respect to the shift between $\mathcal{D}^s$ and $\mathcal{D}^t$. Note that in UDA the label space is shared between the two domains. This is a huge advantage in producing the shared label space. UDA has been widely used and implemented for the task of classification considering different modalities, e.g., image, text, and many more X. Chen and Cardie, 2018; Ganin et al., 2016; Tzeng et al., 2017. However, XDC aims at a different problem where there is no information regarding the target space, and there is no assumption on the type and the shape of the target data distribution, $\mathcal{D}^t$. This makes the problem more challenging as establishing alignment between $\mathcal{D}^s$ and $\mathcal{D}^t$ in the feature space is not as straightforward as it is in the case of UDA.

The most similar work to our model in terms of application is the work of Z. Wang et al., 2019, in which they use a general and large scale dataset to train a classifier and employ that classifier to cluster a small-scale target dataset using linkage clustering approach. Our work is similar to Z. Wang et al., 2019 as we also use linkage clustering approach to perform our final clustering, however, we propose to expose the model to the target dataset to learn more effective features, and as we show in the ablation study, this is very helpful in creating a more representative feature space, and improving the performance of downstream task, i.e., linkage-based clustering.

5.4 Methodology

5.4.1 Overview

We approach the cross-domain clustering problem by domain adversarial learning and linkage-based clustering. Our idea is to expose the target dataset to the model during training and use the adversarial loss introduced in Ganin et al., 2016 to update the features. This helps to learn features that are more discriminative between the source and target datasets. We also propose to use a Graph Convolutional Networks (GCN) Kipf and Welling, 2016a to train a classifier on

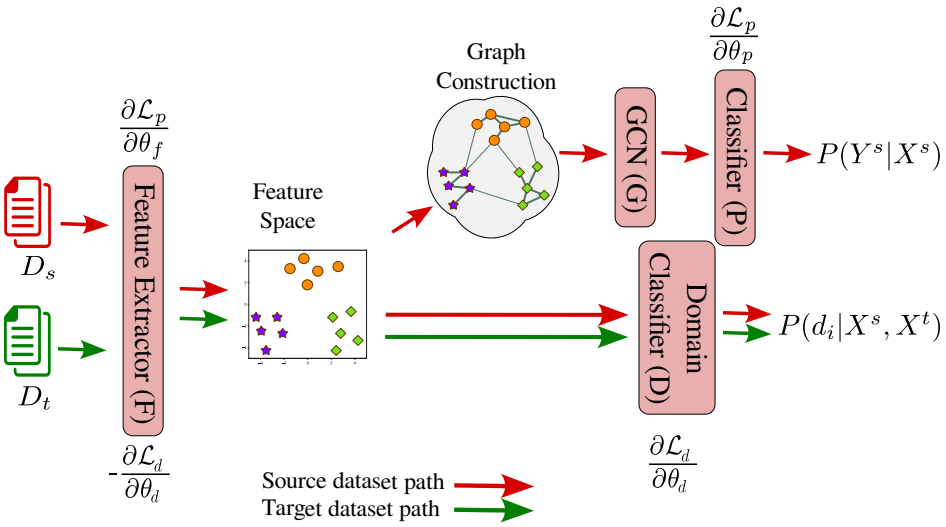


Figure 5.1: The training module of our proposed model.

labeled source dataset. The graph generation process is discussed in Section 5.4.3. Our proposed model is different from Z. Wang et al., 2019 as we incorporate the feature extraction module in the training to learn more effective features, and additionally we expose the model to the target dataset in an adversarial fashion to learn more discriminative features. The proposed method is illustrated in Figure 5.1. In what follows, we first discuss how we employ GCN classifier for text, then describe the learning and inference processes.

5.4.2 GCN Classifier

The idea of using GCN for text classification is not new and previous studies has demonstrated the effectiveness of this approach Yao et al., 2019. In this work, we practice this idea and apply GCN on multiple graphs that are constructed using the vector representations that are obtained from the feature space. Converting text to graphs is a common practice and various methodologies exist for this task, for instance, dependency tree parsers Do and Rehbein, 2020; C. Zhang et al., 2019, constituency tree parsers Marcheggiani and Titov, 2020; K. Yang and Deng, 2020, information extraction L. Huang et al., 2020; Vashishth et al., 2018, and similarity in the feature space Jia et al., 2020; Zhou et al., 2020 have

Algorithm 3 Training process of XDC

Input: $\mathcal{X}^s$, $\mathcal{Y}^s$, and $\mathcal{X}^t$

```
for each epoch do
    for each mini-batch do
        Obtain features from source using Eq. (5.2).
        Update the node embeddings in this batch.
        Generate IPS graph as presented in Sec. 5.4.3.
        Train a GCN classifier using Eq. (5.1).
        Concatenate source and target inputs.
        Obtain features from combined inputs, Eq. (5.2).
        Train an MLP on the combined features with L_d.
        Run Adam optimizer to minimize L in Eq. (5.5).
        Update model parameters θ_f, θ_p, and θ_d.
    end for
end for
```

Output: trained G_p and G_f

been exploited to construct graphs from text for a variety of applications such as question answering, topic models, etc. In this work we employ the similarity based approach to generate graphs per each short document. Similarity-based graph construction methods have been proven to be effective for text clustering and topic modeling Z. Wang et al., 2019; Zhou et al., 2020.

As Figure 5.1 shows, both source and target datasets go through feature extractor module $\mathbf{F}$ and the feature vectors of source dataset are used to construct graphs. The graph construction module is explained comprehensively in Z. Wang et al., 2019, but we summarize its main points here. For each feature vector associated to each data point (called pivot node), we construct a graph based on similarity in the embedding space. First, the top u most similar vectors are selected to constitute the neighborhood of the pivot node, and then edges are added in a 2-hop neighborship: k_1 is the number of edges in the first hop and k_2 is the number of edges in the second hop. This graph is called Instance Pivot Subgraph (IPS).

The constructed graphs are then fed into a GCN module $\mathbf{G}$ and parameters θ_p in classifier $\mathbf{P}$ and θ_f in feature extractor $\mathbf{F}$ are updated using the supervised

loss function $\mathcal{L}_p$. The bottom path is the domain classifier $\mathbf{D}$ where feature vectors of source and target datasets are inputted into a fully connected neural network and parameters θ_d and θ_f are updated using the supervised loss function $\mathcal{L}_d$. In our proposed model, the GCN model $\mathbf{G}$ is formulated as:

$$\mathbf{G} = \text{GCN}(A, F) = \sigma([F||\text{agg}(A, F)]), \tag{5.1}$$

where σ is the sigmoid function, $||$ is the concatenation operation, and agg is an aggregation function such as mean aggregation or attention aggregation Veličković et al., 2018b. Additionally, A is the adjacency matrix of the constructed graph generated by feature vectors f after each batch and F is the feature matrix obtained from the following equation:

$$F = \mathbf{F}(x_i^s). \tag{5.2}$$

5.4.3 Cross Domain Adversarial Learning

More formally, the adversarial domain adaptation problem is optimized by playing a minimax game with two competitive optimizations: (a) minimizing the error in $\mathcal{E}(\mathbf{F}, \mathbf{G}, \mathbf{P})$ on the feature extractor $\mathbf{F}$, GCN model $\mathbf{G}$, and classifier $\mathbf{P}$ to guarantee source lower risk; (b) minimizing the error over domain classifier $\mathbf{D}$ but maximizing the error over feature extractor $\mathbf{F}$ in $\mathcal{E}(\mathbf{F}, \mathbf{D})$:

$$\mathcal{E}(\mathbf{F}, \mathbf{G}, \mathbf{P}) = \mathbb{E}_{(x_i^s, y_i^s) \sim \mathcal{D}^s} L_y(\mathbf{P}(\mathbf{G}(\mathbf{F}(x_i^s))), y_i^s), \tag{5.3}$$

$$\begin{aligned} \mathcal{E}(\mathbf{F}, \mathbf{D}) = -\mathbb{E}_{x_i^s \sim \mathcal{D}^s} \log[\mathbf{D}(\mathbf{F}(x_i^s))] - \\ \mathbb{E}_{x_i^t \sim \mathcal{D}^t} \log[1 - \mathbf{D}(\mathbf{F}(x_i^t))], \end{aligned} \tag{5.4}$$

where $L_y(\cdot, \cdot)$ is the cross-entropy loss function. The minimax game in adversarial domain adaptation is defined as:

$$\begin{aligned} &\min_{\mathbf{F}, \mathbf{G}, \mathbf{P}} \mathcal{E}(\mathbf{F}, \mathbf{G}, \mathbf{P}) - \eta \mathcal{E}(\mathbf{F}, \mathbf{D}) \\ &\max_{\mathbf{F}, \mathbf{D}} \mathcal{E}(\mathbf{F}, \mathbf{D}), \end{aligned} \tag{5.5}$$

Algorithm 4 Inference in XDC

Input: G_p, G_f, and $\mathcal{X}^t$
1: **for** each epoch **do**
2: **for** each mini-batch **do**
3: Obtain features from target inputs using G_f.
4: Update the node embeddings in this batch.
5: Generate IPS graph as presented in Section 5.4.3.
6: Use the saved G_p classifier on the IPS graphs.
7: Perform cluster assignment based on link prediction results.
8: **end for**
9: **end for**
Output: Cluster Assignments

where η is a hyper-parameter to balance source risk and domain adversary.

The detailed training process is presented in Algorithm 3.

5.4.4 Inference

Once the GCN model is trained we can save its parameters and use them in the test phase. Since we require to obtain features for the target dataset as well, we fix the weights of the encoder and use them in the test phase. In fact, we use the trained **F** to obtain new features from target dataset and repeat the graph generation process per data-point and perform the linkage-based cluster assignment as follows. Employing **G** we find scores between the pivot node and other nodes present in its related graph. These scores and the smoothness assumption (i.e., if two data-points are close to each other, their labels are likely to be the same), enable us to use pseudo-label propagation algorithm to find a label for unlabeled data-points Zhan et al., 2018. Inference procedure is presented in Algorithm 4 and Figure 5.2.

Labels are propagated to all nodes in a connected components and connected components are formed based on the scores produced by the GCN classifier, if the score between two nodes is larger than a threshold then an edge is added between the two nodes until a component is formed. If the size of the

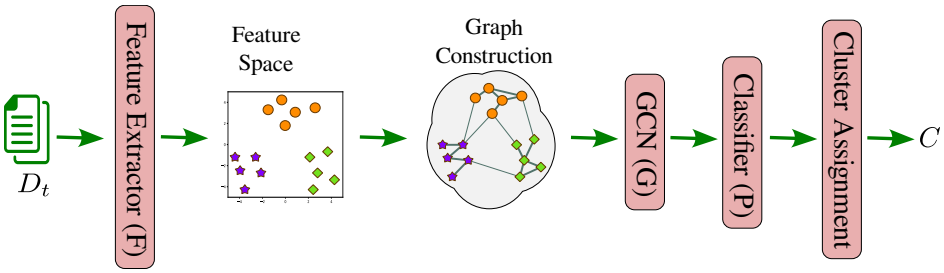


Figure 5.2: The inference module of our proposed model.

component is greater than a predefined value it will be discarded and its associated nodes will be considered for the next iteration, otherwise it will be assigned a new pseudo label.

5.5 Experiments

In this section we introduce the datasets and compare the performance of our model against the existing baselines. Furthermore, we conduct an ablation study in which we inspect our model when the domain classifier component is not present. Next, we discuss sparse clusters and qualitatively analyze some of so called singleton clusters. Finally, we explain the implementation details of different components of our model.

5.5.1 Datasets

We consider following three benchmark datasets:

Tweets: Contains 30,322 tweets belong to 269 topics in the TREC microblog aiming at real-time filtering task for monitoring a stream of social media posts[10]. This is particularly a useful dataset as the documents are short and data is large scale enough to be used as the source dataset for other target datasets.

[10] https://trec.nist.gov/data/microblog.html

GoogleNews: This dataset was collected by Yin and Wang, 2014 from Google News on November 27, 2013, and the authors crawled the titles and snippets of 11,109 news articles belonging to 152 clusters[11]. This is the second largest dataset and we consider it as the source dataset for Tweets dataset.

[11] https://news.google.com/

Reuters: This dataset was originally constructed for multi-class classification and datapoints may belong to multiple classes, thus similar to Yin and Wang, 2016 documents with more than one class are removed and the new dataset consists of 9,447 documents from 66 topics.

Since the task is cross-domain clustering we need to consider a source dataset for each target dataset. To this end, we examine different datasets as the source datasets and report the one that yields the best performance. Based on our experiment, larger datasets are more useful to be the source dataset. Similar observation was reported in Z. Wang et al., 2019 where CASIA dataset with 200K samples was used as source while the target datasets have 18K, 36K, and 68K samples.

5.5.2 Baselines

We compare our model with state-of-the-art methods in short text clustering including OSDM J. Kumar et al., 2020, MStreamF Yin et al., 2018, DMM Yin and Wang, 2014, Sumblr Shou et al., 2013, and DTM Blei and Lafferty, 2006. The choice of baselines is based on the fact that these methods are also ignorant of number of clusters in their model and their main focus is short text clustering. DTM and DMM are statistical topic models that discover the abstract "topics" or hidden semantic structures that occur in a collection of documents. The rest of the baselines are specifically designed for short text clustering. Other text clustering methods in the literature such as D. Zhang et al., 2021 that make prior assumption regarding the target dataset, or Guan et al., 2020 that focus on clustering based on feature extraction are excluded from the list of baselines in our experiments.

5.5.3 Metrics

We adopt four mainstream metrics to evaluate the performance of our model: Accuracy, Normalized Mutual Info (NMI), Homogeneity score (h), and V-Measure (VM).

Table 5.1: Performance result for the three datasets and six baselines in terms of accuracy, normalized mutual info (NMI) score, homogeneity, and V-Measure score. Second to last row is the results of ablated XDC where target dataset is not exposed during the training. Note that MStreamF has two variants: MF-G and MF-O. For all the baselines we directly report the performance measures from J. Kumar et al., 2020.

Source Dataset	Tweets				GoogleNews				Tweets			
Target Dataset	GoogleNews				Tweets				Reuters			
Model	ACC	NMI	h	VM	ACC	NMI	h	VM	ACC	NMI	h	VM
DTM Blei and Lafferty, 2006	0.65	0.81	0.83	0.81	0.24	0.80	0.82	0.80	0.67	0.54	0.66	0.53
DMM Yin and Wang, 2014	0.33	0.59	0.59	0.59	0.15	0.64	0.62	0.64	0.65	0.45	0.47	0.45
Sumblr Shou et al., 2013	0.61	0.57	0.55	0.57	0.54	0.70	0.76	0.70	0.65	0.46	0.40	0.46
MF-G Yin et al., 2018	0.51	0.78	0.75	0.78	0.70	0.79	0.74	0.79	0.45	0.36	0.32	0.36
MF-O Yin et al., 2018	0.42	0.68	0.65	0.68	0.24	0.75	0.69	0.74	0.58	0.36	0.37	0.36
OSDM J. Kumar et al., 2020	**0.88**	0.80	**0.95**	0.80	0.66	0.83	**0.93**	0.83	**0.93**	0.55	**0.95**	0.48
XDC (w/o $\mathcal{L}_d$)	0.66	0.83	0.84	0.83	0.68	0.82	0.86	0.81	0.47	0.54	0.84	0.53
XDC	0.72	**0.92**	0.87	**0.92**	**0.72**	**0.89**	0.91	**0.90**	0.50	**0.61**	**0.95**	**0.56**

ACC: In a classification problem accuracy is simply computed by comparing ground truth with the predictions. However, in clustering there is no association between the ground truth and the predicted cluster labels. That is why to calculate clustering accuracy the best match between the ground truth and the cluster labels should be found:

$$acc(y, c) = \max_{perm \in P} \frac{1}{n} \sum_{i=0}^{n-1} 1(perm(c_i) = y_i),$$

where P is the set of all permutations (there are $k!$ permutations where k is the number of obtained clusters).

NMI: Given two splits, one based on ground truth, Y, and another obtained from the predicted clustering assignment, C, NMI calculates how these two splittings agree with each other using the following formula:

$$NMI(Y, C) = \frac{I(Y, C)}{\sqrt{H(Y)H(C)}},$$

where $I(Y, C)$ is the mutual information between I and C and $H(\cdot)$ is the entropy function.

Homogeneity and V-Measure: Homogeneity measures how much a cluster contains samples belonging to a single class and is defined as:

$$h = 1 - \frac{H(Y|C)}{H(C)},$$

where $H(.)$ is the entropy function. Another similar measure to Homogeneity is called *Completeness* that measures how much similar samples are put together by the clustering algorithm and is given by:

$$c = 1 - \frac{H(Y|C)}{H(Y)},$$

and V-Measure is defined as the weighted mean of Homogeneity and completeness:

$$\text{VM} = \frac{(1+\beta)hc}{\beta h + c}.$$

5.5.4 Results

Table 5.1 presents the results for the six baselines and our model (last row), for three different settings. In this table the best result in each column is in boldface and the second best result is underlined. In the first setting we use Tweets dataset as the source dataset and GoogleNews as the target dataset. In this case we outperform state-of-the-art in two out of four reported metrics, namely NMI and V-Measure and we obtain the second place in the other two. In the second scenario, we switch places and use GoogleNews as the source dataset and Tweets as the target dataset, In this scenario, we outperform all the baselines with noticeable margins in all metrics except homogeneity where we obtain comparable results with OSDM. Finally, we use Tweets dataset as the source for Reuters dataset and again outperform all four baselines in terms of NMI, *h*, and V-measure. The accuracy for the final case is much lower compared with OSDM. While we could not find any reasonable explanation for this, we manually investigate it in section 5.5.6. Overall, our model stands in the first

or second place in all scenarios and produces better or comparable results with state-of-the-art baselines.

5.5.5 Ablation Study

In this section we examine the effect of inclusion of domain classifier in the model. We claim the domain classifier helps the model learn better features which ultimately yields to better cluster assignments. To this end we remove the domain classifier from the model and repeat the experiments. Last row in Table 5.1 demonstrates the performance results for the ablated version of XDC (i.e., XDC without $\mathcal{L}_{\lceil}$). As it is evident, in all cases XDC outperforms its ablated version. Additionally, we can observe that XDC without domain classifier cannot beat OSDM either, which shows the significance of the discriminator module.

5.5.6 Qualitative Analysis

In this section, we investigate the issue of singleton clusters. As we make no prior assumptions on the target dataset, and since singleton clusters are pure, we may end up with larger number of clusters than expected, many of which are sparse or singletons. We manually inspect some of the singleton clusters and find out that most of them are very vague in terms of topic and could belong to any class. Table 5.2 lists some of these examples from Reuters dataset. The last column in this table indicates the topics that might be mixed up with the actual topic. For instance, "british petroleum raises north sea butane prices by 1550 dlrs a tonne today" can be clustered as *gas*, *fuel*, or *dlr*, while the true label for this data-point is *nat-gas*. Further inspections of this dataset also reveals that topics are very similar to each other which explains the overall low performance for this dataset. Moreover, 13.6% of the classes contain only one data-point in Reuters dataset which also clarifies the large number of singleton clusters in our analysis.

Table 5.2: Examples of singleton clusters from Reuters dataset and possible topic mix-ups. Note that the text is preprocessed and all stop-words and punctuations are removed.

Sentence (preprocessed)	True Label	Possible Topic Mix-ups
british petroleum raises north sea butane prices by 1550 dlrs a tonne today	nat-gas	gas, fuel, dlr
silver state mining corp said it expects gold production this year to be more than double	gold	silver, gold, trade
us senate panel votes to limit county loan rate changes starting with 1988	grain	instal-debt, rice
shv said it making tender offer for up to 33 mln shares in imperial continental gas	acq	gas, trade

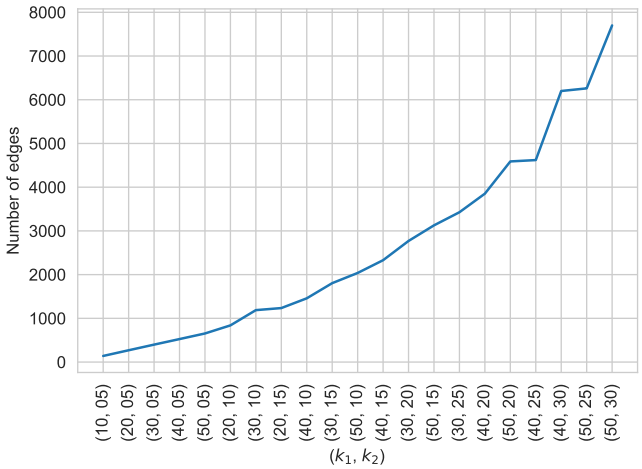


Figure 5.3: Number of edges in the resulting graphs per (k_1, k_2) pair. A step in this figure is an indication of forming a dense component in the graph. The intuition behind this is that if there are dense clusters in the data, the graph will suddenly get a lot of edges as k_1 and k_2 approach the optimal values.

5.5.7 Hyperparameter Tuning

Our model contains several hyperparameters that affect the final performance. Here we inspect how to select the set of optimal hyperparameters. The first hyperparameter is the maximum size (max_size) of the connected components in the label propagation algorithm described in 5.4.4. We tweaked this parameter in the range of $[100, 3000]$ with the step of 100 and realized that the performance is not very sensitive to this parameter, meaning that the performance metrics changed notably only 3 times during the tuning, and we managed to quickly find a reasonable value for max_size which is 2000.

Furthermore, we fine-tune the parameters that control the size and structure of the IPS graph, namely k_1 and k_2.[12] To this end, we plot the number of edges per (k_1, k_2) pair and find where there is a jump in the plot. Because if k_1 and k_2 are small then we would end up with small and sparse IPS graph and if they are large then we would have large and dense IPS graphs, To find the optimal graph size/density we can do a grid search and plot the number of edges per (k_1, k_2) pair. We consider the range $[10, 50]$ with the step of 10 for k_1 and $[5, 30]$ and the step of 5 for k_2. We do a grid search and make sure that $k_1 > k_2$. Figure 5.3 illustrates the average number of edges in IPS graphs generated during the inference for different combinations of k_1 and k_2. This graph is also sorted based on the number of edges to find the shift in the trend. As this figure shows, there are two jumps in the graph, one from (40, 25) to (40,30) with a smooth transition to (50,25) and then another jump to (50,30). As the choice of (k_1, k_2) also affects the time complexity we select $k_1 = 40$ and $k_2 = 30$.

[12] Reminder: To construct the IPS graph edges are added in a 2-hop neighborship per document in the embedding space: k_1 is the number of edges in the first hop and k_2 is the number of edges in the second hop.

5.5.8 Implementation Details

Preprocessing: We employ a pretrained BERT Devlin et al., 2019 model to obtain initial representation for all the documents. Then we use an efficient implementation of kNN[13] to produce k nearest neighbors of each input sample among all samples. The kNN matrix is used to compute the Instant Pivot Subgraphs (IPS). The feature matrix and kNN matrix for both source and target datasets, and label information for source dataset are the inputs of our model.

[13] https://github.com/facebookresearch/faiss

Encoder: To learn low dimensional representations during training from both source and target datasets we use a pretrained BERT Devlin et al., 2019 model. For this pretrained model we unfreeze the last two layers and update the weights during training. Doing so, allows the features to be updated by both label and domain classifiers in a more efficient manner.

Label Classifier: We use a Graph Convolutional Network Kipf and Welling, 2016a for the label classifier. We implement four convolutional layers with 512 neurons in each layer following by three fully connected layers to obtain a two

dimensional array. Note that this is a binary classifier that works on edges and decides whether a predicted edge exist in the graph or not (link prediction).

Domain Classifier: Following Ganin et al., 2016 we implement a two layer fully connected network followed by a softmax classifier to perform domain classification. This is also a binary classification task which aims at distinguishing source samples from target samples.

5.6 Summary

In this work we proposed to use domain adversarial training of neural networks to learn clusters of a source dataset and used learned parameters of the trained model to obtain cluster assignments for a target dataset. We showed that the domain adaptation component of the proposed model helps to improve the performance of the clustering task on short text datasets, significantly. Our model is not sensitive to the number of clusters and makes no prior assumption about the target dataset. As the future direction of this work, we plan to use self-supervised techniques to extract features from input data. For instance, (D. Zhang et al., 2021) used augmentation technique to obtain even more useful features. Another area of improvement is to incorporate the number of clusters into the model and compare it against related baselines such as D. Zhang et al., 2021.

Chapter 6

Explainable Knowledge Graph Embedding

Knowledge graph embedding is a state-of-the-art method for learning low-dimensional representations of entities and relationships in a knowledge graph. However, its interpretability, explainability, and fairness are still major challenges to be addressed. Interpretability refers to the ability to understand how the embeddings are derived and how they reflect the underlying data. Explainability refers to the ability to provide justifications or reasons for predictions or decisions made based on the embeddings. Fairness refers to the requirement that the embeddings do not perpetuate or introduce biases based on sensitive attributes. In this chapter, we systematically review the existing literature on interpretability, explainability, and fairness in knowledge graph embedding. By examining these three aspects, the chapter provides an insight into the current limitations and opportunities in knowledge graph embedding. This survey concludes that, while progress has been made in the field of knowledge graph embedding, there is still much work to be done in the field of trustworthy knowledge graph embedding. In particular, we believe the focus of future research should be on developing methods that are more transparent, robust, and accountable, so that the results of knowledge graph embeddings can be trusted and used effectively in a wide range of applications.

6.1 Introduction

Knowledge Graphs (KG) are object models to Knowledge Base (KB) realization, and they allow to express and manipulate data in the most intuitive manner. Exploiting the rich semantic and structure of KGs has led to considerable advances in different areas of AI technologies such as content recommendation X. Wang et al., 2019, semantic search Xiong et al., 2017, word sense disambiguation Simov et al., 2016, scene graph generation J. Gu et al., 2019, and many more. A KG comprises a set of entities, $\mathcal{E}$, and a set of relations, $\mathcal{R}$. Every triplet in the form of $\langle h, r, t \rangle$, where $h, t \in \mathcal{E}$ and $r \in \mathcal{R}$, is called a *fact*, and such structures are often called multi-relational data, in contrast to graphs with only one relation type among nodes, i.e., edges. Hence, $\langle h, r, t \rangle$ indicates that there is a relationship, r, between the head entity, h, and the tail entity, t.

Knowledge Graph Embedding (KGE) is a technique used to represent entities and relationships in a knowledge graph in a low-dimensional vector space. KGE aims to map entities and relationships in a knowledge graph to a vector space, where similar entities and relationships are located close to each other. KGE is widely used in natural language processing and information retrieval to make predictions and recommendations, perform information extraction and question answering, and improve search engines performance Rezayi, Lipka, et al., 2021. KGE models like TransE Bordes et al., 2013a, ComplEx Trouillon et al., 2016, DistMult B. Yang et al., 2015, etc are some of the most popular models that have been proposed to learn the embeddings of entities and relationship in a knowledge graph. There are several survey papers that provide a comprehensive overview of KGE. For example, Q. Wang et al., 2017 and Dai et al., 2020 are two such survey papers that cover the latest developments in KGE.

While existing surveys on KGs mainly focus on KG completion Z. Chen et al., 2020, KG applications Ji et al., 2021, and KG Embedding Dai et al., 2020; Q. Wang et al., 2017, this chapter instead focuses on some important issues related to KGs that have been largely ignored in existing survey papers, such as interpretability, explainability, and fairness. Interpretability and explainability are closely related concepts and are sometimes used interchangeably, but in

this chapter, we distinguish between them in the field of KGE. Interpretability involves providing justifications for predictions or decisions based on the embeddings. The aim of interpretability in KGE is to make the embeddings easy to understand for human users. Techniques like probing and embedding visualization are used to enhance interpretability. On the other hand, explainability refers to understanding the mechanisms behind the generation of the embeddings and how they reflect the underlying data. For example, determining which triples are the most influential in the performance of the downstream task is a part of explainability in KGE. Finally, fairness in KGE refers to ensuring that the embeddings are free from biases related to sensitive attributes. such as race, gender, religion, etc.

Interpretability is a crucial aspect in the development of KGE models as it allows human users to comprehend and trust the results produced by the models. In applications such as recommendation systems, link prediction, and question answering, the results of the KGE models must be transparent and understandable. For example, in a recommendation system, interpretability can help users understand why a certain product is recommended to them based on their previous purchases or search history. Similarly, in link prediction, interpretability can help users understand how two entities are related based on the relationships in the knowledge graph. In this way, interpretability enhances the trust and confidence in the results produced by KGE models and plays a critical role in their widespread adoption.

Explainability in KGE aims to shed light on the mechanisms behind the creation of embeddings and how they represent relationships between entities in a knowledge graph. The importance of explainability in KGE lies in the fact that it helps ensure the reliability and robustness of these models. For example, consider a KGE model that is used to predict the likelihood of a drug-disease interaction. The explainability of this model is essential in determining the factors that influence the prediction, such as the type of drug and the underlying medical condition. Without explainability, it would be difficult to validate the accuracy and reliability of the model's predictions. Additionally, measuring explainability in knowledge graph embeddings is also a challenge, with most

methods focusing on finding influential facts and using the performance of link prediction as a measure.

With the increasing use of KGE models in decision-making systems, it is crucial to ensure that the models are fair and do not discriminate against certain groups of entities or relationships. Fairness in KGE is particularly important in applications where the models are used to make decisions that impact people's lives, such as in hiring, lending, or criminal justice. The goal of fairness in KGE is to ensure that the models are unbiased and that the results produced are not influenced by sensitive attributes such as gender, race, religion, etc. It has been discovered that bias exists in knowledge graph embeddings. For instance, (Fisher et al., 2019) showed that *journalism* is more likely associated with people in Jewish community while African Americans are more likely to be American football player.

6.2 Interpretable Knowledge Graph Embedding

Interpretability in AI refers to the extent to which the results of a model or its decision-making process can be understood and explained by a human. Interpretability is crucial for ensuring trust in AI systems and improving their accountability. Interpretability is also important for making the AI models more user-friendly, which can increase their adoption and usage Ferreira et al., 2020. For example, a model that predicts medical diagnoses must be interpretable to ensure that patients and doctors understand the reasoning behind a diagnosis, and can make informed decisions Bai et al., 2018. In all cases, interpretable AI models can help to increase trust, improve accountability, and increase the overall effectiveness of the technology.

KGE models learn a mapping from data space to an embedding space, where each entity and relationship is represented as a vector. This mapping is trained with the aim of preserving the relational information in the knowledge graph and capturing its structure. The goal of interpretability in KGE, in this chapter, is to understand the learned mapping from the data space to the embedding space and to better comprehend the properties of the learned embeddings. This

includes what the learned embeddings represent, and how the embeddings are related to the properties of the entities in the knowledge graph.

By understanding the learned embeddings, one can gain insight into the workings of the model and make informed decisions about how to use the model or how to further improve it. Additionally, interpretability can help with debugging the model, as well as identifying potential limitations in the model's behavior. For instance, a book recommendation system that is based on a Knowledge Graph that encapsulates books, authors, and the relationships between them, as well as user preferences and reading behavior, may not be interpretable. In such a scenario, it would be challenging to comprehend the underlying mechanisms by which the model generates its recommendations. For example, it would be unclear whether the recommendations are generated based on the content of the books or the popularity of their authors, or if the model is biased towards specific genres or topics Yu et al., 2019.

One of the well-established techniques for interpreting the low-dimensional vectors generated by AI models is referred to as "probing". This method aims to understand which information is encoded in the vectors and how it can be used to make predictions. By training a separate auxiliary classification task, known as a "probing task", on the vectors, it is possible to uncover which properties of the elements in the model are captured in the embedding. The assumption behind this approach is that if certain information about a node is encoded in its vector representation through the embedding process, it should be possible to recover that information from the embedding alone. Adi et al., 2016 is one of the earliest works in the area of NLP.

Inspired by this work, (Ettorre et al., 2021) proposed to predict structural, semantic, and contextual features. For instance, *neighborhood size*, *entity type*, and in the case of entities of type student, the *student's university* are among the properties they tried to predict given the embeddings as input to a logistic regression classifier. The embeddings are generated by well-known KGE approaches such as DistMult, ComplEx and TransE and the takeaway of this work is that KGE techniques are capable of capturing information about the characteristics of elements within a knowledge graph, including structural, semantic, and

context-specific information. Conversely, authors in Jain et al., 2021 conducted a classification and clustering task to predict entity types and they found that KGEs cannot effectively capture the semantic information in KGs. As an example, the classifier's performance worsened as the entity types became more fine-grained. For instance, it is easy to distinguish between *people* and *organizations* but difficult to differentiate between *scientists* and *writers*. Other similar works in this area include Gad-Elrab, Stepanova, et al., 2020; Gad-Elrab, Ho, et al., 2020; Moon et al., 2017; Y. Zhao et al., 2020 but they are limited to a specific task, e.g., link prediction, or specific feature such as *entity type*.

In another line of work, the model interpretation is measured by a specific metric. For instance authors in Sengupta et al. aim to improve the interpretability of the embeddings by incorporating a coherence measure as a regularization term in the overall loss function, utilizing entity co-occurrence statistics from text sources. More formally, they defined the following regularization term:

$$coherence@k = \sum_{i=2}^{k} \sum_{j=1}^{i-1} p_{ij}$$

where p_{ij} is the PMI[14] score between entities e_i and e_j extracted from text data. It has been demonstrated that $coherence@k$ has a strong correlation with human interpretability of topics learned through topic modeling methods Lau et al., 2014, hence the authors anticipate obtaining interpretable embeddings by maximizing coherence@k. The final metric is calculated as follows as a measure of interpretability:

[14] Point-wise Mutual Information

$$\mathbf{WI}(e_i) = \sum_{j=1, j\neq i}^{k+1} p_{ij}$$

The experimental results shows that the proposed method demonstrates improved interpretability while retaining comparable performance levels.

Following the idea of exploiting word embedding in interpreting KGE, (Allen et al., 2021) aims to understand the geometric properties of relation representations in knowledge graphs. The authors establish a set of conditions (partitioning relations into three types) that are required to map subject enti-

ties to related object entities. These conditions provide a basis for considering the loss functions of existing knowledge graph models. This study shows that the better a model's architecture satisfies the established conditions, the better its performance in link prediction. In general, they determine explicit requirements for representation of each relation type.

The final category in interpretable KGE is visualization. The authors in Ettorre et al., 2022 investigate the relationship between the embeddings and the structure and semantics of the knowledge graph through visualization. They developed a tool called *Stunning Doodle* that can visualize the KG and provide additional statistics, such as the distance in the data space as well as in the embedding space, but this tool is limited in terms of functionalities.

The existing work in the area of interpretable KGE has several limitations that hinder its practical application and effectiveness. One limitation is the difficulty in directly relating the embeddings to the structure and semantics of the knowledge graph. Although various methods have been proposed to induce interpretability in the embeddings, including the use of auxiliary tasks and regularization terms, the results are still far from perfect. Another limitation is the lack of tools and methods that can effectively visualize the embeddings and their relationship with the knowledge graph. Visualization tools such as Stunning Doodle provide some statistics and visualization, but they are limited in terms of functionalities and do not provide a complete understanding of the embeddings. Furthermore, existing methods struggle with capturing fine-grained properties of elements in the knowledge graph, particularly when it comes to semantic information.

One of the potential future directions in the area of interpretable KGE is the integration of domain knowledge into the embedding process. As (Sengupta et al., 2020) shows, by incorporating constraints and prior information, the embeddings can better reflect the specific characteristics of the domain and thus improve their interpretability. Another direction could be to develop more standard methods for evaluating and measuring interpretability. Additionally, incorporating machine learning techniques, such as reinforcement learning,

to optimize interpretability with respect to user preferences could also be a promising avenue for future research.

6.3 Explainable Knowledge Graph Embedding

Explainable Artificial Intelligence (XAI) is a subfield of Artificial Intelligence (AI) that seeks to develop AI systems that can effectively communicate their decision-making processes and reasoning to humans in a transparent manner. The goal of XAI is to enhance the interpretability and accountability of AI systems by providing human-understandable explanations for the decisions, actions, and inferences made by these systems. This is particularly important in domains such as healthcare Ghassemi et al., 2021, finance Bussmann et al., 2020, and legal Deeks, 2019, where the interpretability and transparency of the decision-making process is crucial for building trust among stakeholders.

In the context of KGE, and in this chapter, explainability is defined as the ability to understand and interpret the embeddings generated by KGE models. KGE models map entities and relationships in a knowledge graph into a low-dimensional vector space, which is used for downstream tasks such as link prediction, entity classification, and many more. However, the embeddings generated by these models can be considered a "black box" as it is difficult to understand how they are generated and how they contribute in the performance of the model. There are a limited number of studies in the literature when it comes to explainability in KGE. The major approach is to analyze the vulnerability of knowledge graph embeddings through designing efficient adversarial attack strategies H. Zhang et al., 2019. In other words, existing methods try to identify which training facts have been most influential to the prediction A. Rossi et al., 2022.

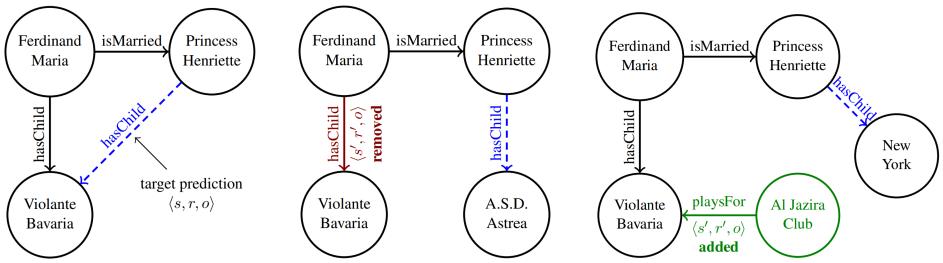
Explainability in KGE is important because it provides a clear understanding of the underlying data and how the embeddings are generated. This helps to assess the reliability and accuracy of the embeddings, and to identify any potential errors or weaknesses in the system. Furthermore, having a clear understanding of the mechanisms behind the generation of the embeddings enables

the development of better algorithms, and helps to optimize the performance of the KGE system. For example, for predicting the efficacy of potential treatments for specific diseases, consider the following triple ⟨Drug A, treats, Disease X⟩, in which a KGE model may predict that Drug A is effective in treating Disease X. However, it is crucial for the user of the model, such as a doctor, to understand the reasoning behind the model's prediction in order to have *trust* in its accuracy. Explainability in KGE allows for the revelation of the most influential training facts that lead to the prediction, such as ⟨Drug A, structuralySimilarTo, Drug B⟨, which is known to be effective in treating Disease X. This information can aid the doctor in making a more informed decision about prescribing the drug to their patient.

Prior to the seminal work on explainability in KGE H. Zhang et al., 2019, various methods for attacking GNN models are proposed, including J. Chen et al., 2018; Zügner et al., 2018. However, it should be noted that these methods cannot be generalized to KGE methods, as the underlying data structures and representation learning processes of graphs and knowledge graphs are inherently distinct. For adversarial attacks on KGE, (H. Zhang et al., 2019) studies the vulnerability of KGE by designing an adversarial attack strategy and showed how KGE can be attacked by manipulating the training set by adding and/or deleting certain facts. These types of attacks are referred to as data poisoning in the literature. In order to determine the necessary additions or deletions, the method aims to minimize the *plausibility* of a targeted fact by making changes (i.e. adding or deleting facts) to the training set. To do so, under the attack scheme, they perturb the facts that involve the K-hop neighbors of head or tail entity in the target fact. To address efficiency concerns, this method randomly selects entities from the K-hop neighborhood, which leads to a performance reduction that is not substantial.

(Bhardwaj et al., 2021a) proposed two adversarial attacks including Bhardwaj et al., 2021a and Bhardwaj et al., 2021b. The former addresses the sampling issue by constructing a set of decoy facts, and reducing the prediction confidence of the target fact by improving the prediction confidence of decoy facts, and the latter uses instance attribution methods[15] to identify influential triples

[15] Instance attribution methods are techniques used in interpretable machine learning to identify the training samples that are most influential to the model's predictions on test instances.



(a) Original KG (b) Remove (c) Add

Figure 6.1: Illustration of adversarial deletion (b) and addition (c) in CRIAGE. the original KG (a) has two facts: ⟨*Ferdinand Maria, isMarried, Princess Henriette*⟩ and ⟨*Ferdinand Maria, hasChild, Violante Bavaria*⟩. The link prediction algorithm correctly predicts the new fact ⟨*Princess Henriette, hasChild, Vilante Bavaria*⟩ (a). However, when a new triple ⟨*Violante Bavaria, playsFor, Al Jazira Club*⟩ is added, the new predicted link becomes ⟨*Princess Henriette, hasChild, New York*⟩, which is incorrect (c). This example highlights the potential for adversarial changes in the KG to lead to incorrect predictions by the model (image from Pezeshkpour et al., 2019.)

as adversarial deletions. Both these methods perform better than random sampling at the cost of efficiency, as they require larger search through training instances to find the most influential triples per each target fact. Similar to these methods is CRIAGE Pezeshkpour et al., 2019 which examines the impact of adding and deleting facts on the performance of the link prediction. For a target triple $\langle h, r, t\rangle$, CRIAGE finds the triples of the form $\langle h', r', t\rangle$ that maximizes the difference between score functions when trained on the original knowledge graph, G and its modified version, e.g., $G - \langle h', r', t\rangle$ in case of deletion. They also approximate the change in the embeddings using Taylor expansion which improves the efficiency, dramatically. 6.1 illustrates the effect of adversarial deletion and addition in the prediction of the KGE model.

Two recent studies Lawrence et al., 2021 and A. Rossi et al., 2022 have taken a similar approach to evaluate the robustness of KGE models against adversarial modifications to the training data. In Lawrence et al., 2021, the authors aim to improve efficiency by storing the model parameter updates for all training samples, but this leads to memory overhead for the model. On the other hand,

(A. Rossi et al., 2022) introduces Kelpie, a framework for explaining link predictions in embedding-based models. Kelpie is applicable to any embedding-based link prediction model, regardless of its architecture, and determines the training data subset responsible for a specific prediction. Kelpie considers both necessary (absence of facts) and sufficient (presence of facts) scenarios to explain the prediction.

There are several future directions for explainable KGE. One direction is adversarial robustness. KGE models are vulnerable to adversarial attacks, where attackers can modify the training data to manipulate the model's predictions. Future research should focus on developing methods that can detect and defend against such attacks, while still maintaining high accuracy. Another challenge in explainable KGE is scalability. Current explainability methods for KGE models require a search through the training instances to find the most influential triples that lead to a prediction for each target fact. As KGs become larger, this search can become computationally expensive. To address this challenge, future research should focus on developing explainability methods that can handle large-scale KGs efficiently. This could involve designing more scalable algorithms that can search through the training instances more quickly or utilizing techniques such as sampling or approximation to reduce the computational overhead of the search process.

Moreover, considering the extensive body of research on explainable Graph Neural Networks (GNNs) Cucala et al., 2022; H. Yuan et al., 2022, there is potential for integration with explainable KGE methods to address the scalability challenge. Heterogeneous GNN methods, in particular, have the ability to handle relational data which is similar to knowledge graph data. This makes them a promising approach for improving the scalability of explainable KGE methods. By leveraging the strengths of GNNs to handle the relational structure of KG data, future work could investigate how explainable GNN techniques could be applied to explainable KGE, providing interpretable explanations for link predictions while efficiently handling large-scale KGs.

6.4 Fair Knowledge Graph Embedding

Fairness in AI refers to the property of a model or system where its outcomes are impartial, unbiased and not influenced by irrelevant characteristics of the input data or individuals. The goal is to ensure that the model does not discriminate based on protected attributes such as race, gender, religion, etc. A biased AI systems can lead to unfair and harmful consequences for individuals and communities, and negatively impact decision-making in areas such as hiring, lending, and criminal justice. Mehrabi et al., 2021 contains a comprehensive study about fairness and bias in machine learning techniques.

Similarly, in KGE, fairness is the property of a KGE model where it does not produce biased representations of entities and relations in the knowledge graph based on irrelevant characteristics of the entities, such as gender, race, religion, etc. The goal is to ensure that the KGE model does not propagate or amplify existing biases in the knowledge graph, and that it generates fair and impartial embeddings for all entities. This can be evaluated in several ways, such as ensuring that the embeddings are not correlated with protected attributes, or that the representations are diverse and not dominated by a single group. Achieving fairness in KGE is important because knowledge graphs play a crucial role in a wide range of applications, including information retrieval Reinanda et al., 2020, recommendation systems Q. Guo et al., 2020, and question answering Yani and Krisnadhi, 2021, and it is important to ensure that these systems do not perpetuate biases or discrimination.

Fairness is important in KGE as the embeddings generated by KGE models are used in a variety of applications that can have significant impacts on people's lives. For example, KGE models may be used in various applications of natural language processing, or decision-making algorithms, where biases in the embeddings could result in unfair or discriminatory outcomes. If a KGE model is biased, it can reinforce and augment the existing bias in society, and results in harm to marginalized groups. Ensuring fairness in KGE is crucial for building responsible and trustworthy AI systems that can make positive contributions to society, while avoiding negative consequences. For instance, a KGE model

trained on a knowledge graph of job applicants may generate biased embeddings if it unfairly favors certain races or genders which could result in discriminatory hiring outcomes.

Bias in Knowledge Graph Embedding (KGE) models can arise from either the data used to train the model or from the model itself. In this survey, we focus specifically on the sources of bias that come from the model, as opposed to the data Radstok et al., 2021. However, it is important to note that both the data and the model can contribute to biases in KGE, and a comprehensive approach to fairness should take both sources into account. For a more in-depth survey on bias in knowledge graphs, please refer to Kraft and Usbeck, 2022.

The "Word Embedding Association Test" Caliskan et al., 2017 is a widely used method for identifying bias in word embeddings. It calculates the cosine distance between an entity embedding and the average embedding of two sets of attribute words, such as male and female. For example, in a test on gender bias, the target words could be "nurse" and "doctor", while the attribute words could be "caring" and "intelligent". The test would measure the similarity between the target words and each set of attribute words and compare the mean similarity between the two target words. If the mean similarity between "nurse" and the attribute words "caring" is significantly higher than the mean similarity between "doctor" and "caring", then this may indicate a gender bias in the embeddings towards associating the word "nurse" with "caring".

Following this idea, Bourli and Pitoura, 2020 proposes to measure bias with a similar formulation as follows:

$$S_{(a,b)}(x, y) = \cos(\mathbf{a} + \mathbf{r} - \mathbf{b}, \mathbf{x} + \mathbf{r} - \mathbf{y})$$

where r is the relation ("has occupation" in our case) and (a, b) are the seed pair of words ((a,b) = (female, male)). Experimenting on WikiData dataset, they find that "gender bias in occupations exists and is amplified by the embeddings".

However, Fisher, Palfrey, et al., 2020 suggests that "distance-based metrics are not suitable for measuring bias in knowledge graph embeddings" and instead define a more complex metric to measure the bias in KGE. They use the score function of the KGE method, e.g., $s(h, r, t) = ||\mathbf{h} + \mathbf{r} - \mathbf{t}||$ for TransE, the

objective is to evaluate the effect of the sensitive attribute (e.g., gender) on the model's predictions of a person's profession (p) through the score function:

$$s(e_{person}, r_p, e_{profession})$$

They find that their score is relevant such that there is a strong correlation between the proposed bias score and the ratio of male to female in most professions in WikiData. For instance, not surprisingly, the bias measure for the profession "military commander" is much higher for men and it is consistent with the number of males to females, 1,077 to 0. This is also the case for the profession "banker" with the ratio of 6,664 to 288 which is naturally expected to be neutral.

More recently, on measuring bias in KGE, Du et al., 2022 proposes a group level metric as follows:

$$B = \frac{1}{|F|} \sum_{h \in F} s(h, r, t) - \frac{1}{|M|} \sum_{h \in M} s(h, r, t)$$

where M and F are the sets of all male and female person entities with t respectively, and r is the relation $hasProfession$. Higher values of B is an indication of bias towards male. Similar to previous work they find evidence from data (e.g., male to female ratio) that their metric in fact measures bias in KGE. They also claim that their proposed metric is more comprehensive than previous metrics as it considers additional structural information in its evaluation.

Another line of work in fair KGE is debiasing. The aim of debiasing techniques is to modify the learning algorithm or the training data to eliminate or mitigate the presence of bias. Bose and Hamilton, 2019 is among the first studies that discussed debiasing from KGE. For each sensitive attribute, they design a filter in the form of an adversarial regularizer to predict the presence of the sensitive attribute based on the node embeddings:

$$\sum_{k=1}^{K} \sum_{a^k \in A_k} \log(D_k(\mathbf{h}, a^k))$$

where A_k is a set of sensitive attributes and D_k is the binary classifier (discriminator). The proposed method demonstrates its efficacy on the MovieLens1M dataset, however, its performance decreases significantly when evaluated on the triple prediction task of the FB15K dataset. Additionally, it has been shown to be unable to remove more than one source of bias simultaneously. To address this issue, Fisher, Mittal, et al. propose an adversarial loss function to balance the prediction with respect to each of the protected groups:

$$L = \sum_{p} \sum_{j} \text{KL}(G_j, \text{softmax}(s(e_p, r_j, t)))$$

where e_p is the head entity of type person, and r_j is a sensitive relationship, such as hasReligion, KL is the Kullback-Leibler Divergence, and G_j is the balance distribution. For instance, if there are 3 tail entities then $G_j = [0.33, 0.33, 0.33]$. Their experiments shows that their model outperforms the adversarial regularizer proposed by Bose and Hamilton, 2019 while remains as efficient. Finally, researchers in Arduini et al., 2020 propose another adversarial scheme in which the discriminator is a binary classifier that attempts to predict the sensitive attribute from a filtered embedding. Experimental results demonstrates that the proposed method is able to remove gender bias in KGs by obtaining close to 50% accuracy for the discriminator classifier which is expected.

One potential future direction for fair KGE could be studying the trade-off between accuracy and fairness in KGE models. Existing works struggle to find this balance and this issue has not been properly explored in the literature.

Interpretability, explainability, and fairness are important concerns in Knowledge Graph Embedding (KGE). The ability to understand how the embeddings are generated and how they can be utilized is crucial for their application in various domains. In this survey, we reviewed several existing methods for improving interpretability and explainability in KGE and methods for mitigating bias. We also discussed challenges and limitations in this field, including the trade-off between interpretability and accuracy and the need for robust and scalable eval-

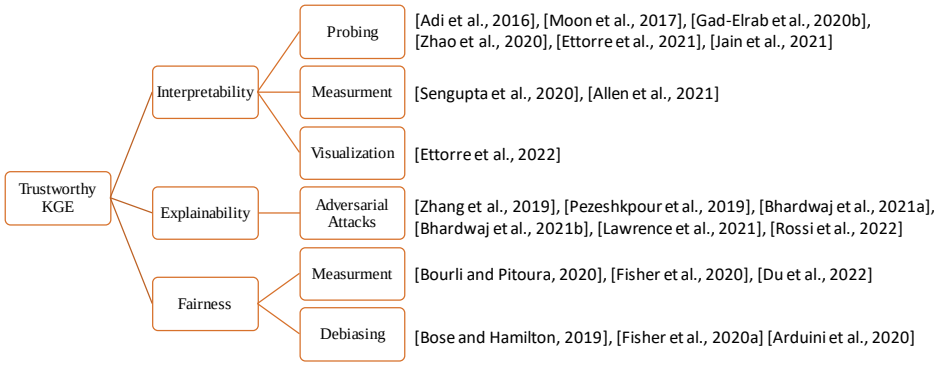


Figure 6.2: Related works on trustworthy knowledge graph embedding methods

uation metrics. Figure 6.2 illustrates the most notable studies in these areas in three categories.

Given the challenges and limitations in the field of interpretability, explainability, and fairness in KGE, there are several avenues for future research. One promising direction is to borrow methods from Graph Neural Networks (GNNs) for improving interpretability and fairness in KGE. Another direction is to use word embeddings as a guide in interpreting KGE and to leverage the learned structure from word embeddings to better understand KGE. Additionally, improving interpretability in KGE could also help in mitigating bias as it would provide insight into the sources of bias and allow for better-informed debiasing strategies.

Chapter 7

Conclusion

Representation learning has been a topic of interest for machine learning researchers in recent years. It involves learning a meaningful representation of complex data, which can be used for various tasks such as classification, clustering, and prediction. Despite the significant progress made in this field, there is still room for improvement in terms of accuracy, generalization, and robustness of the learned representations. In this dissertation, we have explored the use of auxiliary knowledge sources, including text, knowledge graphs, and similar datasets, to enhance representation learning.

The use of auxiliary knowledge sources has shown significant potential for improving representation learning. In particular, transfer learning from text data has proven to be a powerful technique for improving the performance of language models in natural language processing tasks. Recent research has shown that pretraining language models on large amounts of text data followed by fine-tuning on downstream tasks can lead to state-of-the-art performance. For example, GPT-3, a language model with 175 billion parameters trained on a large corpus of text data, has shown impressive performance on various language tasks. Moreover, the use of knowledge graphs has shown promising results in improving the accuracy of machine learning models in tasks such as recommendation and fraud detection. Knowledge graphs capture relationships between entities in a structured way, which can be leveraged to improve the accuracy of predictions. Finally, the use of similar datasets has also been explored as a

way to improve representation learning. By incorporating additional similar datasets into the training process, we can provide additional data to help reduce overfitting and improve generalization performance.

However, the success of using auxiliary knowledge sources in representation learning heavily relies on the quality and relevance of the auxiliary knowledge sources. For example, in the case of text data, the quality of the corpus used for pretraining and the relevance of the downstream tasks for fine-tuning are crucial factors that determine the success of the approach. Similarly, in the case of knowledge graphs, the quality of the relationships captured in the graph and their relevance to the downstream task are essential factors. In the case of similar datasets, the datasets must be truly similar in terms of data distribution and task objectives for the approach to be effective. Therefore, careful selection and integration of auxiliary knowledge sources are crucial for the success of these techniques.

In conclusion, the use of auxiliary knowledge sources in representation learning has shown significant potential for improving the performance of machine learning models in various domains. Transfer learning from text data, the use of knowledge graphs, and incorporating similar datasets have shown promising results in improving the accuracy, generalization, and robustness of the learned representations. However, the success of these techniques heavily relies on the quality and relevance of the auxiliary knowledge sources. Future research should focus on exploring new and innovative ways of incorporating auxiliary knowledge sources into representation learning and developing techniques for automatic selection and integration of auxiliary knowledge sources. Overall, the exploration of auxiliary knowledge sources in representation learning is an exciting and promising area of research that can lead to significant advancements in machine learning.

Bibliography

Abavisani, M., & Patel, V. M. (2018). Adversarial domain adaptive subspace clustering. *ICISBA*, 1–8.

Abu-El-Haija, S., Perozzi, B., Kapoor, A., Alipourfard, N., Lerman, K., Harutyunyan, H., Steeg, G. V., & Galstyan, A. (2019). MixHop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. *Proceedings of the 36th International Conference on Machine Learning*, 21–29.

Adi, Y., Kermany, E., Belinkov, Y., Lavi, O., & Goldberg, Y. (2016). Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *ICLR*.

Ahmad, M. A., Borbora, Z., Srivastava, J., & Contractor, N. (2010). Link prediction across multiple social networks. *IEEE International Conference on Data Mining Workshops*, 911–918.

Ahmed, N., Rossi, R. A., Lee, J., Willke, T., Zhou, R., Kong, X., & Eldardiry, H. (2020). Role-based graph embeddings. *IEEE Transactions on Knowledge and Data Engineering*, (11).

Aiello, L. M., Barrat, A., Schifanella, R., Cattuto, C., Markines, B., & Menczer, F. (2012). Friendship prediction and homophily in social media. *ACM Transactions on the Web (TWEB)*, *6*(2), 1–33.

Al Hasan, M., & Zaki, M. J. (2011). A survey of link prediction in social networks. In *Social network data analytics* (pp. 243–275). Springer.

Allen, C., Balazevic, I., & Hospedales, T. (2021). Interpreting knowledge graph relation representation from word embeddings. *ICLR*.

Arduini, M., Noci, L., Pirovano, F., Zhang, C., Shrestha, Y. R., & Paudel, B. (2020). Adversarial learning for debiasing knowledge graph embeddings. *ACM MLG*.

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., & Ives, Z. (2007). Dbpedia: A nucleus for a web of open data. In *The semantic web* (pp. 722–735). Springer.

Bai, T., Zhang, S., Egleston, B. L., & Vucetic, S. (2018). Interpretable representation learning for healthcare via capturing disease progression through time. *KDD*, 43–51.

Balazevic, I., Allen, C., & Hospedales, T. (2019). TuckER: Tensor factorization for knowledge graph completion. *EMNLP-IJCNLP*, 5185–5194.

Bamler, R., Salehi, F., & Mandt, S. (2020). Augmenting and tuning knowledge graph embeddings. *Uncertainty in Artificial Intelligence*, 508–518.

Benson, A. R., Abebe, R., Schaub, M. T., Jadbabaie, A., & Kleinberg, J. (2018). Simplicial closure and higher-order link prediction. *Proceedings of the National Academy of Sciences*, *115*(48), E11221–E11230.

Bhardwaj, P., Kelleher, J., Costabello, L., & O'Sullivan, D. (2021a). Adversarial attacks on knowledge graph embeddings via instance attribution methods. *EMNLP*, 8225–8239.

Bhardwaj, P., Kelleher, J., Costabello, L., & O'Sullivan, D. (2021b). Poisoning knowledge graph embeddings via relation inference patterns. *ACL*, 1875–1888.

Blei, D. M., & Lafferty, J. D. (2006). Dynamic topic models. *ICML*, 113–120.

Bodenreider, O. (2004). The unified medical language system (umls): Integrating biomedical terminology. *Nucleic acids research*, *32*(suppl_1), D267–D270.

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013a). Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 2787–2795.

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013b). Translating embeddings for modeling multi-relational data. *NeurIPS*, 2787–2795.

Bose, A., & Hamilton, W. (2019). Compositional fairness constraints for graph embeddings. *ICML*, 715–724.

Bourli, S., & Pitoura, E. (2020). Bias in knowledge graph embeddings. *ASONAM*, 6–10.

Bussmann, N., Giudici, P., Marinelli, D., & Papenbrock, J. (2020). Explainable ai in fintech risk management. *Frontiers in Artificial Intelligence*, *3*, 26.

Cai, L., & Wang, W. Y. (2018). Kbgan: Adversarial learning for knowledge graph embeddings. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 1470–1480.

Cai, R., Li, Z., Wei, P., Qiao, J., Zhang, K., & Hao, Z. (2019). Learning disentangled semantic representation for domain adaptation. *IJCAI*.

Caliskan, A., Bryson, J. J., & Narayanan, A. (2017). Semantics derived automatically from language corpora contain human-like biases. *Science*, *356*, 183–186.

Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka, E. R., & Mitchell, T. M. (2010). Toward an architecture for never-ending language learning. *Twenty-Fourth AAAI Conference on Artificial Intelligence*.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of artificial intelligence research*, *16*.

Chen, J., Fang, H.-r., & Saad, Y. (2009). Fast approximate knn graph construction for high dimensional data via recursive lanczos bisection. *JMLR*, *10*(9).

Chen, J., Wu, Y., Xu, X., Chen, Y., Zheng, H., & Xuan, Q. (2018). Fast gradient attack on network embedding. *arXiv preprint arXiv:1809.02797*.

Chen, X., & Cardie, C. (2018). Multinomial adversarial networks for multi-domain text classification. *NAACL*, 1226–1240.

Chen, Z., Wang, Y., Zhao, B., Cheng, J., Zhao, X., & Duan, Z. (2020). Knowledge graph completion: A review. *IEEE Access*, *8*, 192435–192456.

Cucala, D. J. T., Grau, B. C., Kostylev, E. V., & Motik, B. (2022). Explainable gnn-based models over knowledge graphs. *ICLR*.

Cucerzan, S. (2007). Large-scale named entity disambiguation based on wikipedia data. *Proceedings of the 2007 joint conference on empirical methods in nat-*

ural language processing and computational natural language learning (EMNLP-CoNLL), 708–716.

Dai, Y., Wang, S., Xiong, N. N., & Guo, W. (2020). A survey on knowledge graph embedding: Approaches, applications and benchmarks. *Electronics*, *9*(5), 750.

Das, M., Li, J., Fosler-Lussier, E., Lin, S., Rust, S., Huang, Y., & Ramnath, R. (2020). Sequence-to-set semantic tagging for complex query reformulation and automated text categorization in biomedical ir using self-attention. *Proceedings of the 19th SIGBioMed Workshop on Biomedical Language Processing*, 14–27.

Daza, D., Cochez, M., & Groth, P. (2020). Inductive entity representations from text via link prediction.

De Cao, N., Izacard, G., Riedel, S., & Petroni, F. (2020). Autoregressive entity retrieval. *International Conference on Learning Representations*.

Deeks, A. (2019). The judicial demand for explainable artificial intelligence. *Columbia Law Review*, *119*(7), 1829–1850.

Dehghani, M., Rothe, S., Alfonseca, E., & Fleury, P. (2017). Learning to attend, copy, and generate for session-based query suggestion. *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 1747–1756.

Deng, Y., Xie, Y., Li, Y., Yang, M., Du, N., Fan, W., Lei, K., & Shen, Y. (2019). Multi-task learning with multi-view attention for answer selection and knowledge base question answering. *AAAI*, *33*.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. *NAACL*, 4171–4186.

Do, B.-N., & Rehbein, I. (2020). Neural reranking for dependency parsing: An evaluation. *Association for Computational Linguistics, ACL*.

Doddington, G. R., Mitchell, A., Przybocki, M. A., Ramshaw, L. A., Strassel, S. M., & Weischedel, R. M. (2004). The automatic content extraction (ace) program-tasks, data, and evaluation. *Lrec*, *2*(1), 837–840.

Dooley, D. M., & Griffiths, E. J. e. a. (2018). Foodon: A harmonized food ontology to increase global food traceability, quality control and data integration. *Science of Food*, *2*(1).

Du, Y., Zheng, Q., Wu, Y., Lan, M., Yang, Y., & Ma, M. (2022). Understanding gender bias in knowledge base embeddings. *ACL*, 1381–1395.

Duan, L., Ma, S., Aggarwal, C., Ma, T., & Huai, J. (2017). An ensemble approach to link prediction. *Transactions on Knowledge and Data Engineering*, *29*(11), 2402–2416.

Erhan, D., Courville, A., Bengio, Y., & Vincent, P. (2010). Why does unsupervised pre-training help deep learning? *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 201–208.

Ettorre, A., Bobasheva, A., Faron, C., & Michel, F. (2021). A systematic approach to identify the information captured by knowledge graph embeddings. *IEEE/WIC/ACM WI-IAT*, 617–622.

Ettorre, A., Bobasheva, A., Michel, F., & Faron, C. (2022). Stunning doodle: A tool for joint visualization and analysis of knowledge graphs and graph embeddings. *ESWC*, 370–386.

Fellbaum, C. (1998). *Wordnet: An electronic lexical database*. MIT Press.

Feng, S. Y., Gangal, V., Kang, D., Mitamura, T., & Hovy, E. (2020). Genaug: Data augmentation for finetuning text generators. *DeepIO*.

Feng, S. Y., Gangal, V., Wei, J., Chandar, S., Vosoughi, S., Mitamura, T., & Hovy, E. (2021). A survey of data augmentation approaches for nlp. *ACL-IJCNLP*.

Ferreira, L. A., Guimarães, F. G., & Silva, R. (2020). Applying genetic programming to improve interpretability in machine learning models. *IEEE CEC*, 1–8.

Fisher, J., Mittal, A., Palfrey, D., & Christodoulopoulos, C. (2020). Debiasing knowledge graph embeddings. *EMNLP*, 7332–7345.

Fisher, J., Palfrey, D., Christodoulopoulos, C., & Mittal, A. (2019). Measuring social bias in knowledge graph embeddings. *arXiv preprint arXiv:1912.02761*.

Fisher, J., Palfrey, D., Christodoulopoulos, C., & Mittal, A. (2020). Measuring social bias in knowledge graph embeddings. *KG-BIAS workshop, AKBC*.

Gabrilovich, E., Ringgaard, M., & Subramanya, A. (2013). Facc1: Freebase annotation of clueweb corpora, version 1 (release date 2013-06-26, format version 1, correction level 0) [Accessed: 2020-10-15].

Gad-Elrab, M. H., Stepanova, D., Tran, T.-K., Adel, H., & Weikum, G. (2020). Excut: Explainable embedding-based clustering over knowledge graphs. *ISWC 2020*, 218–237.

Gad-Elrab, M. H., Ho, V. T., Levinkov, E., Tran, T.-K., & Stepanova, D. (2020). Towards utilizing knowledge graph embedding models for conceptual clustering. *ISWC*, 281–286.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., & Lempitsky, V. (2016). Domain-adversarial training of neural networks. *UMLR*, *17*(1), 2096–2030.

Gesese, G. A., Alam, M., & Sack, H. (2020). Semantic entity enrichment by leveraging multilingual descriptions for link prediction.

Ghassemi, M., Oakden-Rayner, L., & Beam, A. L. (2021). The false hope of current approaches to explainable artificial intelligence in health care. *The Lancet Digital Health*, *3*(11), e745–e750.

Grundkiewicz, R., Junczys-Dowmunt, M., & Heafield, K. (2019). Neural grammatical error correction systems with unsupervised pre-training on synthetic data. *ACL BEA*.

Gu, J., Zhao, H., Lin, Z., Li, S., Cai, J., & Ling, M. (2019). Scene graph generation with external knowledge and image reconstruction. *CVPR*, 1969–1978.

Gu, Y., Tinn, R., Cheng, H., Lucas, M., Usuyama, N., Liu, X., Naumann, T., Gao, J., & Poon, H. (2021). Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare*, *3*(1).

Guan, R., Zhang, H., Liang, Y., Giunchiglia, F., Huang, L., & Feng, X. (2020). Deep feature-based text clustering and its explanation. *IEEE Transactions on Knowledge and Data Engineering*.

Guo, Q., Zhuang, F., Qin, C., Zhu, H., Xie, X., Xiong, H., & He, Q. (2020). A survey on knowledge graph-based recommender systems. *TKDE*, *34*(8), 3549–3568.

Guo, Z., & Barbosa, D. (2018). Robust named entity disambiguation with random walks. *Semantic Web*, *9*(4), 459–479.

Hao, Y., Cao, X., Fang, Y., Xie, X., & Wang, S. (2020). Inductive link prediction for nodes having only attribute information. *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, *3*(4), 5.

Hasibi, F., Balog, K., & Bratsberg, S. E. (2017). Dynamic factual summaries for entity cards. *SIGIR*, 773–782.

Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Hirsch, S., Guy, I., Nus, A., Dagan, A., & Kurland, O. (2020). Query reformulation in e-commerce search. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1319–1328.

Hoffart, J., Yosef, M. A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., Taneva, B., Thater, S., & Weikum, G. (2011). Robust disambiguation of named entities in text. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 782–792.

Howard, J., & Ruder, S. (2018). Universal language model fine-tuning for text classification. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 328–339.

Huang, L., Wu, L., & Wang, L. (2020). Knowledge graph-augmented abstractive summarization with semantic-driven cloze reward. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 5094–5107.

Huang, X., Zhang, J., Li, D., & Li, P. (2019). Knowledge graph embedding based question answering. *WSDM*, 105–113.

Jaderberg, M., Mnih, V., Czarnecki, W. M., Schaul, T., Leibo, J. Z., Silver, D., & Kavukcuoglu, K. (2017). Reinforcement learning with unsupervised auxiliary tasks. *International Conference on Learning Representations*.

Jain, N., Kalo, J.-C., Balke, W.-T., & Krestel, R. (2021). Do embeddings actually capture knowledge graph semantics? *ESWC*, 143–159.

Jansen, B. J., Spink, A., Blakely, C., & Koshman, S. (2007). Defining a session on web search engines. *Journal of the American Society for Information Science and Technology*, *58*(6), 862–871.

Ji, S., Pan, S., Cambria, E., Marttinen, P., & Philip, S. Y. (2021). A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*.

Jia, R., Cao, Y., Tang, H., Fang, F., Cao, C., & Wang, S. (2020). Neural extractive summarization with hierarchical attentive heterogeneous graph network. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 3622–3631.

Jiang, J., Liu, L.-P., & Hassoun, S. (2020). Learning graph representations of biochemical networks and its application to enzymatic link prediction.

Jiang, J.-Y., & Wang, W. (2018). Rin: Reformulation inference network for context-aware query suggestion. *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 197–206.

Jiang, X., Ji, P., & Li, S. (2019). Censnet: Convolution with edge-node switching in graph neural networks. *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 2656–2662.

Jiang, X., Zhu, R., Ji, P., & Li, S. (2020). Co-embedding of nodes and edges with graph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Jiang, X., Zhu, R., Li, S., & Ji, P. (2020). Co-embedding of nodes and edges with graph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Jin, D., Jin, Z., Zhou, J. T., & Szolovits, P. (2020). Is bert really robust? a strong baseline for natural language attack on text classification and entailment. *AAAI*, *34*.

Karpathy, A., & Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3128–3137.

Kartsaklis, D., Pilehvar, M. T., & Collier, N. (2018). Mapping text to knowledge graph entities using multi-sense lstms. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 1959–1970.

Kazemi, S. M., & Poole, D. (2018). Simple embedding for link prediction in knowledge graphs. *Advances in neural information processing systems*, 4284–4295.

Kenter, T., & De Rijke, M. (2015). Short text similarity with word embeddings. *CIKM*.

Kipf, T. N., & Welling, M. (2016a). Semi-supervised classification with graph convolutional networks. *ICLR*.

Kipf, T. N., & Welling, M. (2016b). Variational graph auto-encoders. *Bayesian Deep Learning Workshop (NIPS 2016)*.

Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations*.

Kornblith, S., Shlens, J., & Le, Q. V. (2019). Do better imagenet models transfer better? *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2661–2671.

Kraft, A., & Usbeck, R. (2022). The lifecycle of "facts": A survey of social bias in knowledge graphs. *PAC ACL*, 639–652.

Krippendorff, K. (2018). *Content analysis: An introduction to its methodology*. Sage publications.

Kumar, A., Mishra, S., Singh, S. S., Singh, K., & Biswas, B. (2020). Link prediction in complex networks based on significance of higher-order path index (shopi). *Physica A: Statistical Mechanics and its Applications*, *545*, 123790.

Kumar, J., Shao, J., Uddin, S., & Ali, W. (2020). An online semantic-enhanced dirichlet model for short text stream clustering. *Proceedings of the 58th annual meeting of the Association for Computational Linguistics*, 766–776.

Kunegis, J., De Luca, E. W., & Albayrak, S. (2010). The link prediction problem in bipartite networks. *International Conference on Information Processing and Management of Uncertainty in Knowledge-based Systems*, 380–389.

Laskar, M. T. R., Huang, X., & Hoque, E. (2020). Contextualized embeddings based transformer encoder for sentence similarity modeling in answer selection task. *LREC*, 5505–5514.

Lau, J. H., Newman, D., & Baldwin, T. (2014). Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. *EACL*, 530–539.

Lawrence, C., Sztyler, T., & Niepert, M. (2021). Explaining neural matrix factorization with gradient rollback. *AAAI*, *35*, 4987–4995.

Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., & Kang, J. (2020). Biobert: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, *36*(4).

Li, L., Deng, H., Dong, A., Chang, Y., Baeza-Yates, R., & Zha, H. (2017). Exploring query auto-completion and click logs for contextual-aware web search and query suggestion. *WWW*, 539–548.

Li, R., Li, L., Wu, X., Zhou, Y., & Wang, W. (2019). Click feedback-aware query recommendation using adversarial examples. *The World Wide Web Conference*, 2978–2984.

Li, S., & Fu, Y. (2016). Unsupervised transfer learning via low-rank coding for image clustering. *IJCNN*, 1795–1802.

Lin, S.-C., Yang, J.-H., Nogueira, R., Tsai, M.-F., Wang, C.-J., & Lin, J. (2020). Query reformulation using query history for passage retrieval in conversational search.

Lin, W.-A., Chen, J.-C., Castillo, C. D., & Chellappa, R. (2018). Deep density clustering of unconstrained faces. *CVPR*, 8128–8137.

Lin, Y., Liu, Z., Sun, M., Liu, Y., & Zhu, X. (2015). Learning entity and relation embeddings for knowledge graph completion. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2181–2187.

Lin, Y., Shen, S., Liu, Z., Luan, H., & Sun, M. (2016). Neural relation extraction with selective attention over instances. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2124–2133.

Lissandrini, M., Mottin, D., Palpanas, T., & Velegrakis, Y. (2020). Graph-query suggestions for knowledge graph exploration. *Proceedings of The Web Conference 2020*, 2549–2555.

Liu, W., Zhou, P., Zhao, Z., Wang, Z., Ju, Q., Deng, H., & Wang, P. (2020). K-bert: Enabling language representation with knowledge graph. *AAAI*, *34*.

Lloyd, S. (1982). Least squares quantization in pcm. *IEEE transactions on information theory*, *28*(2), 129–137.

Longpre, S., Wang, Y., & DuBois, C. (2020). How effective is task-agnostic data augmentation for pretrained transformers? *EMNLP*.

Malaviya, C., Bhagavatula, C., Bosselut, A., & Choi, Y. (2020). Commonsense knowledge base completion with structural and semantic context. *Proceedings of the 34th AAAI Conference on Artificial Intelligence*.

Marcheggiani, D., & Titov, I. (2020). Graph convolutions over constituent trees for syntax-aware semantic role labeling. *EMNLP*, 3915–3928.

Marcus, M., Kim, G., Marcinkiewicz, M. A., MacIntyre, R., Bies, A., Ferguson, M., Katz, K., & Schasberger, B. (1994). The penn treebank: Annotating predicate argument structure. *Human Language Technology*.

Mavromatis, C., & Karypis, G. (2020). Graph infoclust: Leveraging cluster-level node information for unsupervised graph representation learning. *arXiv preprint arXiv:2009.06946*.

McCann, B., Bradbury, J., Xiong, C., & Socher, R. (2017). Learned in translation: Contextualized word vectors. *NIPS*.

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., & Galstyan, A. (2021). A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, *54*, 1–35.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *NIPS*, *26*.

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

Milne, D., & Witten, I. H. (2008). Learning to link with wikipedia. *Proceedings of the 17th ACM conference on Information and knowledge management*, 509–518.

Mintz, M., Bills, S., Snow, R., & Jurafsky, D. (2009). Distant supervision for relation extraction without labeled data. *ACL*, 1003–1011.

Moon, C., Jones, P., & Samatova, N. F. (2017). Learning entity type embeddings for knowledge graph completion. *CIKM*, 2215–2218.

Mou, L., Lu, X., Li, X., & Zhu, X. X. (2020). Nonlocal graph convolutional networks for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*.

Navigli, R., & Ponzetto, S. P. (2012). BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, *193*, 217–250.

Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., & Ng, A. Y. (2011). Multimodal deep learning. *Proceedings of the 28th international conference on machine learning (ICML-11)*, 689–696.

Otto, C., Wang, D., & Jain, A. K. (2017). Clustering millions of faces by identity. *IEEE transactions on pattern analysis and machine intelligence*, *40*(2), 289–303.

Pachev, B., & Webb, B. (2018). Fast link prediction for large networks using spectral embedding. *Journal of Complex Networks*, *6*(1), 79–94.

Pan, S., Hu, R., Long, G., Jiang, J., Yao, L., & Zhang, C. (2018). Adversarially regularized graph autoencoder for graph embedding. *Proceedings of*

the 27th International Joint Conference on Artificial Intelligence, 2609–2615.

Pan, S. J., & Yang, Q. (2009). A survey on transfer learning. *TKDE*, *22*(10), 1345–1359.

Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, *22*(10), 1345–1359.

Pass, G., Chowdhury, A., & Torgeson, C. (2006). A picture of search. *Proceedings of the 1st international conference on Scalable information systems*, 1–es.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *EMNLP*.

Pezeshkpour, P., Irvine, C., Tian, Y., & Singh, S. (2019). Investigating robustness and interpretability of link prediction via adversarial modifications. *NAACL*, 3336–3347.

Pujara, J., Augustine, E., & Getoor, L. (2017). Sparsity and noise: Where knowledge graph embeddings fall short. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 1751–1756.

Qin, Z., Cen, C., Jie, W., Gee, T. S., Chandrasekhar, V. R., Peng, Z., & Zeng, Z. (2018). Knowledge-graph based multi-target deep-learning models for train anomaly detection. *2018 International Conference on Intelligent Rail Transportation (ICIRT)*, 1–5.

Radstok, W., Chekol, M., Schaefer, M., et al. (2021). Are knowledge graph embedding models biased, or is it the data that they are trained on? *ISWC*.

Ratinov, L., Roth, D., Downey, D., & Anderson, M. (2011). Local and global algorithms for disambiguation to wikipedia. *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, 1375–1384.

Reinanda, R., Meij, E., de Rijke, M., et al. (2020). Knowledge graphs: An information retrieval perspective. *Information Retrieval*, *14*(4), 289–444.

Reisinger, J., & Mooney, R. (2010). Multi-prototype vector-space models of word meaning. *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 109–117.

Rezayi, S., Lipka, N., Vinay, V., Rossi, R. A., Dernoncourt, F., King, T. H., & Li, S. (2021). A framework for knowledge-derived query suggestions. *BigData*, 510–518.

Rezayi, S., Soleymani, S., Arabnia, H. R., & Li, S. (2021). Socially aware multimodal deep neural networks for fake news classification. *MIPR*, 253–259.

Rezayi, S., Zhao, H., Kim, S., Rossi, R., Lipka, N., & Li, S. (2021). Edge: Enriching knowledge graph embeddings with external text. *NAACL*, 2767–2776.

Rezayi, S., Zhao, H., & Li, S. (2022). Xdc: Adversarial adaptive cross domain face clustering (student abstract). *AAAI*, *36*, 13035–13036.

Rosset, C., Xiong, C., Song, X., Campos, D., Craswell, N., Tiwary, S., & Bennett, P. (2020). Leading conversational search by suggesting useful questions. *Proceedings of The Web Conference*, 1160–1170.

Rossi, A., Firmani, D., Matinata, A., Merialdo, P., & Barbosa, D. (2020). Knowledge graph embedding for link prediction: A comparative analysis.

Rossi, A., Firmani, D., Merialdo, P., & Teofili, T. (2022). Explaining link prediction systems based on knowledge graph embeddings. *SIGMOD*, 2062–2075.

Rossi, R. A., Rao, A., Kim, S., Koh, E., & Ahmed, N. K. (2020). From closing triangles to closing higher-order motifs. *Proceedings of The Web Conference (WWW)*.

Rossi, R. A., Zhou, R., & Ahmed, N. K. (2018). Deep inductive network representation learning. *WWW*, 953–960.

Rosso, P., Yang, D., & Cudré-Mauroux, P. (2020). Beyond triplets: Hyper-relational knowledge graph embedding for link prediction. *Proceedings of The Web Conference*, 1885–1896.

Rücklé, A., Moosavi, N. S., & Gurevych, I. (2019). Coala: A neural coverage-based approach for long answer selection with small data. *AAAI*, *33*.

Salha, G., Hennequin, R., & Vazirgiannis, M. (2020). Simple and effective graph autoencoders with one-hop linear models.

Salha, G., Limnios, S., Hennequin, R., Tran, V.-A., & Vazirgiannis, M. (2019). Gravity-inspired graph autoencoders for directed link prediction. *CIKM*, 589–598.

Samanta, S., Selvan, A. T., & Das, S. (2013). Cross-domain clustering performed by transfer of knowledge across domains. *NCVPRIPG*, 1–4.

Sengupta, T., Pragadeesh, C., Talukdar, P., et al. (2020). Inducing interpretability in knowledge graph embeddings. *ICON*, 70–75.

Shao, J., Zhang, Z., Yu, Z., Wang, J., Zhao, Y., & Yang, Q. (2019). Community detection and link prediction via cluster-driven low-rank matrix completion. *IJCAI*, 3382–3388.

Sharma, C., Chauhan, J., & Kaul, M. (2020). Learning representations using spectral-biased random walks on graphs.

Sheu, H.-S., & Li, S. (2020). Context-aware graph embedding for session-based news recommendation. *Fourteenth ACM Conference on Recommender Systems*, 657–662.

Shi, C., Ding, J., Cao, X., Hu, L., Wu, B., & Li, X. (2020). Entity set expansion in knowledge graph: A heterogeneous information network perspective. *Frontiers of Computer Science*, *15*(1), 1–12.

Shi, H., Livescu, K., & Gimpel, K. (2021). Substructure substitution: Structured data augmentation for nlp. *ACL-IJCNLP*.

Shi, J., & Malik, J. (2000). Normalized cuts and image segmentation. *TPAMI*, *22*(8), 888–905.

Shou, L., Wang, Z., Chen, K., & Chen, G. (2013). Sumblr: Continuous summarization of evolving tweet streams. *SIGIR*, 533–542.

Silpa-Anan, C., & Hartley, R. (2008). Optimised kd-trees for fast image descriptor matching. *CVPR*, 1–8.

Simov, K., Popov, A., & Osenova, P. (2016). The role of the wordnet relations in the knowledge-based word sense disambiguation task. *GWC*, 396–403.

Sun, H., Dhingra, B., Zaheer, M., Mazaitis, K., Salakhutdinov, R., & Cohen, W. W. (2018). Open domain question answering using early fusion of knowledge bases and text. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 4231–4242.

Sun, Z., Deng, Z.-H., Nie, J.-Y., & Tang, J. (2019a). Rotate: Knowledge graph embedding by relational rotation in complex space. *International Conference on Learning Representations*.

Sun, Z., Deng, Z.-H., Nie, J.-Y., & Tang, J. (2019b). Rotate: Knowledge graph embedding by relational rotation in complex space. *ICLR*.

Sung, M., Jeong, M., Choi, Y., Kim, D., Lee, J., & Kang, J. (2022). Bern2: An advanced neural biomedical named entity recognition and normalization tool. *Bioinformatics*, *38*(20), 4837–4839.

Tay, Y., Luu, A. T., & Hui, S. C. (2017). Non-parametric estimation of multiple embeddings for link prediction on dynamic knowledge graphs. *AAAI*, 1243–1249.

Tay, Y., Luu, A. T., Hui, S. C., & Brauer, F. (2017). Random semantic tensor ensemble for scalable knowledge graph link prediction. *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, 751–760.

Toutanova, K., Chen, D., Pantel, P., Poon, H., Choudhury, P., & Gamon, M. (2015). Representing text for joint embedding of text and knowledge bases. *Proceedings of the 2015 conference on empirical methods in natural language processing*, 1499–1509.

Tran, P. V. (2018). Learning to make predictions on graphs with autoencoders. *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, 237–245.

Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., & Bouchard, G. (2016). Complex embeddings for simple link prediction.

Tzeng, E., Hoffman, J., Saenko, K., & Darrell, T. (2017). Adversarial discriminative domain adaptation. *CVPR*, 7167–7176.

Van Gansbeke, W., Vandenhende, S., Georgoulis, S., Proesmans, M., & Van Gool, L. (2020). Scan: Learning to classify images without labels. *ECCV*, 268–285.

Vashishth, S., Joshi, R., Prayaga, S. S., Bhattacharyya, C., & Talukdar, P. (2018). Reside: Improving distantly-supervised neural relation extraction using side information. *EMNLP*, 1257–1266.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018a). Graph attention networks. *International Conference on Learning Representations*.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018b). Graph attention networks. *ICLR*.

Verma, G., Vinay, V., Bansal, S., Oberoi, S., Sharma, M., & Gupta, P. (2020). Using image captions and multitask learning for recommending query reformulations. *European Conference on Information Retrieval*, 681–696.

Vrandečić, D., & Krötzsch, M. (2014). Wikidata: A free collaborative knowledgebase. *Communications of the ACM*, *57*(10), 78–85.

Wang, Q., Mao, Z., Wang, B., & Guo, L. (2017). Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, *29*(12), 2724–2743.

Wang, R., Li, B., Hu, S., Du, W., & Zhang, M. (2019). Knowledge graph embedding via graph attenuated attention networks. *IEEE Access*, *8*, 5212–5224.

Wang, X., He, X., Cao, Y., Liu, M., & Chua, T.-S. (2019). Kgat: Knowledge graph attention network for recommendation. *KDD*, 950–958.

Wang, X., Macdonald, C., & Ounis, I. (2020). Deep reinforced query reformulation for information retrieval.

Wang, Z., Zhang, J., Feng, J., & Chen, Z. (2014). Knowledge graph embedding by translating on hyperplanes. *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, 1112–1119.

Wang, Z., & Li, J. (2016). Text-enhanced representation learning for knowledge graph. *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, 1293–1299.

Wang, Z., Zheng, L., Li, Y., & Wang, S. (2019). Linkage based face clustering via graph convolution network. *CVPR*, 1117–1125.

Wei, J., Huang, C., Vosoughi, S., Cheng, Y., & Xu, S. (2021). Few-shot text classification with triplet networks, data augmentation, and curriculum learning. *NAACL*.

Wei, J., & Zou, K. (2019). Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *EMNLP-IJCNLP*.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., … Rush, A. M. (2020). Huggingface's transformers: State-of-the-art natural language processing.

Wu, L., Petroni, F., Josifoski, M., Riedel, S., & Zettlemoyer, L. (2020a). Scalable zero-shot entity linking with dense entity retrieval.

Wu, L., Petroni, F., Josifoski, M., Riedel, S., & Zettlemoyer, L. (2020b). Scalable zero-shot entity linking with dense entity retrieval. *EMNLP*, 6397–6407.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Philip, S. Y. (2020). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*.

Xia, M., Kong, X., Anastasopoulos, A., & Neubig, G. (2019). Generalized data augmentation for low-resource translation. *ACL*.

Xian, Y., Fu, Z., Zhao, H., Ge, Y., Chen, X., Huang, Q., Geng, S., Qin, Z., de Melo, G., Muthukrishnan, S., & Zhang, Y. (2020). CAFE: coarse-to-fine neural symbolic reasoning for explainable recommendation. In M. d'Aquin, S. Dietze, C. Hauff, E. Curry, & P. Cudré-Mauroux (Eds.), *The 29th ACM international conference on information and knowledge management (cikm)* (pp. 1645–1654). ACM.

Xiong, C., Power, R., & Callan, J. (2017). Explicit semantic ranking for academic search via knowledge graph embedding. *WWW*, 1271–1279.

Yahoo. (2013). Yahoo search query log [Accessed: 2020-10-16].

Yan, J., Raman, M., Zhang, T., Rossi, R. A., Zhao, H., Kim, S., Lipka, N., & Ren, X. (2020). Learning contextualized knowledge structures for commonsense reasoning. *CoRR*, *abs/2010.12873*. https://arxiv.org/abs/2010.12873

Yang, B., Yih, W.-t., He, X., Gao, J., & Deng, L. (2015). Embedding entities and relations for learning and inference in knowledge bases. *International Conference on Learning Representations*.

Yang, K., & Deng, J. (2020). Strongly incremental constituency parsing with graph neural networks. *Advances in Neural Information Processing Systems*, *33*, 21687–21698.

Yang, L., Zhan, X., Chen, D., Yan, J., Loy, C. C., & Lin, D. (2019). Learning to cluster faces on an affinity graph. *CVPR*, 2298–2306.

Yang, W., Xie, Y., Lin, A., Li, X., Tan, L., Xiong, K., Li, M., & Lin, J. (2019). End-to-end open-domain question answering with bertserini. *NAACL*.

Yang, Y., Lichtenwalter, R. N., & Chawla, N. V. (2015). Evaluating link prediction methods. *Knowledge and Information Systems*, *45*(3), 751–782.

Yani, M., & Krisnadhi, A. A. (2021). Challenges, techniques, and trends of simple knowledge graph question answering: A survey. *Information*, *12*, 271.

Yao, L., Mao, C., & Luo, Y. (2019). Graph convolutional networks for text classification. *AAAI*, *33*, 7370–7377.

Yih, S. W.-t., Chang, M.-W., Meek, C., & Pastusiak, A. (2013). Question answering using enhanced lexical semantic models. *ACL*.

Yin, J., Chao, D., Liu, Z., Zhang, W., Yu, X., & Wang, J. (2018). Model-based clustering of short text streams. *KDD*, 2634–2642.

Yin, J., & Wang, J. (2014). A dirichlet multinomial mixture model-based approach for short text clustering. *KDD*, 233–242.

Yin, J., & Wang, J. (2016). A text clustering algorithm using an online clustering scheme for initialization. *KDD*, 1995–2004.

Yu, S., Wang, Y., Yang, M., Li, B., Qu, Q., & Shen, J. (2019). Nairs: A neural attentive interpretable recommendation system. *WSDM*.

Yuan, F., Yao, L., & Benatallah, B. (2019). Darec: Deep domain adaptation for cross-domain recommendation via transferring rating patterns. *arXiv*.

Yuan, H., Yu, H., Gui, S., & Ji, S. (2022). Explainability in graph neural networks: A taxonomic survey. *IEEE TPAMI*.

Zeng, D., Liu, K., Chen, Y., & Zhao, J. (2015). Distant supervision for relation extraction via piecewise convolutional neural networks. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 1753–1762.

Zhan, X., Liu, Z., Yan, J., Lin, D., & Change Loy, C. (2018). Consensus-driven propagation in massive unlabeled data for face recognition. *ECCV*, 568–583.

Zhang, C., Li, Q., & Song, D. (2019). Aspect-based sentiment classification with aspect-specific graph convolutional networks. *EMNLP-IJCNLP*, 4568–4578.

Zhang, D., Nan, F., Wei, X., Li, S.-W., Zhu, H., McKeown, K., Nallapati, R., Arnold, A. O., & Xiang, B. (2021). Supporting clustering with contrastive learning. *NAACL*, 5419–5430.

Zhang, H., Zheng, T., Gao, J., Miao, C., Su, L., Li, Y., & Ren, K. (2019). Data poisoning attack against knowledge graph embedding. *IJCAI*, 4853–4859.

Zhang, M., Wang, Q., Xu, W., Li, W., & Sun, S. (2018). Discriminative path-based knowledge graph embedding for precise link prediction. *European Conference on Information Retrieval*, 276–288.

Zhang, M., & Chen, Y. (2018). Link prediction based on graph neural networks. *NIPS*, 5165–5175.

Zhang, M., Cui, Z., Jiang, S., & Chen, Y. (2018). Beyond link prediction: Predicting hyperlinks in adjacency space. *AAAI*, 6.

Zhang, W., Paudel, B., Wang, L., Chen, J., Zhu, H., Zhang, W., Bernstein, A., & Chen, H. (2019). Iteratively learning embeddings and rules for knowledge graph reasoning. *The World Wide Web Conference*, 2366–2377.

Zhao, J., Wang, T., Yatskar, M., Ordonez, V., & Chang, K.-W. (2018). Gender bias in coreference resolution: Evaluation and debiasing methods. *NAACL*.

Zhao, Y., Zhang, A., Xie, R., Liu, K., & Wang, X. (2020). Connecting embeddings for knowledge graph entity typing. *ACL*, 6419–6428.

Zhong, J., Guo, W., Gao, H., & Long, B. (2020). Personalized query suggestions. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1645–1648.

Zhou, D., Hu, X., & Wang, R. (2020). Neural topic modeling by incorporating document relationship graph. *EMNLP*, 3790–3796.

Zhu, C., Wen, F., & Sun, J. (2011). A rank-order distance based clustering algorithm for face tagging. *CVPR*, 481–488.

Zhu, R., Jiang, X., Lu, J., & Li, S. (2021a). Cross-domain graph convolutions for adversarial unsupervised domain adaptation. *IEEE Transactions on Neural Networks and Learning Systems*.

Zhu, R., Jiang, X., Lu, J., & Li, S. (2021b). Transferable feature learning on graphs across visual domains. *ICME*, 1–6.

Zhu, R., & Li, S. (2022). Self-supervision based semantic alignment for unsupervised domain adaptation. *SDM*, 1–9.

Zhu, R., Tao, Z., Li, Y., & Li, S. (2021). Automated graph learning via population based self-tuning gcn. *SIGIR*, 2096–2100.

Zou, D., & Lerman, G. (2019). Encoding robust representation for graph generation. *2019 International Joint Conference on Neural Networks (IJCNN)*, 1–9.

Zügner, D., Akbarnejad, A., & Günnemann, S. (2018). Adversarial attacks on neural networks for graph data. *KDD*, 2847–2856.