Designing Big Data Frameworks for Quality-of-Data Controlling in Large-Scale Knowledge Graphs

Hussein Baalbaki

SORBONNE UNIVERSITY

DOCTORAL SCHOOL

Research laboratory LISITE@ISEP

T H E S I S

to obtain the degree of

DOCTOR FROM SORBONNE UNIVERSITY

Discipline : Computer science

Presented and supported by :

Hussein BAALBAKI

22 december 2023

Designing Big Data Frameworks for Quality-of-Data Controlling in Large-Scale Knowledge Graphs

Under the direction of :

Mme. Lina Mroueh – Director, ISEP

M. Rafael Angarita – Co-encadrant, Université Paris Nanterre | LIP6 Sorbonne Université

M. Hassan Harb – Co-encadrant, Lebanese University

M. Hussein Hazimeh – Co-encadrant, Lebanese University

Reviewers :

Mme Marta Rukoz – Professor, LAMSADE | Université Paris Nanterre

M. Ali Kadhum Idrees - Professor, University of Babylon

Jury president:

M. Jean-Francois Pradat-Peyre – Professor, Université Paris Nanterre | LIP6 Sorbonne Université

Jury members:

M. Ali Jaber – Professor, Lebanese University

Mme Shohreh Ahvar – Researcher at Nokia-Bell-Labs

M. Mourad Ouziri – Professor, Paris Descartes Computer Science Laboratory

Abstract

Les Knowledge Graphs (KG) sont la représentation la plus utilisée dínformations structurées sur un domaine particulier, composée de milliards de faits sous la forme déntités (nœuds) et de relations (bords) entre eux. De plus, les informations de type sémantique des entités sont également contenues dans les KG. Le nombre de KG ná cessé dáugmenter au cours des 20 dernières années dans divers domaines, notamment le gouvernement, la recherche universitaire, les domaines biomédicaux, etc. Les applications basées sur lápprentissage automatique qui utilisent les KG incluent la liaison déntités, les systèmes de questions-réponses, les systèmes de recommandation, etc. Les Open KG sont généralement produits de manière heuristique, automatiquement à partir de diverses sources, notamment du texte, des photos et dáutres ressources, ou sont sélectionnés manuellement. Cependant, ces KG sont souvent incomplètes, cést-à-dire quíl existe des liens manquants entre les entités et des liens manquants entre les entités et leurs types déntités correspondants. Dans cette thèse, nous abordons l'un des problèmes les plus difficiles auxquels est confronté le Knowledge Graph Completion (KGC), à savoir la prédiction de liens. Prédiction générale des liens en KG qui inclut la prédiction de la tête et de la queue, triple classification.

Ces dernières années, les KGE ont été formés pour représenter les entités et les relations du KG dans un espace vectoriel de faible dimension préservant la structure du graphe. Dans la plupart des travaux publiés tels que les modèles translationnels, les modèles de réseaux neuronaux et autres, la triple information est utilisée pour générer la représentation latente des entités et des relations.

Dans cette thèse, plusieurs méthodes ont été proposées pour KGC et leur efficacité est démontrée empiriquement dans cette thèse. Tout d'abord, un nouveau modèle d'intégration KG, TransModE, est proposé pour la prédiction de liens. TransModE projette les informations contextuelles des entités dans un espace modulaire, tout en considérant la relation comme vecteur de transition qui guide léntité tête vers léntité queue.

Deuxièmement, nous avons travaillé sur la construction dún modèle KGE simple et de faible complexité, tout en préservant son efficacité. KEMA est un nouveau modèle KGE parmi les modèles KGE les plus bas en termes de complexité, tout en obtenant des résultats prometteurs.

Enfin, KEMA++ est proposé comme une mise à niveau de KEMA pour prédire les triplets manquants dans les KG en utilisant lópération arithmétique des produits dans un espace modulaire. Les expériences approfondies et les études dáblation montrent léfficacité du modèle proposé, qui rivalise avec les modèles de pointe actuels et établit de nouvelles références pour KGC.

Knowledge Graphs (KGs) are the most used representation of structured information about a particular domain consisting of billions of facts in the form of entities (nodes) and relations (edges) between them. Additionally, the semantic type information of the entities is also contained

in the KGs. The number of KGs has steadily increased over the past 20 years in a variety of fields, including government, academic research, the biomedical fields, etc. Applications based on machine learning that use KGs include entity linking, question-answering systems, recommender systems, etc. Open KGs are typically produced heuristically, automatically from a variety of sources, including text, photos, and other resources, or are hand-curated. However, these KGs are often incomplete, i.e., there are missing links between the entities and missing links between the entities and their corresponding entity types. In this thesis, we are addressing one of the most challenging issues facing Knowledge Graph Completion (KGC) which is link prediction. General Link Prediction in KGs that include head and tail prediction, triple classification. In recent years, KGE have been trained to represent the entities and relations in the KG in a low-dimensional vector space preserving the graph structure. In most published works such as the translational models, neural network models and others, the triple information is used to generate the latent representation of the entities and relations.

In this dissertation, several methods have been proposed for KGC and their effectiveness is shown empirically in this thesis. Firstly, a novel KG embedding model TransModE is proposed for Link Prediction. TransModE projects the contextual information of the entities to modular space, while considering the relation as transition vector that guide the head to the tail entity. Secondly, we worked on building a simple low comlexity KGE model, meanwhile preserving its efficiency. KEMA is a novel KGE model among the lowest KGE models in terms of complexity, meanwhile it obtain promising results. Finally, KEMA++ is proposed as an upgrade of KEMA to predict the missing triples in KGs using product arithmetic operation in modular space. The extensive experiments and ablation studies show efficiency of the proposed model, which compete the current SoTA models and set new baselines for KGC.

The proposed models establish new way in solving KGC problem other than transitional, neural network, or tensor factorization based approaches. The promising results and observations open up interesting scopes for future research involving exploiting the proposed models in domain-specific KGs such as scholarly data, biomedical data, etc. Furthermore, the link prediction model can be exploited as a base model for the entity alignment task as it considers the neighbourhood information of the entities.

KEYWORDS: *Knowledge Graph , Knowledge Graph Completion, Link Prediction, Knowledge Graph Embedding*

Shorts Contents

Contents

List of Figures

List of Tables

List of Abbreviations

KG: Knowledge Graph
KGC: Knowledge Graph Completion
KGE: Knowledge Graph Embeddings
WN: WordNet
FB: FreeBase

Dedication

I am, dedicating this work to my Mother, Father, and all family, who supported me durinh this journey. I hope this will be a little gift for them, as they always encouraged me to learn and to develop my abilities. Also i am dedicating this work to my lovely wife, She has encouraged and supported me when i faced any obstacle.

Introduction

1 General Introduction

In the third century before Christ's birth, one of the most ambitious projects of knowledge collection in history was Established. it was the famous Great Library of Alexandria. This library was collected through copying all books found on the numerous ships that docked in the Alexandria harbor. This was just an example of the numerous attempts of mankind to centralize knowledge. Such ambitious projects have found a recent embodiment in the internet: Birth of Wikipedia, a global, open-source and collaborative project. Jimmy Wales, a co-founder of Wikipedia presented it in the following terms: "Imagine a world in which every single person on the planet is given free access to the sum of all human knowledge. That's what we are doing.". Wikipedia, now, counts nearly sixty million articles in more than three hundred languages. It is widely used by people around the world. However, the text format in wikipedia limits its benefit in some modern usage. Particularly, text of natural language is not machine understandable, what hinders its processing by modern intelligent systems such as recommender systems or chatbots. In the last decades, knowledge graphs rise to record and organize knowledge by listing entities and verified facts involving these entities. Knowledge graphs differ first according to its accessibility. Some are public such as ConceptNet and Wikidata. Others are private and are simply projects that companies launch to organize their domain expertise. Knowledge graphs also differ according to the specialty. Freebase, for example, contains general knowledge, and not dedicated to one topic. In opposition, others are topic-specific such as WordNet, which specifically records knowledge in the field of linguistics.

Wikidata is now the most unavoidable knowledge graph. It was found in 2012, after organizing the knowledge contained in wikipedia textual articles in the form of a structured knowledge base that would be easily queryable and processable by machines.

Because of their size, knowledge graphs are difficult to process manually. As a result, methods to automatize processing tasks have been developed simultaneously with the knowledge bases themselves. Two examples of such tasks researchers are working to automatize are: the KG completion, and the KG correction. Because of their size, knowledge graphs are difficult to process manually. As a result, methods to automatize processing tasks have been developed simultaneously with the knowledge bases themselves. Two examples of such tasks researchers are working to automatize are: the KG completion, and the KG correction. Despite the first methods were symbolic and primitive, these methods still prove to solve efficiently some problems. They can either be used to raise unusual facts as possible errors or to propose missing facts.

In the same way as first methods, a prominent field in data mining is statistics. Statistics provides a lot of tools to estimate underlying distributions of observed data. An important example of statistics is machine learning, which originally used mainly statistical methods. A turning point was ease access to graphics processing units in the last fifteen years, what takes the development to a higher level,

and give the opportunity to models requiring heavy tensor computations. One of the consequences was the revolutionary performances of neural networks in the field of computer vision. As well as other, knowledge graphs also benefited from this cheap access to computation power and many new machine learning models were proposed to automatically solve tasks such as construction, completion and correction by involving vector representations of knowledge graphs.

This thesis works on the solution of knowledge graphs completion problem. To realize the added values offered by our proposed models, chapter 1 gives a proper introduction to the concepts presented. Chapter 2 introduces the different classifications of KGE models, and shows examples of each classification, clarifying the way it work. In its simplest form, a knowledge graph is a collection of facts in the form of triples (h, r, t) were a relation r links a head entity h to a tail entity t. An example of a fact can be (Alex, child,Emma). Completing a knowledge graph then comes down to find new triples that has a high plausibility to be true given the existing ones. An example is shown in Figure 1.

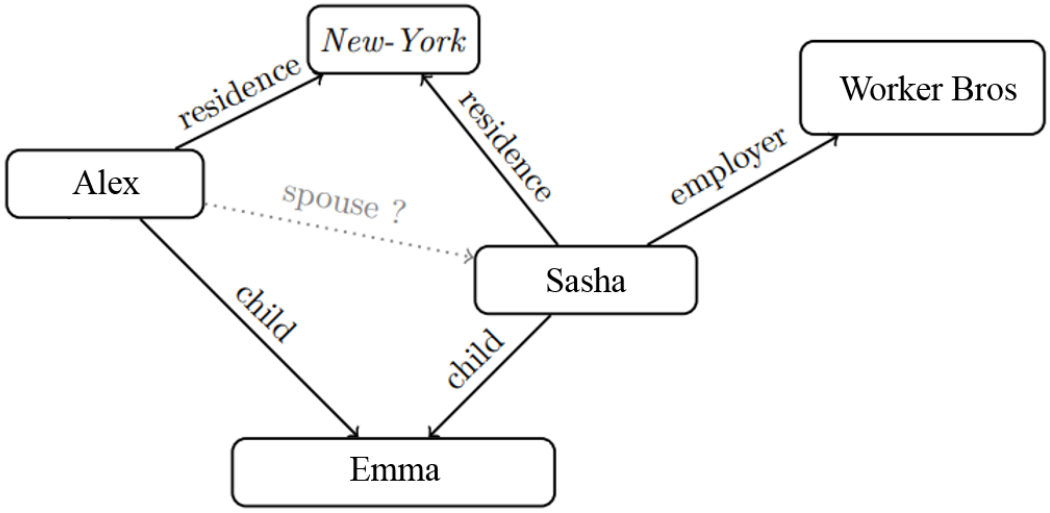


Figure 1 Example of a knowledge graph..

Following these definitions, Chapter 3 introduce a knowledge graph embedding model for knowledge graph completion, and reviews the existing methods. The focus of the chapter is on building a KGE model based on transition combined to modular arithmetic, mainly relying on vector representations, or embeddings, of entities and relations in modular space. This chapter experiments the model TransModE, and shows the promising results compared to existing models, taking into consideration the low complexity of the model.

The proposed models in this thesis are characterized by simplicity. Mainly knowledge graph embedding models must improve its efficiency while taking into consideration to keep its complexity as low as possible. Therefore the characteristics, low complexity and high efficiency, are must for KGE model. In chapter 4, we introduce two models: KEMA and and its enhanced version KEMA++. In this chapter, we work on producing two models each focuses on one of the characteristics while preserving the other. The most prominent feature of KEMA is that it can be considered one of the simplest KGE models in the literature, meanwhile it achieves excellent results compared to its alternatives. On the other hand, KEMA++ focuses on achieving perfect results with a moderate complexity.

Eventually a conclusive chapter puts the contributions of this thesis in perspective with the rest of the field to isolate promising axis for future works.

Semantic network [1] was first built in the 1960s. It was the basis of what was called later Knowledge Graphs (KGs). The term "Knowledge Graph" was first mentioned in literature in 1972 [2], which was later used by Google in 2012 during the announcement of "Google Knowledge Graph" [3]. As defined in [4], a KG describes real-world entities and their interactions in a graph form having a schema built of entities and relations. KG allows potential connections between arbitrary entities with each other and covers various topical domains.

1.1 Knowledge Graph Definition

A knowledge graph KG is a directed multigraph in which nodes are entities and edges, typed by a relation, represent known facts linking two entities. These graphs can encode a wide variety of information. In its simplest form, a knowledge graph consist of facts in the form of triplets (h,r,t) where a relation r links a head entity h to a tail entity t. An example of a fact can be $(Paris,\ capital\ of,\ France)$. Each of the head and the tail of a triplet can be either entity, literal, or label, as the equation shows:
$h,\ t\ \in\ E\ \cup\ L\ \cup\ B$
Where E, L, and B are respectively the set of entities, the set of literals, and the set of labels of G.

1 An Overview About Knowledge Graph Embeddings Domain

1.1 Introduction to knowledge graph

Knowledge graphs allow users to visualize knowledge facts about real-world entities (nodes) and the interrelations between them (edges), stored in the form of RDF triples. They incorporates knowledge from structured repositories such as DBpedia, or by extracting knowledge from semi-structured web resources such as Wikipedia. The term knowledge graph has appeared firstly in 2012, when Google introduced a knowledge panel in their search results. It allows users to visualize consolidated knowledge from heterogeneous data resources such as personal websites and social media channels in a unified and categorized knowledge panel. Similarly, other knowledge graphs such as Wikidata [5], YAGO [6], and PROSPERA [7] did. They allow individuals to visualize knowledge facts about entities in a graphical representation. A user must enter an entity name as a keyword query, and he will get a knowledge graph as an outcome.

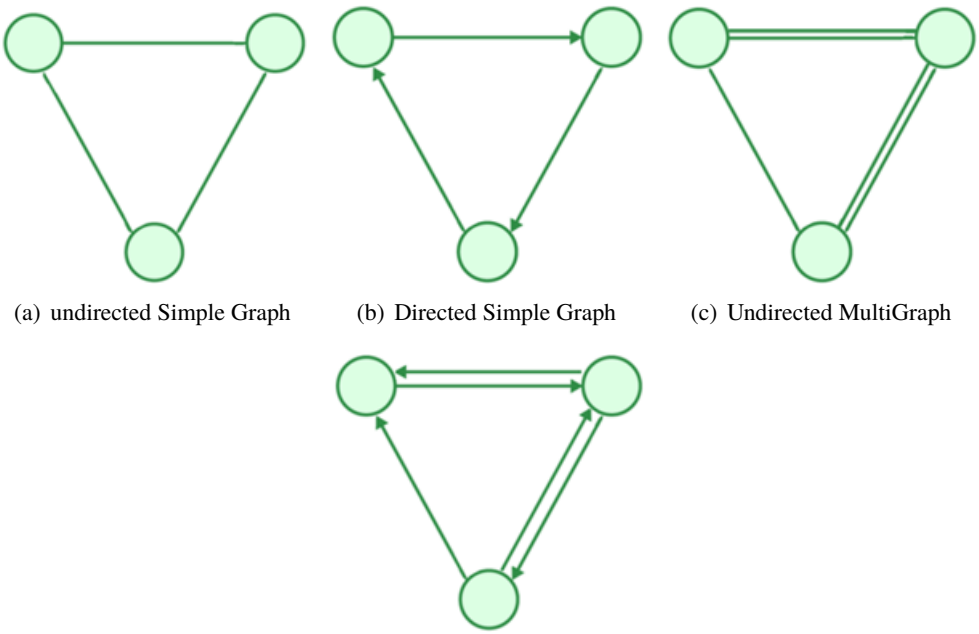
1.1.1 Graph

In the last two decades, the world has witnessed the rise of social media network platforms, connected devices, internet-of-thing IOT, and other copious data sources. As a result, a significant increase in the research of graph-structured data is observed. Graphs are a mathematical representation of a network designed to study, analyze, and learn from complex systems in real-world. A graph is built of group of objects represented by nodes, beside a set of interactions between them represented in the graph by edges. Social network can be an illustrative example of graph, where the users are the nodes, and the interactions in between them like friendship, shares, and others are represented as edges connecting them. Many other fields also use graph formalization for its encoding, like recommender systems, question-answering, etc.

Formally, a graph G is denoted as a pair $G(V,E)$, where V represents a set of nodes and E represents the set edges. The edge in the graph is denoted by the pair of nodes at its ends, for example the edge by $(u,v)E$ is connecting the nodes u and v. By the mean of this edge, these two nodes are called to be adjacent. Whereas 2 edges can be adjacent if they share a common node in between.

Graph types differ according to the connectivity between the nodes and the nature of the edges:
The types of graphs depending on the nature of the edges are:

- Undirected Graph G = (V, E): connect each pair of nodes through an undirected edge denoted as follows:$(u,v) \in E \Leftrightarrow (v,u) \in E$, as shown in Figures Fig.1.1.



(a) undirected Simple Graph (b) Directed Simple Graph (c) Undirected MultiGraph

(d) Directed MultiGraph

Figure 1.1 Types of Graphs.

- Directed Graph G= (V, E): connect each pair of nodes through a directed edges (also called arcs). The edge $e \in E$, *where* $E = (u,v)|(u,v) \in V^2$.

  The types of graphs depending on the nature of the edges are:

  - Simple graph G: it consists of at most one edge between each pair of nodes.
  - Multi-graph G: it consists of multiple edges i.e., edges with the same source and target nodes.

  A hybrid type according to both types of graphs:

  - An Undirected Multi-graph G is an ordered triple $G = (V,E,R)$, where V and E are the set of nodes and edges respectively, and R is an indicator function such that $R : E \Leftrightarrow (u,v)|u,v \in V$, assigning to each edge an unordered pair of endpoint nodes.
  - A Directed Multi-Graph G is an ordered pair $G = (V,E)$, where V is the set of nodes and E is the set of directed edges. It consists of multiple arcs i.e., arcs with the same source and target nodes, as well as self-loops.

1.1.2 Knowledge Graph

1.1.2.1 KG Definition

Although knowledge graph KG has been studied widely in the last decade, it remains one of the most fundamental problems in artificial intelligence (AI) research and data engineering. KGs are large networks consisting of huge amounts of facts organized as entities, represented as nodes and relations given by directed labelled edges connecting the nodes. The facts in a KG are presented in the form of a triplet, the main elements of KG. The triplet < h, r, t > is built of the head entity h, the tail entity t, and r is the relationship connecting them. For instance, the entities Paris, France, and others are represented as nodes, and their relations are represented as directed edges. In the example of knowledge graph is shown in Fig. 1.2, < Paris, capital of, France> is a valid triplet. Apart

from the entities and the relations, the KGs also comprise a special kind of relation that denotes the semantic types of the entities. Semantic types of entities also referred to as classes, are used in KGs to group similar entities together. In Figure 1.2, the semantic type is given by '*is a*' relation, hence the triplet represents that *James* is an instance of class *Person*.

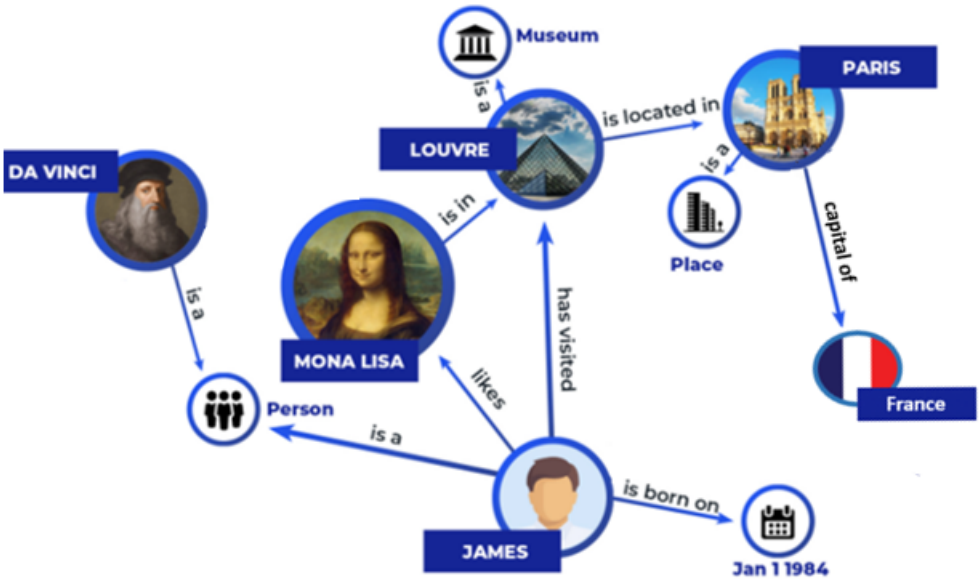


Figure 1.2 Sub graph extracted from FreeBase.

1.1.2.2 Triplet in Knowledge Graph

A triplet (h,r,t) in a KG G consist of $h \in E$ is the subject, $r \in R$ is the relation, and $t \in E$ is the object. The subject is referred to as the head entity and the object as the tail entity. The triplets with literals as objects, namely $t \in L$ are known as attributive triplets. Lastly, the triplets with $t \in B$ represent the classes of the entities.

As mentioned before, an entity can also be linked to a class using a special kind of relation. For example, in Freebase the relation *isA* represents the relation that is used to state that an entity is an instance of a label in the respective KGs. A triplet $(Paris, isA, City)$, states that $City \in B$ is a class and $Paris \in E$ is an entity in G and is an instance of City. The labels in a KG are organized in a hierarchical tree structure. An entity can belong to more than one label in a KG.

1.1.2.3 Relations in Knowledge Graph

Depending on the nature of the objects in a triplet, the relations are classified into two main categories:

- Object Relation, in which an entity is linked to another entity. For instance, in the triplet $(Paris, capital\ of, France)$, the relation $capital\ of$ is an object relation connecting the head entity *Paris* to the tail entity *France*.
- Data Type Relation, where the head entity is linked to a literal. For example, in the triplet $(James, is\ born\ on, "Jan\ 1\ 1984")$, the relation *birthdate* is a data type relation.

1.1.2.4 Literals in Knowledge Graph

The literals in a KG encode that there is no additional information of the entities to be represented. The different types of literals present in a KG are:

- Text: Different information is stored in a KG in form of free natural language texts such as labels, entity descriptions, comments, titles, etc.
- Numeric: Date, area, size, and other data stored as integers, floats, and so on also provide important information about an entity in a KG.
- Image: also encode useful information about the entities. For instance, the gender of a person or the shape of an object can be determined by analyzing the respective images of the entities.
- Other Types of literals: External URIs containing an image, text, audio, videos, etc. linked to the entities also contain beneficial information.

Figure 1.2 illustrates an example of a KG extracted from Freebase consisting of entities, relations, labels, and literals. Here, *DaVinci*,*Monalisa*, *and James* are the entities. The entity *DaVinci* is of type *person*. The relations *is a*, and *is in* are object relations as they link two entities. The example also consists of triples with text and numeric literals. The relations in the attributive triples such as *Person*, *Museum* are the data type relations.

1.1.3 Applications of KG

Knowledge graph has vast amount of different and important applications. It actually enables fast retrieval of structured data about target entities upon user search. For example, when you search for a famous person, place, or a popular topic on Google, Google Knowledge Panel will pop up as shown in Fig.2 alongside with search results. This Knowledge Panel helps the users understanding the subject of interest quickly. The data source for Knowledge Panel is the Google Knowledge Graph launched in 2012. Knowledge Graph (KG) was first used in literature in 1972. Later, in 2010 Google Knowledge Graph was created from Freebase, which was a popular open source KG acquired by Google. The Google Knowledge Graph was later augmented by integrating many other data source such as the Wikidata.

Indeed, KG information integration is not limited to the construction of Google Knowledge Graph, but has many more applications in different domains. E-Commerce companies seek to build users and products KGs and merge with other companies' KGs in order to gain business intelligence and better sell their products to the right person. Hospitals and clinics share medical conditions about patients in the form of KG to facilitate better treatment in case the patients move to live at different places. Financial institutions also integrate knowledge bases to track down illegal activities such money laundering. Ride sharing companies such as Uber are also embarking on similar efforts to crack down collusion between ill-intentioned drivers. To identify the matching entity or entity alignment is therefore very important under the above application scenarios. Besides, knowledge graph is also and important source of information for AI-powered virtual assistant such as Siri, Alexa, and Google Assistant. Dialogs are first analyzed with Natural Language Understanding (NLU) algorithms to extract a few keywords as cues for locating the subgraph of the KG as the useful information. By traversing a few hops on the KG, sensible responses can be generated using Natural Language Generation models. Other applications of KGs including music recommendation systems based on music KGs or event forecasting system based on temporal KGs.

1.1.4 Challenges facing KG

KGs are mostly constructed from semi-structured knowledge, such as Wikipedia, or harvested from the web with a combination of statistical and linguistic methods. As a result, KGs try to make a good trade-off between completeness and correctness. In order to further increase the utility of such

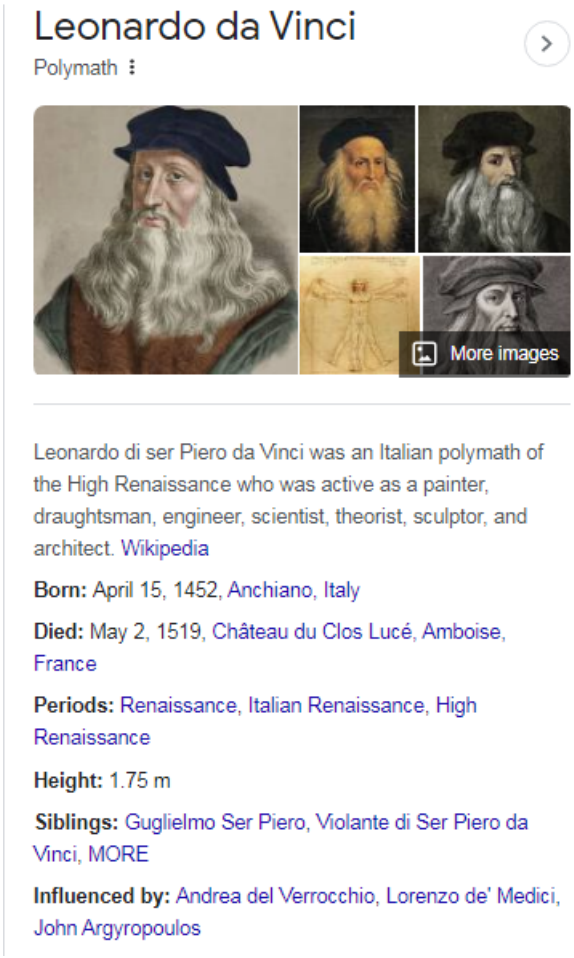


Figure 1.3 Google Knowledge Panel.

knowledge graphs, various refinement methods have been proposed, which try to infer and add missing knowledge to the graph, or identify erroneous pieces of information.

Researchers are working on solving the challenges facing KG in order to get the maximum benefit of KG. One Challenge is the correctness problem, which is formed due to adding incorrect data or depending on untrusted data source. The result of such a problem can destroy the trust of the end user with the returns of KG. to the Many challenges are facing KGs the correctness and the completeness. Another challenge which we address in this thesis is the completeness problem.

However, given the limited information accessible to each individual and the limitation of algorithms, it is nearly impossible for a Knowledge Graph to perfectly capture every single piece of facts about the world. There still missing links between the entities and missing links between the entities and their corresponding entity types. As such, Knowledge Graphs are often incomplete and many researchers have developed different algorithms to predict missing facts in knowledge graphs. As a result, for particular KG-based applications such as question-answering systems, incomplete KGs might not offer the right response to a correctly interpreted question. Given the KG in Figure 1.2, it would not be possible to answer the question, "Who painted Mona Lisa?", even though both the entity Mona Lisa and Da Vinci are included in the KG. Therefore, there arises a necessity to predict the missing relation connecting *DaVinci* to *MonaLisa* in the triplet $< DaVinci, ?, MonaLisa >$, or to identify if $< DaVinci, Painted, MonaLisa >$ is a correct triple for the KG. Additionally, it would be impossible to provide a response to the question, "Is France a place?". Here also, the entity *France* and the class *Place* are there in the KG, but the information that *France* is an instance of

the class *Place* is missing. Knowledge Graph Completion (KGC) aims to tackle the aforementioned challenges by addressing the issues of incompleteness and sparsity, hence improving the structure of the KG.

1.1.5 Existing Knowledge Graph

The KGs can be broadly categorized into public (open) KGs, and private (enterprise) KGs. Open KGs are accessible and available online for all people. Some of the most prominent and widely used public KGs are DBpedia [8], Freebase [9], YAGO [10], Wikidata[5], etc. Each public KG have been created in specific domains, such as media [11], life sciences [12], etc. Some of these KGs are extracted from Wikipedia, whereas the others are created by community. Unlike public KGs, private KGs are confidential to the respective companies such as Google [3], eBay [13], and Amazon [14], and they are used for commercial purposes.

1.1.5.1 Public Knowledge Graph

Public knowledge graphs (KGs) such as WordNet [15], Freebase [9], and DBpedia [8] are open, online available KGs that can be publicly viewed. In Table 1.1, we present some of the public KGs with their characteristics.

These KGs allow users to search (typically a keyword query), visualize and save them as well. Moreover, some of them provide SPARQL endpoints API to query and use their data when building applications. For instance, Wikidata allows individuals to query and visualize their KGs interactively.

Table 1.1 A number public knowledge graphs and their statistics .

–	Wikidata	DBpedia	YAGO	NELL
instances	17,581,152	5,109,890	5,130,031	1,974,297
axioms	1,633,309,138	397,831,457	1,435,808,056	3,402,971
Average outdegree	9.83	13.52	17.44	5.33
Average indegree	41.25	47.55	101.86	1.25
classes	30,765	754	576,331	290
relations	11,053	3,555	93,659	1,334
Release	Live	Biyearly	>1 year	1-2 days

1.1.5.1.1 DBpedia

DBpedia [8] knowledge graph mainly extracts its information from Wikipedia. It provides semantic interpretation of Wikipedia resources by enabling users to SPARQL query these resources. It is first open available dataset was released in 2007. Right now, it contains over 4.58 million entities. In Figure 1.4, we show the number of some entities available on DBpedia.

1.1.5.1.2 Wikidata

Wikidata [5] collects its information from various resources, mainly, Wikipedia, Wikivoyage, Wikisource, and other resources. This knowledge base can be edited by public. Wikidata is open and free, which it can be accessed by the public. It supports multilingual documents and information about entities in a structured and interlinked model. The number of entities for this knowledge graph is given in Figure 1.5.

1.1.5.1.3 YAGO

YAGO [6] extracts information automatically from Wikipedia mainly. In addition, it extracts information from other resources such as WordNet and GeoNames. It provides public access for the public using Turtle and SPARQL as well as CSV format. YAGO current version which is YAGO3 contains up to 10 million entities. The number of entities available in this knowledge graph is given in Figure 1.6.

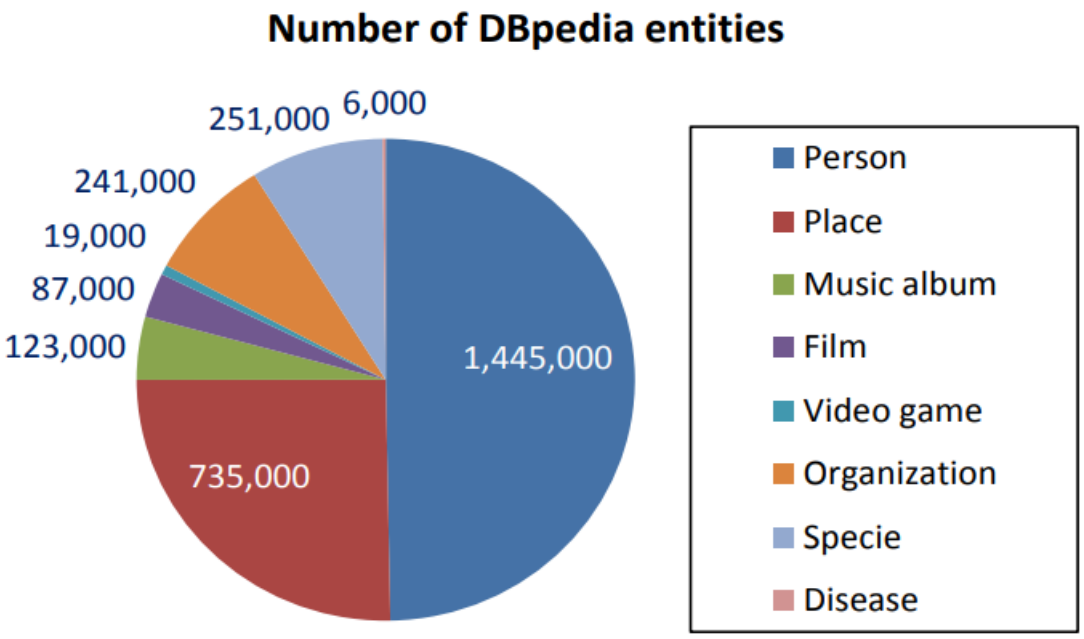


Figure 1.4 The total number of several DBpedia entities.

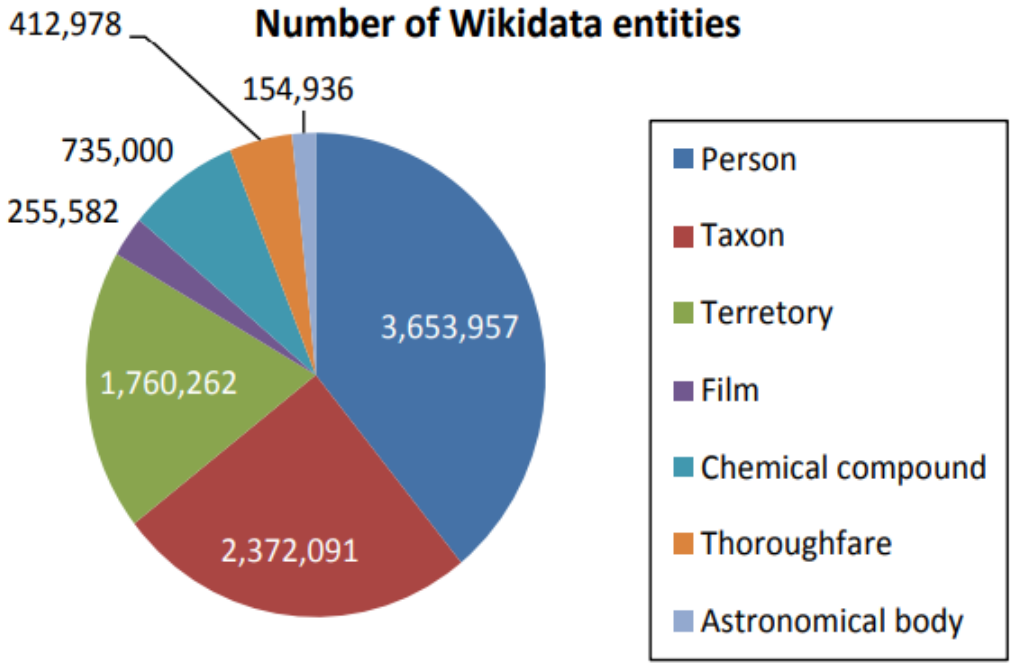


Figure 1.5 The total number of several WikiData entities.

1.1.5.2 Private Knowledge Graphs

Private knowledge graphs such as Google Knowledge Vault can be be accessed only for browsing and cannot be queried, i.e., end users can not extract data from them. Thus, they are neither convenient to create applications. Moreover, these graphs cannot be deeply analyzed. However, in this section, we did our best trying to describe and overview some of these KGs.

1.1.5.2.1 Google knowledge graph

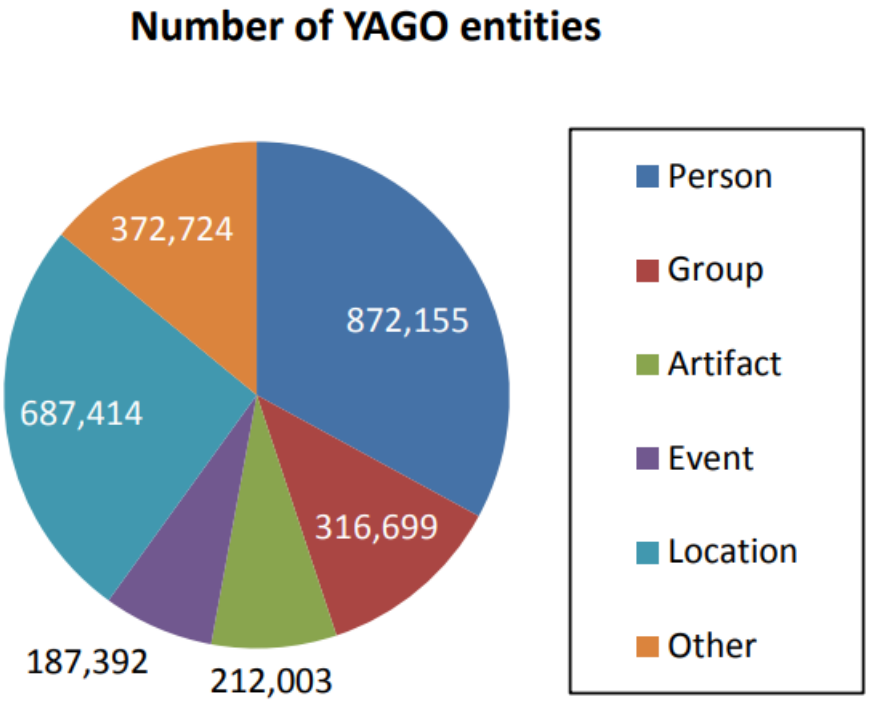


Figure 1.6 The total number of several YAGO entities.

Table 1.2 A number private knowledge graphs and their statistics .

–	Google knowledge vault	Google knowledge graph	Yahoo! knowledge graph
instances	45,000,000	570,000,000	3,443,743
relations	4,469	35,000	800
classes	1,100	1,500	250
facts	271,000,000	18,000,000,000	1,391,054,990

Google knowledge graph [16] is the largest private knowledge graph on the web with more than 18 billion entities. It is created to collect information about objects in the real world. It is mainly intended to assist humans when searching for such a query, by providing an integrated information which is clearly displayed at the top of each search result as a knowledge panel, Figure 2. For example, for a specific person, they collect and present his key information such as his date of birth, or how tall he is as well as social profile links. It also connects that person to closely related objects in the knowledge graph, such as a city where he was born, related persons, for instance, if we search for a specific artist, we get his related artists in this knowledge panel. Google knowledge graph is currently supported with 8 languages and English language.

1.1.5.2.2 Facebook entity graph

Facebook entity graph model the relationships and information about more than 1 billion Facebook users. The graph models the social connections between users, such as friendships. In addition, this entity graph contains also connections about check-ins, interests and work and living places. In figure 1.7, we illustrate an example of Facebook entity graph. The nodes represent the entities such as users, locations, workplaces. The edges represent the relations, such as friendship, lives in and works in.

1.1.5.2.3 Yahoo! knowledge graph

Similarly to Google, Yahoo! released in 2012 a so-called knowledge panel at the top of its search

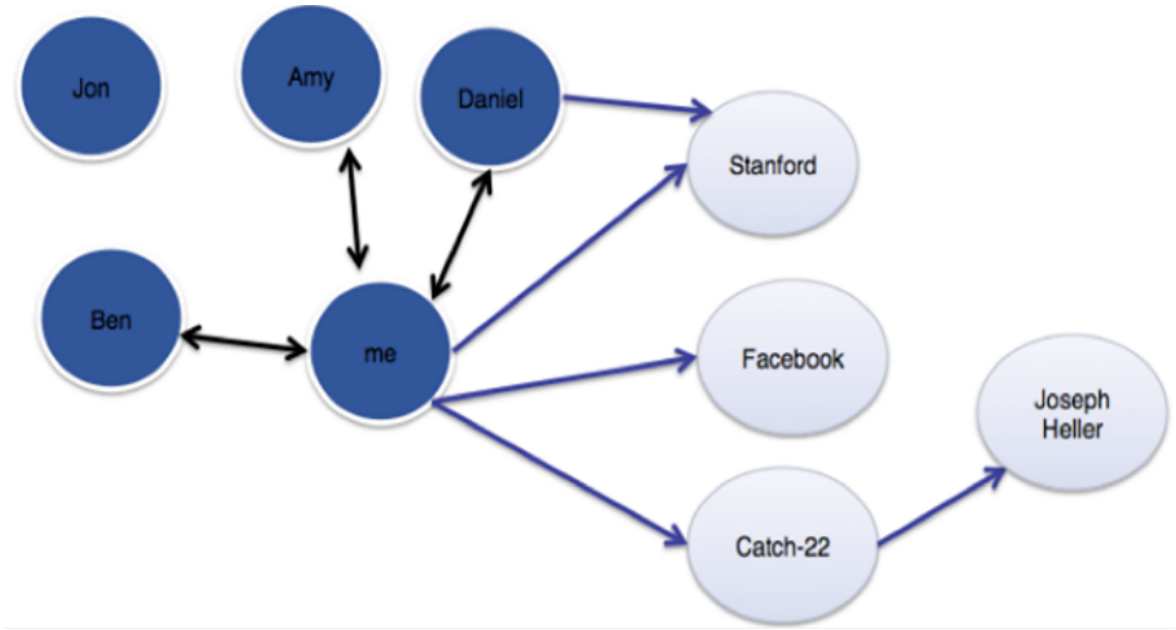


Figure 1.7 Facebook entity graph.

results. It contains an integrated and structured important information about entities. There are other knowledge graphs of the ones listed above such as Bing action graph and LinkedIn professional graph.

1.1.6 Knowledge Graphs Overlapping

Overlapping is the measurement of the similarities and differences between knowledge graphs. Mainly, it consists in measuring the number of instances in common between knowledge graphs. The traditional interlinks used to find similar instances is the *owl* : *sameAs* ontology property. However, this interlink is not enough, in [17] the authors propose a novel approach to find interlinks between two knowledge graphs using arbitrary rules, for example, linking the same cities of the same name. Typical similarity measures between names are, Levenshtein, Jaccard, Jaro and JaroWinkler.

The measures used to find the overlap are as follows: (1) global measures, such as overall size and shape of knowledge (2) class measures, finding the similarity between details of the classes. The resulting overlap measures finds for instance the number of instances in common, the average indegree in common, or the average outdegree in common of such selected classes and knowledge graphs. For instance, find the number of common instances between Wikidata and NELL related to "Social Event" class.

Based on the outcomes of [18], the most riched resource is Wikidata, it contains twice of instances of DBpedia and YAGO. However, DBpedia contains the largest number of places compared to other knowledge graphs. In figure 1.9, we present the results of [18] that shows the range of overlapping between 4 different knowledge graphs

1.2 General approaches for Knowledge Graph Completion

Approaches for Knowledge Graph Completion KGC usually target to increase the coverage of a knowledge graph. They predict missing knowledge such as entities, type and relations between entities as well. These approaches are composed of internal and external methods. Internal methods use the existing knowledge in the knowledge graph itself to predict missing knowledge. On the

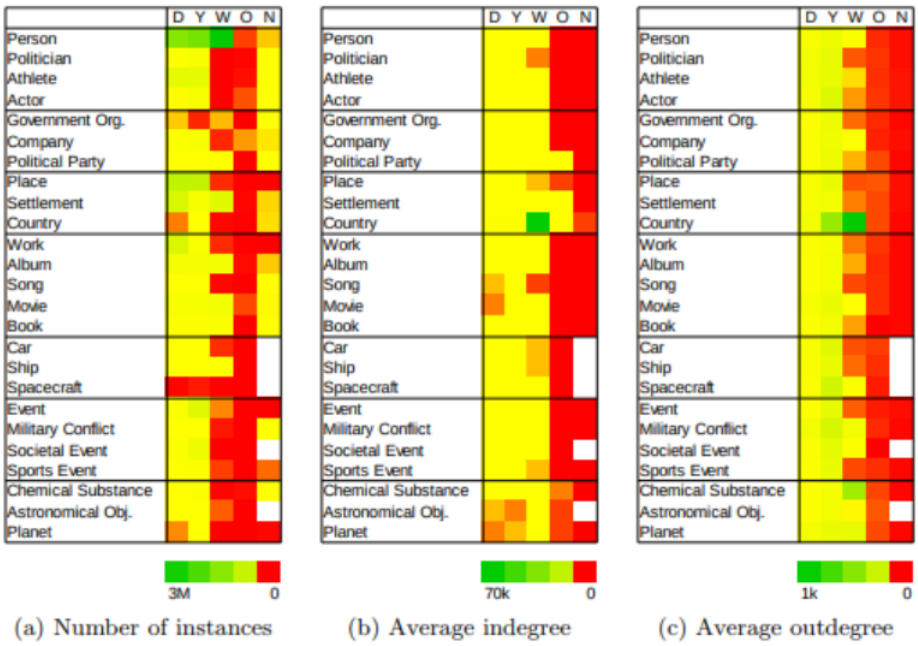


Figure 1.8 Number of instances (a), avg. indegree (b) and avg. outdegree (c) of selected classes. D=DBpedia, Y=YAGO, W=Wikidata, O=OpenCyc, N=NELL..

other hand, external methods use knowledge from other knowledge graphs or large text-corpora. The authors in [19] employ web search engines to complete missing relations in knowledge graphs. They formulate first a structured query and pass it to a search engine. Finally, they use information retrieval and extraction techniques to complete knowledge graphs. The authors in [20] propose the use of reinforcement learning to fill missing values in a knowledge base efficiently. They mainly address three basic attributes, email, job title and affiliation of professors inside an institution. Similarly to [19], they also use the web search engines and information extraction techniques to extract information from the web. Works that use social media to construct knowledge graphs such as [21] work on crawling, parsing, annotating and analyzing social media content to create new knowledge graphs from this content. Works that creates scholarly knowledge graph such as [22] work on integrating data from heterogeneous resources and metadata from DBLP and Microsoft Academic graph to create new scholarly knowledge graphs. Similarly, [23] consolidates data from different resources to build new knowledge graph about artists using LSH techniques. NOUS [24] is an approach for constructing domain knowledge graphs through integrating existing knowledge graphs and information from external resources. In [25], the authors suggest social networks for increasing the coverage of knowledge graph. Specifically, they found what people say on social networks about real-world entities in a knowledge graph. They use three resources mainly: Google+, Facebook, and Twitter. In the recent year also there are many papers published in this field. The authors in [26] [27] [28]address the problem of validating and filling entities and relationships inside knowledge graphs.

Knowledge Graph Embedding models were first proposed as link prediction technique to mainly solve the Knowledge Graph Completion problem. The main goal of Knowledge Graph Embedding(KGE) models is to generate a latent representation of the entities and relations in a KG to a continuous low-dimensional vector space that can be used for different knowledge acquisition tasks and downstream applications. A typical KGE model is characterized by the following steps:

- Representation Space, the low-dimensional space in which the relations and entities are represented. Entities are represented as vectors or modelled as multivariate Gaussian distributions.

Relations, on the other hand, can be represented as vectors, matrices, tensors, multivariate Gaussian distributions, or even mixtures of Gaussians.

- Scoring Function given by $fr(h,t)$ is defined on each triplet < h, r,t > to measure its plausibility. The triples observed (or true triples) in the KG tend to have a higher score and lower scores are assigned to false/negative/corrupted triples.
- Encoding Models for representing and learning relational interactions between the entities. The model learns the representations of the entities and relations by solving an optimization problem that maximizes the total plausibility of observed triples. Negative or corrupted triples are generated in this step and the method used to generate negative samples has an impact on learning the embeddings.
- Auxiliary Information: Any additional information available in the KG, such as literals, that can be leveraged to enrich the embeddings of the entities and the relations. In such a scenario, an ad-hoc scoring function is defined for the additional information and is integrated into the general scoring function.

All the aforementioned steps are depicted in Figure 3. An extensive study of the existing KGE models is provided in Chapter 2. Furthermore, Chapters 3 and 4 discusses the shortcomings in the baseline models and proposes a new KGE model for link prediction in KGs.

1.2.1 Link Prediction

As mentioned earlier, KGs store information in form of triples, hence the KGC problem can be looked upon as a problem of estimating missing parts of the triples. Therefore, KGC is achieved by link prediction that aims to estimate the probability of the existence of links between entities based on the current observed information in the KG [29]. Link prediction can be further categorized into three different types of prediction problems depending on the nature of the missing links. The different types are as follows:

- Head and Tail Prediction: The head or tail entity in a triple $< h, r, ? >$ is predicted by defining a scoring function. For e.g., in reference to the KG shown in Figure 1.2, the prediction of the missing entity in the triple (Da Vinci , Painted, ?) is denoted as the tail prediction as the head entity and relation information are provided. On the other hand, (?, is in, Louvre) is considered as the head prediction since the relation and tail entity are given.
- Triple Classification: A binary classifier is trained to identify whether a given triple is $false(0)$ or $true(1)$. With reference to the triple in the illustration Figure 1.2 as an example, triple classification helps in identifying if $(Paris, capitalof, France)$ is a true triple for the KG.
- Entity Type Prediction: It deals with predicting the special kind of links i.e., the semantic types of the entities in the KGs. The problem is transformed into a classification problem in order to predict the semantic type of each entity in the KG and is given by < e, is A , ? >, where e is the entity.

1.3 Conclusion

In this chapter, we provided an overview of knowledge graphs, we started by presenting facts about the existing private and public knowledge graphs. We then presented a statistical comparison between these knowledge graphs. We, in addition, presented a review of the recent state-of-the-art approaches on knowledge graph completion.

In this thesis, Chapter 3, Chapter 4, we propose a knowledge graph completion approach relying on modular arithmetic.

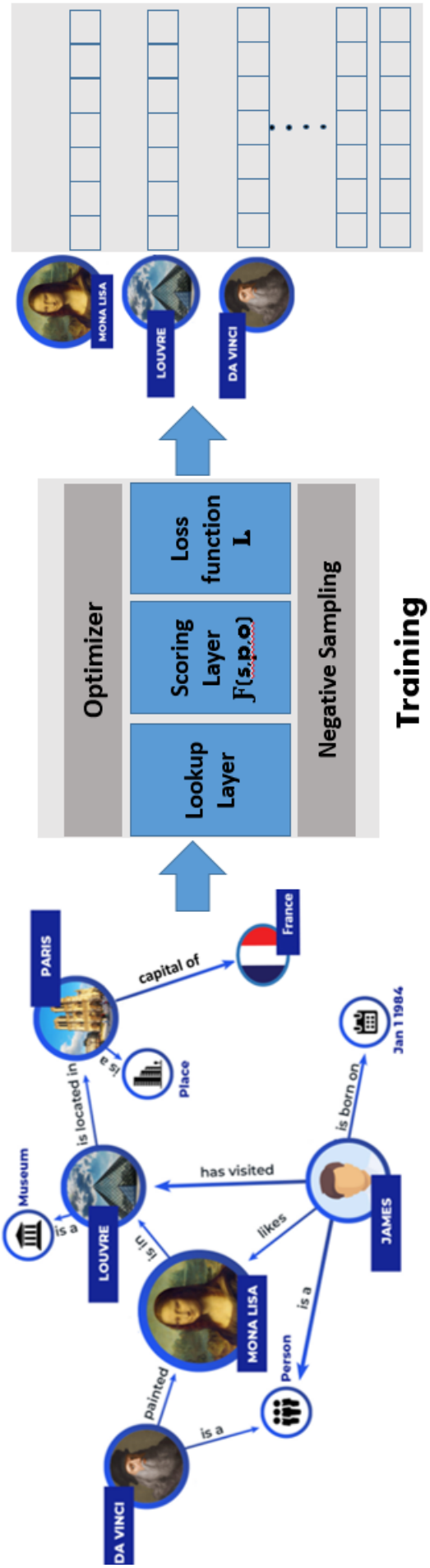


Figure 1.9 KGE framework..

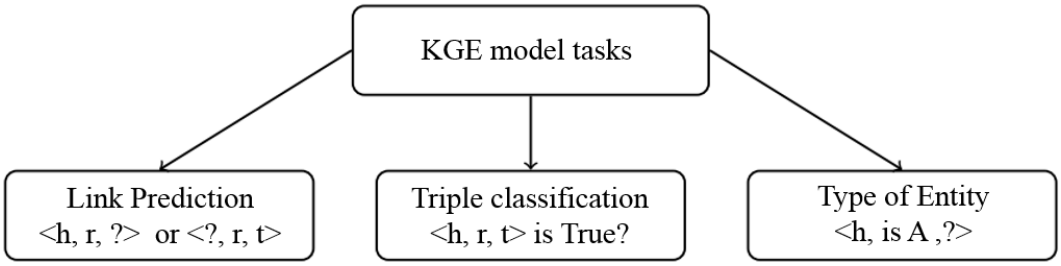


Figure 1.10 tasks of KGE.

2 Literature Review

Since a KG often consists of many complex structures (namely N to 1, 1 to N, N to N), researchers have proposed embedding KGs into different representation spaces to better preserve these information complex structural structures [30, 31, 32, 33, 34]. Each representation space has a different structure and set of characteristics. Nevertheless, besides the basic mathematical spaces, there exist other spaces that offer superior features for KGE. In hyperbolic space, for instance, the region and length grow exponentially with the radius, increasing the amount of space that is accessible for the embedding assignment [35, 30, 36, 37, 38, 39]. Additionally, regularization of embedding vectors is no longer necessary for effective learning since in Lie groups, embedding vectors never diverge infinitely [31]. Consequently, KGE techniques based on distinct representation spaces can capture and maintain distinct structural and attribution details in the original KGs. Nonetheless, neither a comprehensive analysis of KG embedding techniques from the standpoint of representation spaces nor any literature demonstrating how to appropriately select representation space given specific KGE tasks exist.By summarizing KGE approaches based on their mathematical representation spaces'structures and features, we hope to close this gap in this study.

Note that we introduced some algebraic definitions of some basic spaces in this chapter. Some geometric perspectives, including euclidean geometry and hyperbolic geometry, can also be introduced in accordance with them. Simultaneously, we observe that KGE contains a variety of mathematical spaces that each have a unique function and correspond to distinct mathematical structures. The many spaces in KGE frequently have complex connections. For example, because there exist overlapping structures between manifolds and Euclidean geometry, they have inclusion relations. Furthermore, some spaces—like probability space and spherical space—are not identical since they come from separate mathematical structures and cannot be treated in the same category.

Accordingly, in order to classify the special spaces and categorize KGE models more accurately based on three mathematical structures, namely algebraic structure, geometric structure, and analytical structure as shown in (See figure 2.1).We build a systematic, comprehensive, multi-angle categorisation.

The definitions and characteristics of the aforementioned three mathematical structures will be covered in this section, then provide some KGE methods that learn embedding in the respective mathematical structure, as well as summarize how spatial advantages function in KGE models.

2.1 Algebraic Structure

An algebraic structure is a nonempty set on which one or more finite operations meet the stated axiom[40, 41].Vector space, group, and ring are examples of representational algebraic structures.For example,the vector space X is an algebraic structure that includes numerous basic binary operations like addition, subtraction, and multiplication. The majority of techniques in machine learning and

even knowledge graph embedding entail algebraic operations like group embedding [31, 42], user and word embedding [36, 37, 39], mobuis embedding [35, 43, 44] and so on are all techniques in the category of algebraic structure.

2.1.1 Vector Space

The Vector Space is the most extensively used mathematical space in machine learning. Many KGE techniques make significant utilization of vector space because it defines two handy algebraic operations: vector addition and scalar multiplication. The aforementioned methods use these operations to project both entities and relations into the same vector space, with the goal of maintaining the vector representation space's relational relationships between entities. Based on the scalar field, the vector space may be divided into two types: real vector space and complex vector space. We divide vector space-based KGE algorithms into three separate classes based on these categorization criteria: complex vector space-based, real vector space-based, and additional models inside the vector space domain, such as neural network-based and external information-based models. In the following sections, we will go over each of these sets of KGE approaches in detail.

2.1.1.1 Real Vector Space

TransE [45] is a representative KGE algorithm based on real vector space. It projects all entities and relationships in a low-dimensional real vector space R given a knowledge graph. It reflects the head entity, the relation, and the tail entity in a triple as embedding vectors h, r, t. If the triplet (h,r,t) holds, the embedding of the tail entity t should be as near to the head entity embedding h as possible, which adheres to the principal: $h+r=t$. The addition operation is a very basic and effective way of directly expressing the relationships between objects. $f_r(h,t) = |h+r-t|$. Then, the score function of TransE is:

$$s(h,r,t) = -||h+r-t||_{1/2}$$

Although TransE shows a high efficiency when applied in large-scale knowledge graph embedding, it still struggle when dealing with complex relations such as $1-N, N-1$, and $N-N$. In order to overcome such challenge, the authors of transH[46] proposed an extension of TransE called TransH. Indeed, TransH assigns a hyper-plane to each relation, so that the heads and the tails of this relation are projected to. This model enables each entity to have different embedding representations depending on the relation involved in. Specifically, TransH assumes that each relation embedding r lies in a different relation-specific hyperplane w_r . In order to measure the plausibility that a triple (h,r,t) holds, the head entity embedding h and tail entity embedding t are first projected into the relation-specific hyperplane w_r:

$$h_\perp = h - w_r^\perp h w_r, t_\perp = t - w_r^T t w_r$$

where $h_\perp$ and $t_\perp$ represent the projected head and tail entity embeddings, respectively. Therefore, the score function of a triple

$$(h,r,t)$$

is defined as:

$$s(h,r,t) = -||h_\perp + r - t_\perp||_2^2$$

TransH assures that the embedding of t is distinct regardless of the head entity or the relation is the same by projecting the entity embedding to a relation-specific hyperplane, allowing entities to have various embeddings in different relationships. TransH makes it clear that projection is important. Projection is a widely used vector space operation that creates a variety of connections (See figure 2.2.a)

TransH considers that the relation embeddings and entity embeddings belong to the same embedding space, which will restrict variety even if it can manage multi-relations in KGs successfully. On the other hand, an entity can have more than one aspect, and various interactions might concentrate on distinct facets of entities. In order to solve this issue, TransR [47] suggests that entities should be embedded in an entity space $R^k(h,t \in R^k)$ and relations should be embedded in a distinct relation space $R^d(r \in R^d)$, where R^k and R^d are not always the same. The relation-specific mapping matrix $(M_r \in M_{d\times k})$ correlates the entity and relation embeddings by projecting the entity embedding from entity space to relation space. As a result, TransR defines the scoring function as:

$$s(h,r,t) = -||M_r h + r - M_r t||_2^2$$

RESCAL [48] uses bilinear operations with a scoring function to describe the semantic relationships between entities.

$$s(h,r,t) = h^T M_r t = \sum_i \sum_j [M_r]_{ij}.h_i.t_j$$

where each relation is defined as a matrix $M_r M^{d\times d}$, and $[M_r]_{ij}$ denotes the $i-th$ row and $j-th$ column of the matrix M_r. Equivalent linear models have shown excellent performance on downstream tasks, including TuckER [49], LowFER [50], ANALOGY [51], HolE [52], DisMult [53], and SimpleE [54]. Thus, the plausibility of facts may be evaluated by matching the latent semantics of entities and relations using straightforward and effective linear methods (See figure 2.2.b).

Some KGE models have recently rises learn embeddings in the actual vector space. For instance, ReflectE [55] maps attributes and entities according to the Householder matrix by using reflection transformation. In order to capture patterns of relationship, LineaRE [56] perceives a relation as a linear function of entities. Moreover, to get an increased diversity distribution, TimE [57] projects entities into the nonlinear time domain. Numerous real vector space mathematical procedures were discovered to be worthwhile to investigate and apply to KGE.

2.1.1.2 Complex Vector Space.

Complex vector embedding is more capable of handling a wide range of simple relations [58] than real vector embedding, including symmetric and antisymmetric relationships. Many studies have also been conducted on complex embedding techniques for KGs, which embed entities and relations in the Complex Vector Space (that is, h, r, $t \in \mathrm{C}_k$). The first model to employ complex embedding in KGE was ComplEx [58]. It builds a scoring function in complex vector space using the Hermitian dot product specifically:

$$s(h,r,t) = Re(h^T diag(r)\bar{t})$$

where $\bar{t}$ is the transpose of t, $Re(.)$ denotes the operation to obtain the real part of a complex number. Given that the scoring function is not anymore symmetric, the ordering of entities might affect the scores assigned to facts with antisymmetric relations. ComplEx may therefore preserve the dot product's efficiency advantages while successfully capturing antisymmetric relations.

RotatE [59] projects entities and relations into the complex vector space and refers to each relation as a rotation from the source entity to the target entity with the principal $t = h \circ r$ (where $\circ$ denotes the Hadamard product, i.e., elementwise product). This is driven by the Euler's rule $:e^{i\theta} = cos(\theta) + sin(\theta)$. Through the angular transformation as shown in (See figure 2.2.c), several entities may be directly represented in order to capture certain patterns like symmetry, antisymmetry, inversion, and composition. BiQUE [32] enhances the quaternion system to a more powerful algebraic system called biquaternion $q = (w_r + w_i I) + (y_r + y_i I) + (z_r + z_i I), where\ w_r,w_i,y_r,y_i,z_r,z_i \in R$. Using the biquaternion Hamilton product, BiQUE lends itself to a robust geometric interpretation (the Euclidean/hyperbolic rotation). Furthermore, in four-dimensional space, quaternions have an

increased degree of freedom. Notably, it is very simple to interpolate between two quaternions, which facilitates the establishment of rotations. Recently, DualE [34] has used a unique framework that can handle translation as well as rotation operations in the dual quaternion space (where a and b are dual units and $Q_{dual} = a + \varepsilon b$). DualE and QuatE are combined in DualQuatE [60]. The framework of RotatE is carried over into HA-RotatE [61] and CORE [62], which enhance KGs embedding capabilities.

2.1.1.3 Vector Space: Models using Neural Network

KGE models use neural networks [63] in order to learn embedded characteristics. Nevertheless, we are unable to comprehend the precise mathematical characteristics that these neural network-based KGE models represent due to the black-box nature of neural networks. However, as these models produce vector representations, we classify them in conventional vector spaces.

ConvE [63] first reshapes the head entity and relation into a 2D matrix. To describe interactions between entities and relations, it then uses several layers and 2D convolution. The definition of the scoring function is:

$$s(h,r,t) = \alpha(vec(\alpha([h_{2D}, r_{2d}] * \beta))W)t$$

where α denotes activation function, and $vec()$ means the vectorisation operation. h_{2D} and r_{2d} are $2D$ reshaping of h and r (i.e., $h_{2D}, r_{2d} \in R^{k_w \times k_h}$, and the convolutional filter symbol is β. ConvE displays high expressiveness and provides remarkable performance thanks to the nonlinear neural network layer's strong feature extraction capabilities. R-GCN [64] uses Graph Convolutional Networks (GCNs [65, 66]) to handle large-scale relational data by focusing on small graph neighborhoods. This allows the representation of links between entities and relations. The propagation procedure is applied on the hidden state y of layer $l+1$ in order to compute the forward-pass update for an entity:

$$Y_i^{l+1} = \sigma(\sum_{r \in R} \sum_{j \in N_i^r} 1/c_{i,r} W_r^l y_j^l + W_0^l y_i^l)$$

where N_i^r denotes the set of neighbor indices of node i under relation $r \in R$ and $c_{i,r}$ is a normalisation constant. CompGCN [67] simultaneously embeds nodes and relations in a graph by utilizing a range of composition operators that are developed from KGE techniques. It has been shown that this method can generalize a number of multi-relational GCN approaches that are already in use, such as R-GCN [64], Directed-GCN [68], and Weighted-GCN [69]. KG-BERT [70] is a transformer-based [71] model that uses a pre-trained language model BERT [72] to perform knowledge embedding and interpret triples as text sequences. KG-BERT may make use of a wealth of linguistic data found in the text and highlight the most relevant terms related to a trip.

In order to attain state-of-the-art results, Knowformer [73] recently used position-aware relational compositions to encode the semantics of entities occurring in different places inside a relational triple. These compositions have been demonstrated to support the self-attention mechanism [71] in identifying distinct entity responsibilities according to their location.

2.1.1.4 KGE Integrating Auxiliary Information

Within KGE techniques, learning knowledge graph embeddings using auxiliary data is an important topic. To maintain a thorough examination, even if this issue might not be as directly related to the mathematical representation space, we will give a succinct summary of several traditional external-information-based KGE models. Text descriptions, entity types, relational structures or pathways, and other information are often utilized in the work that has already been done [74, 75, 76, 77]. By using CNNs to extract the semantics of entity descriptions in a representation learning fashion, DKRL [78] explores deeper knowledge representation. TKRL [79] uses the type information as relation-specific type constraints and views hierarchical entity types as projection matrices. It has

been shown that hierarchical type information, which is essential for creating knowledge graph representations, is successfully captured by TKRL. Knowledge graph relations are treated by HRS [80] as a three-layer hierarchical relation structure that is easily incorporated into other KGE models to provide a wealth of structural information.

By using relational routes, RSN [81] and Interstellar [82] address the problem of earlier model-sínability to represent long-term relational dependencies of entities effectively. TransO [83] has been proposed recently as a way to improve decision-making in complex situations by seamlessly integrating all available ontology information (i.e., type information, relation constraint information, and hierarchical structure information) within the knowledge embedding process. Furthermore, in the field of knowledge graph embedding, conceptual information [84, 85, 86] is also very important when paired with additional external information like images [87].

2.1.2 Ring

Rings are algebraic structures that generalize fields in mathematics [88, 89, 90]. One of the primary areas of ring theory is commutative rings. A set of integers with addition and multiplication as well as a set of polynomials with the same operations are two examples. Later on, ring theory was found to be beneficial in geometry and analysis [91]. We will define ring first in this section and then talk about KGE models that use ring as an embedding space.

Definition 2.1.2.1: A ring is defined as a set S that has two binary operations (addition: + and multiplication: ·) that meet the following:

- Under addition, S is an Abelian group, which means that:
  - $(a+b)+c = a+(b+c)$ *for every* $a,b,c \in R$;
  - $a+b = b+a$ *for every* $a,b \in S$;
  - There is an element 0 in S such that $x+0 = x$ for all $x \in S$;
  - For each x in S there exists $-x$ such that $x+(-x) = 0$.
- Under multiplication, S is monoid means that:
  - $(a.b).c = a.(b.c)$ *for every* $a,b,c \in S$;
  - There is an element 1 in S such that $x.1 = x$ for all $x \in S$;
- With regard to addition, multiplication is distributive. That means:
  - $(a+b).c = (a.c)+(b.c)$ *for every* $a,b,c \in S$;
  - $a.(b+c) = (a.b)+(a.c)$ *for every* $a,b,c \in S$;

In manifold-based embedding, where entities and relations are embedded to the surface of Mobius ring, MobiusE [43] extends KGE. Due to the additional features on ring, MobiusE has considerably more expressiveness and versatility than TorusE [31] using the scoring function (dist(h + r,t), where + and dist denote addition and distance function particularly constructed on Mobius ring, respectively. Since MobiusE subsumes TorusE, it automatically acquires all of TorusE's relation pattern, such as composition, inversion, and symmetry/antisymmetry. It should be noted that the Mobius band is a non-oriented surface, making it difficult to define the concepts of clockwise and counterclockwise. This information may be useful for problems involving orientation.

2.1.3 Group

Groups, which are algebraic structures made up of a set and an operation, are common in both mathematical and non-mathematical domains [88, 89, 90]. For instance, lie groups are employed in particle physics [92] and symmetry groups are utilized to characterize the symmetries of geometry [93]. Groups are also extensively employed in machine learning due to their special abstract algebraic features. This section will begin with defining groups and go on to discuss KGE models that use groups as embedding spaces.

Definition 2.1.3.1: A function is a (binary) operation on a set G when:

$$* : G \times G \rightarrow G$$

Definition 2.1.3.2: A group is a set G that has an operation $*$ and member $e \in G$ that is known as the identity, such that:

- $(a * b) * c = a * (b * c)$ *where* $a,b,c \in G$;
- $e * a = a$ *for every* $a \in G$

In order to address TransE's regularization issue, TorusE [31] suggests integrating KGEs into a unique algebraic structure called Torus. An Abelian Lie group Torus is created by using the nature projection to extract it from vector space as follows: $R^n \rightarrow T^n, x \mapsto [x]$. Torus ensures that the model never diverges indefinitely.

The first effort to use a finite non-Abelian group in KG embedding to account for relation compositions is DihEdral [94]. Dihedral groups are able to capture all mapping properties including symmetry, inversion, and (non-)Abelian composition. That is because the components in a dihedral group are easily built by rotation and reflection operations, and the multiplication between elements can be Abelian or nonAbelian.

DensE [42] breaks down a relation operator into a scaling transformation and a rotation based on the SO(3) group (SO(3): Special Orthogonal Group in 3 dimensions). For the first time, NagE [95] establishes the importance of the group algebraic structure in relational embedding model creation. In particular, group definition may naturally meet knowledge graph fundamental features (such as composition and inversion), suggesting that group-based models offer a lot of promise for handling KGE problems. In order to reach state-of-the-art performance, more recent models based on group structure, such Module E [96], take into account both entity and relation as group components.

2.2 Geometric Structure

Any two points on a manifold that have isometric neighbourhoods are said to be in a geometric structure [97], which is a locally homogeneous full Riemannian metric. Even if there are few in-depth explanations of geometric structures, in this case we concentrate more on the knowledge graph embedding models constructed on several geometric models/spaces and thoroughly examine them from the perspective of Euclidean geometry, hyperbolic geometry, and spherical geometry.

2.2.1 Euclidean Geometry

The study of geometrical forms and figures using various axioms and theorems is known as Euclidean geometry. In essence, it is introduced for plane or flat surfaces. The Greek mathematician Euclid used this aspect of geometry, and he also wrote about it in his book Elements [98]. The Greek terms "geo," which means "earth," and "metrein," which means "to measure," are the origin of the word geometry.

Euclidean geometry is concerned with various forms and quantities such as squares, triangles, angles, points, and lines. This section will split the KGE models produced in Euclidean geometry into four parts based on several coordinate systems: Cartesian coordinate, Polar coordinate, and Spherical coordinate. It is important to note that there are certain concepts shared by vector space under algebraic structure and Euclidean geometry under geometric structure. This is due to the general strong relationship between geometric and algebraic aspects. However, we shall highlight the benefits of KGE models from a geometric standpoint in this section.

2.2.1.1 Cartesian coordinates

The standard Cartesian coordinates form the foundation for the majority of translation-based models. TransE, for instance, adheres to the following principle: $h + r = t$, meaning that a translation makes the head entity's vector equal to the tail vector. From head to tail, these three vectors are joined to produce a closed path in Cartesian coordinates. It is practical to specify each relation as a rotation from the source entity to the destination entity, as demonstrated by RotatE. For example, two relations $r1$ and $r2$ are inverse if and only if their embeddings are conjugate, which means they are symmetric about the real axis. Geometric transformations in Cartesian coordinates allow for the clear illustration of all related patterns. Furthermore, QuatE [34] expands complex space to quaternions, which have two rotational planes. Tang et al. [99], who were inspired by RotatE, use orthogonal transformations to expand RotatE from the 2D complex domain to high dimensional space while maintaining the capability of improving performance when modeling various patterns. Moreover, there are a number of RotatE-based models that build upon the key concepts of the first RotatE framework, including [100, 101, 102, 103, 61]. In higher dimensional space, for instance, rotational relations are defined by Rotate3D [100] and Rotate4D [103]. Complex relations are still difficult for KGE models to comprehend, though.

PairRE [104] makes use of paired vectors for each relation $[r_h, r_t]$ in order to lessen this issue, where r_h and r_t might have different values to suit the complicated relations. Relation vectors may project entities to any location inside a unit circle on the Cartesian coordinates by using the scoring function. Further research shows that PairRE is also capable of capturing subrelations beyond those that RotatE can represent [59].

There are also other recent developments in Cartesian coordinate-based KGE models. TripleRE [105], for instance, develops the translation component independently while inheriting the projection part from PairRE. Information from tail entity representation is combined with head entity representation in InterHT [106]. HousE [107] uses householder parameterization [108] to capture important relation patterns, whereas TranS [109] is suggested as an effective way to capture single relations. CompoundE [110] uses three basic Euclidean geometric procedures to embed KGs effectively. In order to better fit the complex underlying properties of a KG, CompoundE3D [111] updates the compound transformation. For the purpose of solving the circular relation difficulties, ConE [112] combines an explicit relation with a latent relation as a collaborative relation. To sum up, translation, rotation, reflection, and scaling are the most common geometric transformations used in KGE. It has been demonstrated that these transformations are successful in identifying key relationship patterns and mapping attributes.

2.2.1.2 Spherical Coordinate

For temporal knowledge graphs, STKE [113] depicts entities and relations in a spherical coordinate system. The polar portion φ, the azimuth part θ, and the radial part r are present in every entity. Spherical coordinates allow STKE to consider temporal changes as rotation and scaling of entity embeddings, allowing it to dynamically differentiate between various time-constrained entities.

2.2.2 Hyperbolic Geometry

In order to incorporate hierarchical multi-relational data in the Pointcare ball of hyperbolic space MuRP [35] is presented. MuRP surpasses Euclidean KGE models and delivers higher performance, especially in hierarchical datasets, since hyperbolic surfaces, as compared to Euclidean space, may be found to have more spaces to represent entities and capture hierarchy information with increasing radius [114].

With ATTH [30], logical patterns and hierarchies are simultaneously captured, as opposed to learning in a hyperbolic environment with a fixed curvature as in MuRP. To prevent accuracy mistakes, it learns a unique absolute curvature for each relation (such as rotation and reflection). Additionally, models under various situations (e.g., ATTE, ROTE/H, and REFE/H) are proposed based on ATTH. As was also indicated in the section on polar coordinates, HBE [115] uses an extended Pointcare ball to overcome boundary limitations that may arise in a standard Pointcare ball when capturing hierarchical structures using a polar coordinate system (Pointcare disk). Sun et al. [116] similarly use HyperKA to describe KG embedding in a hyperbolic space, but they start by using a graph neural network (GNN) to embed hyperbolic translation. However, Kai Wang et al. [117] had to consider the question of whether hyperbolic geometry was actually required given the ongoing growth of hyperbolic models. Given that the hyperbolic-based model invariably demands more computing complexity, which in turn implies higher training resource requirements, it begs the issue of whether the advantages outweigh the drawbacks. In order to address this problem, the hyperbolic operation in RotH is made simpler with the suggested RotL and Rot2L. RotL can save more than half of the training time and lessen the computing complexity of RotH [30] by developing Flexible Addition. It is important to remember that knowledge graphs contain a variety of mixed relations and that other types of information are frequently overlooked in favor of hierarchical information. HypHKGE [118] introduces hyperbolic hierarchical transformations for the purpose of extracting hierarchies. MuRMP [119] use the mix-curvature model in conjunction with GNN to more effectively reflect the KGsínherent heterogeneous nature. An ultrahyperbolic manifold is taken into consideration in UltraE [120] in order to solve the non-hierarchical embedding issues. Notable performance has also been shown by a number of modern KGE models based on hyperbolic spaces, such as SEPA [121] and FFTAttH [122].

2.2.3 Spherical Geometry

We first outline an obvious distinction between the ideas of spherical coordinate system and spherical geometry before providing the spherical geometry models to prevent misunderstandings. The previously described spherical coordinate system may be thought of as a tool to intuitively express the locations of points. The fundamental ideas underlying it are derived from Euclidean geometry and consist of points and straight lines. However, the fundamental ideas of spherical geometry [123] are that there are no parallel lines and that any two lines intersect at two points, which is known as a great circle. Since the circular pattern of the vector field produced by spherical embedding has a natural circularity, spheres and ring structures are compatible [124]. As such, spherical-based models are also capable of producing competitive results on intricate relational datasets.

TransC [125] encodes every concept in the knowledge graph as a sphere and each instance as a vector in the same semantic space in order to distinguish between the concepts and the instances. For instance, TransC constructs many alternative locations (e.g., inclusion, intersection, separation) between two idea spheres under various conditions in order to determine the links between concepts and sub-concepts (i.e., subClassOf). In contrast to TransC, HypersphereE [126] expands the sphere into the hypersphere in order to account for instance uncertainty. ManifoldE [33] is another unique model that extends point-wised embedding to manifold-based embedding. ManifoldE utilizes the manifold-based principle $M(h,r,t) = D_r^2$ (where M is the manifold function) in the sphere and hyperplane, respectively, in place of the earlier translational-based concept $h + r = t$. In the Sphere

condition, an entity that was previously only thought of as a point in traditional models is now expanded to a full high-dimensional sphere. ManifoldE minimizes noise through this mean, allowing it to better discern actual facts. Numerous mappings between the surface and the circle's center are possible due to the sphere's smooth surface, which greatly increases the embedding's flexibility. As is well known, KGs frequently have rich structural types including cyclic and hierarchical types. When KGs are embedded in a single curvature space, either hyperbolic or Euclidean space, fails to appropriately represent the KGsíntrinsically diverse structures and as a result. SEA [121] captures a range of logical and structural patterns by combining different existing representations of KGE inquiries using spherical geometry. Interestingly, a more complex structural representation may be achieved by projecting SEPA [121], an alternate variant of the SEA, onto the Pointcare ball.

2.3 Analytical Structure

In general, an analytical structure is one that has a measure. For example, in Euclidean space, the metric (or distance) is clearly defined, allowing us to perform various analytical operations such as differentiation and integration. Analogously, the probability space can be seen as an analytical structure as the probabilistic measure is defined there. We will examine the KGE models in detail in this part by breaking them down into two primary spaces: Euclidean Space and Probability Space.

2.3.1 Probability Space

As far as we are aware, KG2E [127] is the first density-based KGE model. It depicts each entity/relation as a multi-dimensional Gaussian distribution $N(t, \varepsilon)$ in probability space, where the covariance metric ε indicates the related (un)certainty that affects other entities. The mean vector t indicates the entity's position. Furthermore, two comparable approaches based on anticipated likelihood and KL-divergence are put forth to examine the distinction between symmetric and asymmetric data, respectively. A formal consideration of various relation semantics in KGs has been made in previous models. But the conventional translational-based models never provide more than one vector to a single vector, neglecting the possibility that a connection may have more than one meaning. In order to manage numerous relation semantics, TransG [128] is developed by utilizing a Bayesian non-parametric infinite mixture model, which generates several translation components for a relation. TransG would choose the optimal match between h, t, and r that satisfy its score function. By using Gaussian representation, other probabilistic-based models like GaussianPath [129] and DBKGE [130] also take use of the uncertainty of KGs. By embedding relations as multinomial distributions and entities as Dirichlet distributions, DiriE [131] is proposed. This approach develops binary embeddings of knowledge networks for modeling complex connection patterns and uncertainty after evaluating the links between elements using Bayesian inference. Most recently, ItoE [132] introduced the formulation of relations in a KG as stochastic Ito processes, which allows transitions between two nodes to happen with a corresponding likelihood. This technique, which is a mathematical generalization of various state-of-the-art models, enables ItoE to express numerous stochastic trajectories, including loops connected to pathways. The approaches outlined above show that probabilistic embedding is not only good at gathering more ambiguous data but also at recognizing unstructured patterns [133].

2.3.2 Euclidean Space

Nayyeri et al. [134] offer a novel KGE model called FieldE that uses ordinary differential equations (ODEs) for embedding KGs into a Euclidean space in order to improve the structure preservation capabilities of KGE models.

Each relation in R^n is represented as a vector field f_{θ_r}, and each entity is represented by a vector indicated by $e(t)$, solving the ODE: $de(t)/dt = f_{\theta_r}(e(t))$ on a Riemannian manifold, where $e(t)$ lies

on a trajectory (continuous) on the manifold M. Therefore, FieldE may naturally characterize the underlying geometry and record the continuity of changes in the embedding space. Neural ODE approach [135] is introduced in TANGO [136] to learn continuous-time representations of entities and relations dynamically. ODE method may also be applied in temporal KGs. By making it easier to acquire dynamic and continuous representations of entities and relations, these analytical techniques improve parameter effectiveness, adaptive computation, and memory efficiency in a variety of ensuing tasks [137]. Other analytical vantage points in KGEs, such as derivability, differentiability, and integrability, are also worthwhile to investigate in addition to continuous analysis.

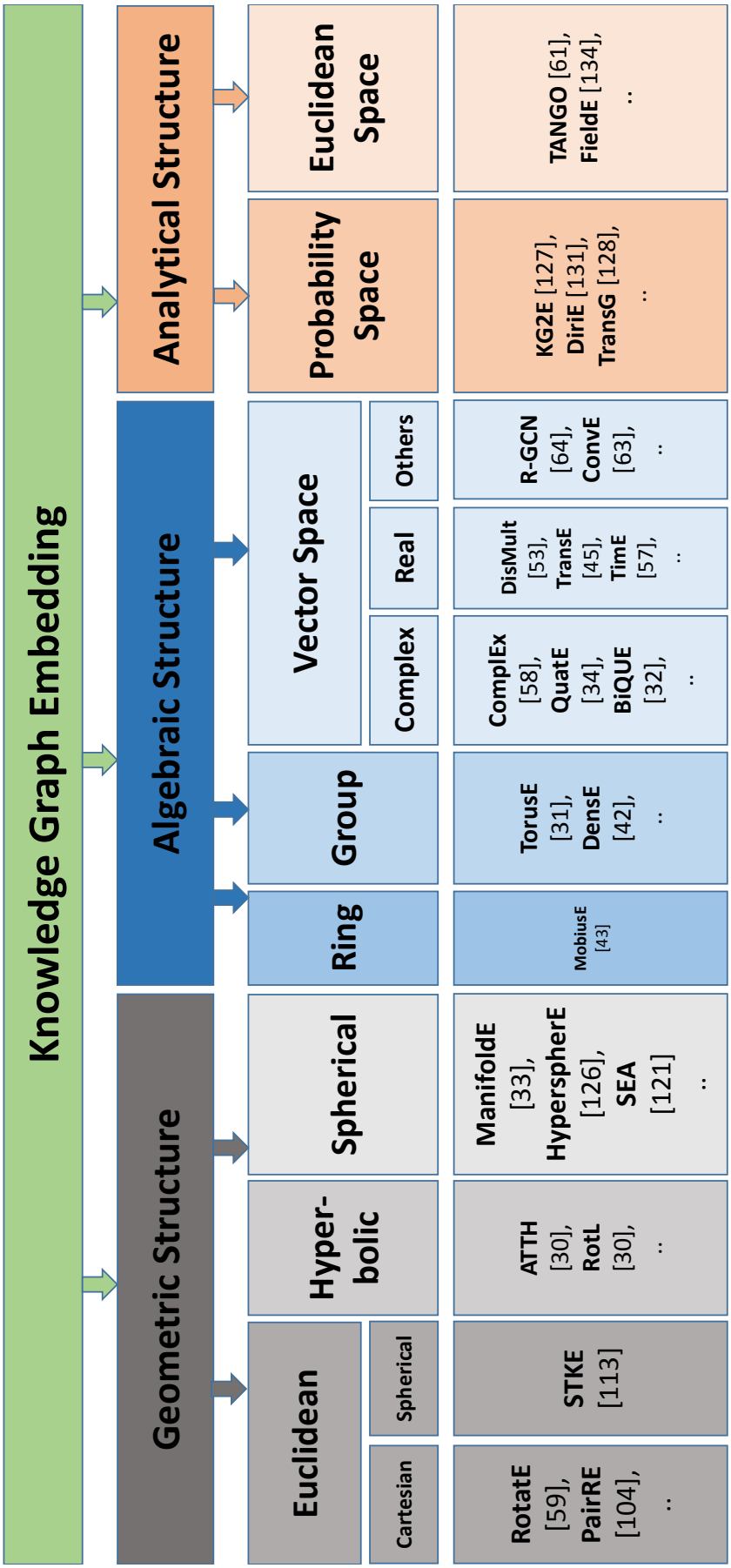
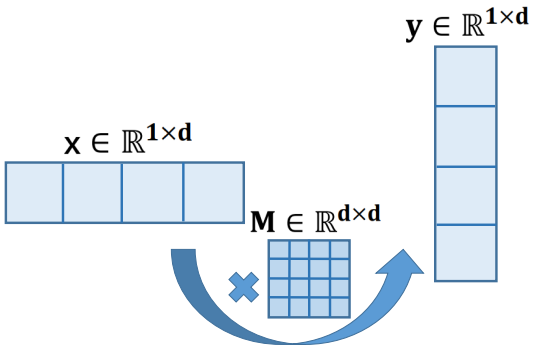
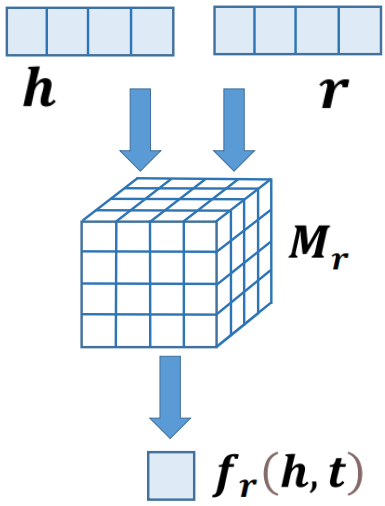


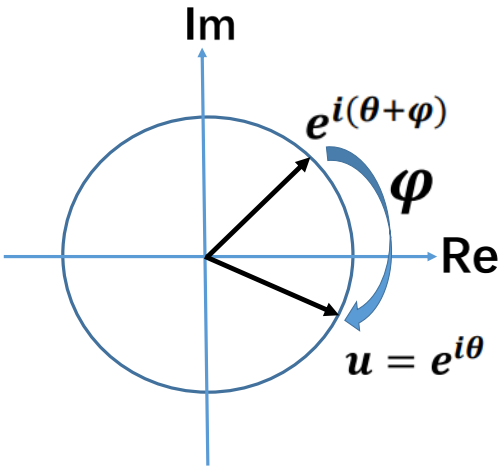
Figure 2.1 KGE classification..



(a) Real Vector Space: Tensor Projection



(b) Real Vector Space: Tensor Projection: Similarity Matching



(c) Complex Vector Space: Transformation

Figure 2.2 An illustrative example of an algebraic operation that may be used to embed knowledge graphs from an algebraic perspective.

3 TransModE: Translational Knowledge Graph Embedding Using Modular Arithmetic

3.1 Introduction

Knowledge graphs are collections of facts in form of triplets. Each triplet (h,r,t) represents the relation r connecting the head h and the tail t entities. Freebase [9], Yago [138], and WordNet [15] are real-world examples of knowledge graph in use. Knowledge graphs have proven to be effective in many areas such as recommendation systems [139], question answering [140], and natural language processing [141].

One of the main challenges facing knowledge graphs is its incompleteness. For example, some real-world knowledge graphs may contain billions of triplets, but it still miss other valid triplets. Since solving this problem manually is infeasible, link prediction has emerged as an optimal solution. Relying on the known triplets of a knowledge graph, link prediction aims to predict the missing triplets and determine the type of relationship connecting the entities of these triplets. Knowledge graph embedding has proven to be very effective link prediction method. It replaces the entities and relations with vectors of continuous numbers holding its semantics, and uses these vectors to predict and infer the patterns of the missing relations.

The relation pattern vary from one relation to another according to its type. A relation type is simple when connecting no more than two entities, otherwise it is complex relation. Each relation type is divided into patterns. Simple relation can be symmetry/anti-symmetry, composition, and inversion, whereas complex relation can be 1-to-N, N-to-1, or N-to-N. The majority of the existing knowledge graph embedding approaches attempt to represent one or more simple relation patterns while ignoring complex relations. For example, the TransE model [45] models the inversion and composition patterns; the DisMult model [53] models the symmetry pattern; RotatE model [59] can model relation patterns including symmetry/anti-symmetry, inversion, and composition, but none of the examples dealt with complex relations.

In this chapter, we propose a new knowledge graph embedding model called TransModE, inspired from TransE model. This model maps the entities to the modulus space and defines each relation as a transition between entities in this space. TransModE kept on the simplicity of TransE while greatly improving its representation ability. TransModE perfectly represents all simple and complex relation patterns: symmetry, anti-symmetry, inversion, composition, 1-to-N, N-to-1, N-to-N. To prove the effectiveness of our proposed model, we evaluate TransModE on a set of knowledge

graph benchmark datasets including FB15k-237 and WN18RR, and we compared the results to state-of-the-art approaches.

The rest of the chapter is organized as follows: Section 3.2 defines and explains the simple and the complex relation patterns, the modulus space and the proposed knowledge embedding model TransModE. The obtained simulation results are exposed in Section 3.3. Finally, we conclude the chapter and provide directions for future work in section 3.4.

3.2 Our Embedding Model: TransModE

In this section we first introduce all the patterns of simple and complex relations studied in the literature of link prediction in knowledge graph. Then we explain the modulus space characteristics and its distance measurement. Later we introduce our novel knowledge graph embedding model TransModE. Last, we mathematically prove the robustness of our proposition in representing all patterns of simple and complex relations.

3.2.1 Types of relations

The relation in knowledge representation is divided into two types, it can be either simple or complex. Each of these types have different representation patterns. For the simple type, the patterns are: symmetry, anti-symmetry, composition, inversion. It is possible for a simple relation to follow more than one pattern, but it can be either symmetry or anti-symmetry at once. For the complex type, a relation can follow only one of the three patterns: 1-to-N, N-to-1, N-to-N, beside the simple relation patterns. For example, a complex relation r can follow the patterns: 1-to-N, symmetry, composition.

3.2.1.1 Simple relation patterns

- *Symmetry*:

  A relation is said to be symmetric if the head and the tail of entities can be switched, so the relation holds in both directions: from head to tail, and from tail to head. Mathematically, a relation r is symmetric if $\forall x,y \in E$, the set of entities

$$(x,r,y) \implies (y,r,x) \tag{3.1}$$

- *Anti-Symmetry*:

  A relation follows the anti-symmetry pattern if it holds from head to tail, and does not hold from tail to head, it holds only in on direction. A relation r is anti-symmetric relation if $\forall x,y \in$ the set of entities E:

$$(x,r,y) \nRightarrow (y,r,x) \tag{3.2}$$

- *Inversion*:

  A relation follows the inversion pattern if there exist another relation that connect the same entities but in opposite direction. A relation r2 is said to be the inverse of r1, if $\forall x,y \in$ E, the set of entities of KG:

$$(x,r_1,y) \implies (y,r_2,x) \tag{3.3}$$

- *Composition*:

A relation follows the composition pattern if it can be broken down into two or more relations.For $x, y, z \in E$, a relation r is a composed relation, if $\exists\ r_1, r_2 \in R$, the set of relations in KG:

$$(x, r_1, z) + (z, r_2, y) \implies (x, r, y) \tag{3.4}$$

Where $x, y, z \in E$, the set of entities.

3.2.1.2 Complex relation patterns

- *1-to-N*:

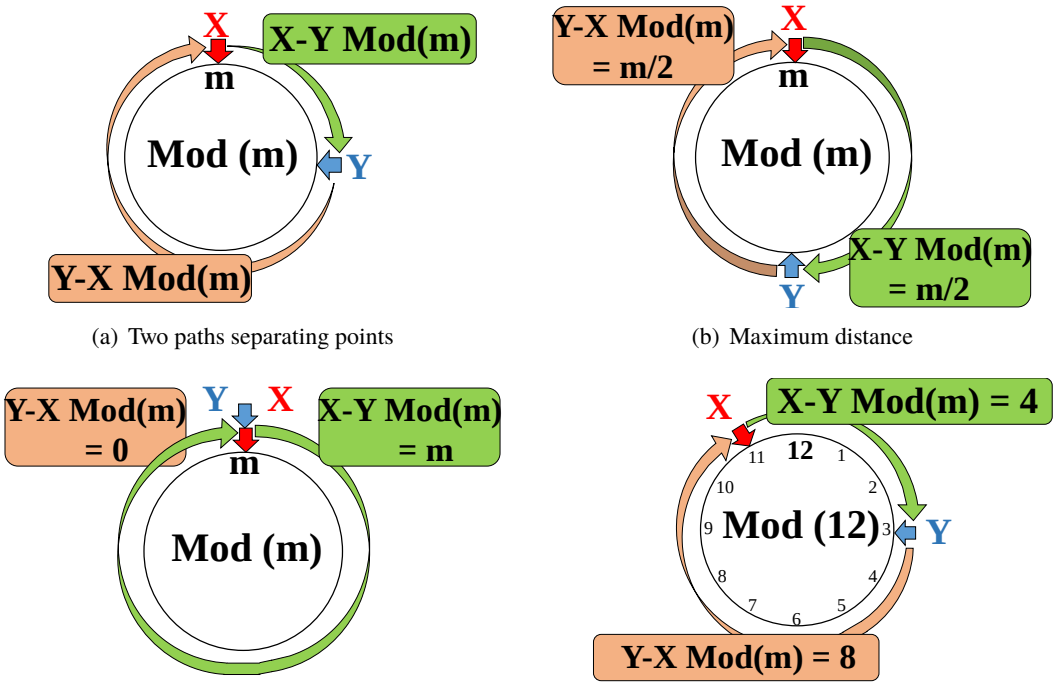
  A relation following this pattern has a single head mapped into more than one tail. In $1-N$ relation, the set of tails $T = \{t_1, t_2, ..t_N\}$ can share a unique head.

- *N-to-1*:

  A relation following this pattern has more than one head mapped into a single tail. In $N-1$ relation, the set of heads $H = \{h_1, h_2, ..h_N\}$ can share the same unique tail t.

- *N-to-N*:

  A relation following this pattern has more than one head mapped more than one tail, joining both $1-N$ and $1-N$ patterns.



(a) Two paths separating points

(b) Maximum distance

(c) Minimum distance

(d) 12-hour clock example

Figure 3.1 Distance measurement in modulus space.

3.2.2 Modular Arithmetic

The Modular Arithmetic is often called as the fifth arithmetic operation, and comes after *addition*, *subtraction*, *multiplication and division* . It has use in many practical situations, particularly in

number theory and cryptology .In its most elementary form, Modular Arithmetic done with a count that resets itself to zero every time a certain whole number N greater than one, known as the modulus (mod), has been reached. Examples are a digital clock in the 12-hour system, which resets itself to 0 every 12 hours $(N = 12)$, and a circular protractor marked in 360 degrees $(N = 360)$, as shown in Fig.3.1[d].

Examples of the use of modular arithmetic occur in ancient Chinese, Indian, and Islamic cultures. In particular, they occur in calendrical and astronomical problems since these involve cycles (man-made or natural), but one also finds modular arithmetic in purely mathematical problems.

The importance of Modular Arithmetic increases when can be combined with other arithmetic operators. In number theory, it is a fundamental tool in the solution of Diophantine equations (particularly those restricted to integer solutions). Moreover, modular arithmetic led to important $19th$-century attempts to prove Fermat's last theorem and the development of significant parts of modern algebra.

Under modular arithmetic (with $modN$), the only numbers are $0,1,2,\ldots,N-1$, and they are known as residues modulo N. Residues are added by taking the usual arithmetic sum, then subtracting the modulus from the sum as many times as is necessary to reduce the sum to a number M between 0 and $N-1$ inclusive. M is called the sum of the numbers modulo N, for example:

$$2+4+3+7 \equiv 6(mod 10), \tag{3.5}$$

where the symbol $\equiv$ is read "is congruent to", $N = 10$ *and* $M = 6$.

The Swiss mathematician Leonhard Euler introduced the idea of congruence modulo a number N and showed that this concept partitions the integers into N congruence classes. In $mod(N)$ space, the elements of each congruence class have the same remainder when divided by N. These integers are considered to be equal in $mod(N)$ space, as shown in Equation:

$$x\ mod(N) = y \quad and\ z\ mod(N) = y \implies x \ \equiv \ z \tag{3.6}$$

For example, if $N = 5$, then 4 and 9 are members of the same congruence class $\{\ldots,4,9,14,19\ldots\}$. Since each congruence class may be represented by any of its members, this particular class may be called, for example, "the congruence class of $4mod5$" or "the congruence class of $9mod5$."

In Euler's system any N numbers that leave different remainders on division by N may represent the congruence classes mod N. Thus, one possible system for arithmetic mod 5 would be $1,2,3,4$. Addition of congruence classes mod N is defined by choosing any element from each class, adding the elements together, and then taking the congruence class mod N that the sum belongs to as the answer. Euler similarly defined subtraction and multiplication of congruence classes. For example, to multiply 3 by $4(mod5)$:

$$\begin{aligned} first\ multiply &: 3 \times 4 = 12, \\ since\ 12 &\equiv 2(mod5) \\ then\ the\ solution\ is\ 3 \times 4 &\equiv 2(mod5). \end{aligned} \tag{3.7}$$

Euler showed that one would get the same result with any two elements from the corresponding congruence classes.

3.2.2.1 Negative values in modular arithmetic

The negative values is calculated exactly like the *mod* of positive numbers.
$(x\ mod\ n) = r$ if there exists some integer q such that $x = qn + r$.
Rearranging the terms, we get : $x \ - \ q \ * \ n = r$.
Notice that since x is a negative integer in the present case, q is currently a negative integer. Therefore

to find $-x \; mod \; n$, just keep adding n to x until the result is between 0 and n. As an example, let us evaluate $x \; mod \; n$ where n = 13, $x = -27$. Add 13 to -27, you get -14, add again, you get -1, and add again, you get 12. Thus, 27 mod 13 $=$ 12. In other words, in the space mod(n), negative value such as x represents moving backward from the value of n. To change a negative value x into positive value, we subtract the absolute value $|x|$ from n, so that the summation of the result y and $|x|$ equals to n.

$$n - |x| = y \Leftrightarrow x = y \; mod(n) \tag{3.8}$$

3.2.2.2 Distance measurement

Measuring distance in modulus space differs from the Euclidean distance in $2d$ space used in TransE model. In $2d$ space, there is only one path connecting two points. The length of this path is the distance separating these two points:

$$dist(x,y) = |x-y| = |y-x|. \tag{3.9}$$

However in modulus space, there are two paths connecting two points. In other words, the distance separating x and y is different from that separating y and x. These two paths are inversely correlated, the length of the first path increases, when the length of the second path decreases and vice versa, as shown in Fig.3.1[a,b,c]. The real distance between x and y in modulus space $dist(x,y)$ is the shortest path between the two points:

$$dist(x,y) = min(|(x-y)mod(m)|, |(y-x)mod(m)|) \tag{3.10}$$

In the 12-clock example in Fig.3.1[d], the distance between "3" and "11" in this mod(12) space is:

$$dist(3,11) = min(|-8mod(12)|, |8mod(12)|) = 4.$$

Basing on the distance measurement in modulus space, we have:

- *Maximum distance*: The maximum distance separating two points x and y in modulus space $mod(m)$ is obtained when y is d apart from x, where d is a multiple of $m/2$. In this case, the distances $dist(x,y)$ and $dist(y,x)$ are equal, as shown in the Fig.3.1[b]:

$$dist(x,y) = min(|-m/2 \; mod(m)|, |m/2 \; mod(m)|) = m/2. \tag{3.11}$$

- *Minimum distance*: The minimum distance separating two points x and y in modulus space mod(m) is obtained when y is congruent to x, as shown in Fig.3.1[c]:

$$dist(x,y) = min(|m \; mod(m)|, |0 \; mod(m)|) = 0. \tag{3.12}$$

3.2.3 TransModE

TransModE is a knowledge graph embedding model that extends TransE. It represents each entity by one embedding vector, and each relation by 2 embedding vectors: r_m the embedding vector defining the modulus space, and r_d the embedding vector coordinating the transition in this space. A relation (h,r,t) in TransModE is the transition from the head h to the tail t in a modulus space of the relation r_m. Meanwhile the transition embedding vector of the relation r_d regulates this transition, as the equation shows:

$$t \equiv (h + r_d)\ mod(r_m) \Leftrightarrow t = h + r_d + k * r_m, \forall\ k \in \mathbb{Z} \tag{3.13}$$

3.2.3.1 TransModE score function:

A score function of a knowledge graph embedding model has two roles which are maximizing the scores of true triplets and minimizing the scores of false triplets. In TransModE, the score function works exactly opposite to the distance equation in modulus space. For example, in a triplet (h,r,t), the closer $(h + r_d)$ is to t, the higher the triplet score will be, and the further it is the lower the score will be. For this reason, TransModE uses the negation of the distance equation in modulus space as a score function F, returning values normalized between 0 and m:

$$F = -dist(h + r_d, t) = -min((h + r_d - t) mod(r_m), (t - (h + r_d)) mod(r_m))$$

$$F = max((h + r_d - t) mod(r_m), (t - h - r_d) mod(r_m)) \tag{3.14}$$

- TransModE score function F maximizes the score of the true triplet:
  The maximum score that a triplet can obtain in the modulus space mod(m) is m, which is congruent to 0, the minimum distance between two points in the space. Thus, after performing transition operation in a true triplet, the head is congruent to the tail, as shown in Fig.3.1[c].

$$maximize(max((h + r_d - t) mod(r_m), (t - (h + r_d)) mod(r_m))) = m \equiv 0$$

$$\implies h + r_d \equiv t \tag{3.15}$$

- TransModE score function minimizes the score of the false triplet:
  Since TransModE score function opposes the distance function of modulus space, then the minimization of the score function F is equivalent to the maximization of the distance equation in modulus space. The maximum distance between 2 points in the modulus space mod(m) is $m/2$. Thus, after performing transition operation in a false triplet, the head is $m/2$ apart from the tail, as shown in Fig.3.1[b].

$$minimize(max((h + r_d - t) mod(r_m), (t - (h + r_d)) mod(r_m))) = m/2$$

$$\implies h + r_d + m/2 \equiv t$$

3.2.3.2 TransModE fully expressive model:

A Fully expressive model is that can infer any type of relation: simple or complex. A simple relation is that connecting no more than two entities, the head and the tail. According to the existing literature, the most important patterns of simple relation are: symmetry, anti-symmetry, inversion, and composition. On the other hand, complex relation is that connecting one or more heads to one or more tails, thus simple relation is a special case of complex. Complex relation can follow one of the following patterns: 1-to-N, N-to-1, N-to-N.

By defining each relation as a transition in a modulus spaces, TransModE can model and infer all the simple relation patterns introduced above. In TransModE, a triplet (h, r, t) is represented as: $t = (h + r_d)\ mod(r_m)$. Formally, we have following results with proves:

- **Lemma 1.** TransModE can infer the symmetry pattern.
  proof:
  $t \equiv (h + r_d)\ mod(r_m)$
  $t - (h + r_d) = kr_m$, for some $k \in \mathbb{Z}$
  $(h + r_d) - t = -kr_m$ and $-k \in \mathbb{Z}$
  $h + r_d \equiv t\ mod\ (r_m)$
  for r_d is a multiple of m, we obtain:
  $\Longrightarrow\ h \equiv t + r_d (mod\ r_m)$

- **Lemma 2.** TransModE can infer the anti-symmetry pattern.
  proof:
  Basing on the proof of the symmetry pattern we obtain:
  TransModE can represent Anti-symmetry$\Leftrightarrow$r$_d$ is not a multiple of r_m.

- **Lemma 3.** TransModE can infer the inversion pattern.
  proof:
  $t \equiv (h + r_d)\ mod(r_m)$
  $t - (h + r_d) = kr_m$
  (h+r$_d$) $- t = (-r_m)k$
  $h + r_d \equiv t\ mod(k)$
  $mod(r_m)$ is the inverse of $mod(k)$.

- **Lemma 4.** TransModE can infer the composition pattern.
  proof:
  $c \equiv (t + r_{d1})\ mod(r_{m1})$ and $t \equiv (h + r_{d2})\ mod(r_{m2})$.
  $c = t + r_{d1} + k * r_{m1}$, and $t = h + r_{d2} + k * r_{m2}, k \in \mathbb{Z}$.
  $c = h + r_{d2} + k * r_{m2} + r_{d1} + k * r_{m1}$
  $c = h + (r_{d2} + r_{d1}) + k(r_{m2} + r_{m1})$
  $\Longrightarrow\ c = (h + (r_{d2} + r_{d1}))mod(r_{m2} + r_{m1})$

- **Lemma 5.** TransModE can infer 1-to-N complex pattern.
  proof:
  $t \equiv (h + r_d)\ mod(r_m)$
  $t = h + r_d + kr_m$
  $t = h + r_d + (k - c + c)r_m$
  $t = h + r_d + cr_m + (k - c)r_m$

  Hence $t \equiv ((h + cr_m) + r_d)\ mod(r_m)$ and $t \equiv (h + r_d)\ mod(r_m)$

- **Lemma 6.** TransModE can infer N-to-1 complex pattern.
  proof:
  $t \equiv (h + r_d)\ mod(r_m)$
  $t = h + r_d + kr_m$
  $t = h + r_d + (k - c + c)r_m$
  $t - cr_m = h + r_d + (k - c)r_m$

  Hence $t - cr_m \equiv (h + r_d)\ mod(r_m)$ and $t \equiv (h + r_d)\ mod(r_m)$

- **Lemma 7.** TransModE can infer N-to-N complex pattern.
  proof:
  By combining the results and the proves of **Lemma 5** and **Lemma 6**, we conclude the ability of TransModE to represent N-to-N relation.

3.3 Simulation Results

In this section, we present the experimental settings in detail. We introduce the knowledge graphs used in this evaluation, the negative sampling strategy, the hyperparameter settings and the evaluation protocols.

3.3.1 Experimental settings:

3.3.1.1 Evaluation knowledge graphs

To evaluate our model, we implemented TransModE using Ampligraph [142] python library, and evaluate it on two widely used knowledge graphs: FB15k-237, and WN18RR.

FB15k-237: is a subset of Freebase [9], a large knowledge graph that stores general knowledge facts. It consists of 14951 entities, 237 relation types, and 310116 triples. It does not contain inverse relationships. The goal of link prediction on FB15k-237 is to model and infer symmetry, anti-symmetry and composition patterns.

WN18RR: is a subset of WordNet [15], a knowledge graph that clusters words into synonym groups and features lexical relationships between words. It consists of 40,943 entities, 11 relation types, and 93003 triples. WN18RR does not contain inverse relationships, it contains symmetry, anti-symmetry and composition relation patterns. The main pattern is the symmetry since almost each word has a symmetric relation in WN18RR, e.g., $also-see$ and $similar-to$ [59].

3.3.1.2 Negative Sampling

To explain about negative sampling, we should define the open world assumption (OWA) [29] and the closed world assumption (CWA) [30]. The CWA states that facts that are not observed in a graph are false, while the OWA is relaxed to assume that unobserved facts can be either missing or false. KGE models prefer the OWA due to the incompleteness nature of KGs. As a result, KGs considered to contain only ground-truth triples, which is insufficient to train KGE model. Beside the ground truth triples, KGE model requires a large number of negative triples to have an efficient training, and these negatives are obtained by negative sampling method. Thus negative sampling becomes a critical point in knowledge representation learning. That is, in a standard KG, E represents the set of entities, R represents the set of relations. $D+$ are sets of the positive triples (h, r,t) contained in a given graph. $D-$ contain the negative triples obtained through corrupting the head, the tail, or the relation of a positive triple of $D+$.

KGE methods are trained through discriminating positive samples from negative ones. The quality of generated negative samples has a direct impact on the performance of learnt knowledge representation in a myriad of downstream tasks, such as recommendation, link prediction and node classification. Poor or too obviously incorrect negative triples fail in facilitating the capture of latent semantics and easily bring about the zero loss problem. In contrast, high-quality negatives will ensure that the training smoothly moves on and the learnt knowledge representation performs.

3.3.1.3 Hyperparameter settings

We use Adam optimizer[143] to tune the hyperparameters on the training set. The loss was calculated using multiclass-nll loss function with margin $\in \{1,2,6,9,12,18,24\}$. The ranges of the hyperparameters during the training process are set as follows: embedding dimension k :

$\{200,300,350,500,600\}$, batch size $b : \{64,128,512,1024,2048\}$, number of negative samples per triplet $eta : 20,30,50,70,100,150$. And finally we use regularization with $lambda \in \{0.0001,0.05\}$ to prevent the model from overfitting. The embedding initialization follows the Xavier strategy [144], and we generate the negative samples following $< subject, object >$ corruption strategy. That is, for a triple, we randomly replace the entity in the subject or the object position by another, but not both at once.

3.3.1.4 Evaluation protocols

We applied three tests to evaluate the performance of link prediction of TransModE. These tests rely on ranking each positive test triple against all its generated negatives according to its score. We compute the ranks of the triplets following the filtered scenario, in which the valid entities outscoring the target one are not taken into consideration while ranking. Rank computation also requires defining a tie-breaking strategy to apply when multiple entities obtain the same score as the target one. In our evaluation, we choose to use the most strict policy in which we assign the worst rank to the target entity.

The first test is Mean Rank (MR) which is calculated as follow:

$$mean(rank_t) \ \forall t \in T \tag{3.16}$$

with T is the set of positive test triples, and $rank_t$ is the rank of triple t against its negatives.

The second evaluation test is Mean Reciprocal Rank (MRR). It is similar to MR, but it uses the reciprocal rank of a triple instead of its rank, what make it less sensitive to outliers [49]:

$$mean(1/rank_t) \ \forall t \in T \tag{3.17}$$

The last evaluation test is Hits@N, which counts the test triples having a rank less than or equal to N.

$$\sum t_N, \ where \ t_N \in T, rank_{t_N} >= N \tag{3.18}$$

3.3.2 Main results

We compare TransModE to several state-of-the-art models, including TransE [45], DistMult [53], ComplEx [58], RotatE [59], and ModE [145] to show the efficiency of our model in inferring the relation patterns for link prediction task. Table 3.1 shows the results of the evaluation tests of our model and the state-of-the-art models based on WN18RR and FB15K-237 datasets respectively.

WN18RR dataset contains composition, symmetry and anti-symmetry relation patterns. Referring to the results in Table 3.1, we conclude that symmetry pattern is dominating in this dataset. This outcome can be inferred from:

- The close results of DistMult and TransE, knowing that the first represents only symmetric relations while the latter represents anti-symmetry and composition patterns. This indicates the significant presence of symmetry pattern in WN18RR.
- The comparison between the results of RotatE and that of TransE, knowing that the first can represent symmetry while the latter can't, meanwhile both can represent composition and anti-symmetry patterns. In this comparison, RotatE clearly surpasses TransE just because of its ability to represent symmetry pattern.
- The significant closeness between the results of RotatE and ModE, knowing that the first can represent anti-symmetry pattern while the latter can't, meanwhile both can represent composition and symmetry patterns. This shows the weak presence of anti-symmetry pattern in WN18RR.

Dataset	WN18RR					FB15K-237			
Model	MR	MRR	Hits@1	Hits@3	Hits@10	MR	MRR	Hits@1	Hits@3	Hits@10
TransE	3384	0.226	-	-	0.501	357	0.294	-	-	0.465
DisMult	5110	0.43	0.39	0.44	0.49	254	0.241	0.155	0.263	0.419
ComplEx	5261	0.44	0.41	0.46	0.51	339	0.247	0.158	0.275	0.428
RotatE	3340	0.476	0.428	0.492	0.571	177	0.338	0.241	0.375	0.533
ModE	1898	0.472	0.427	0.486	0.564	244	0.341	0.244	0.380	0.534
TransModE	**2471**	**0.493**	**0.463**	**0.510**	**0.565**	**179**	**0.345**	**0.245**	**0.380**	**0.535**

Table 3.1 Results of models evaluation on WN18RR and FB15K-237 datasets.

- The comparison between the results of RotatE and that of complEx, knowing that the first can represent composition pattern while the latter can't, meanwhile both can represent symmetry and anti-symmetry patterns. This shows that the presence of composition pattern in WN18RR is weak, but it's better than that of anti-symmetry.

FB15k-237 dataset contains composition, symmetry and anti-symmetry relation patterns. Referring to the results in Table 3.1, we conclude that composition pattern is dominating in this dataset. This outcome can be inferred from:

- The comparison between the results of RotatE and that of complEx, knowing that the first can represent composition pattern while the latter can't, meanwhile both can represent symmetry and anti-symmetry

  patterns. The clear difference in the results shows the importance of being able to represent the composition pattern in FB15k-237.

- The significant closeness between the results of RotatE and ModE, knowing that the first can represent anti-symmetry pattern while the latter can't, meanwhile both can represent composition and symmetry

  patterns. This shows the weak presence of anti-symmetry pattern in FB15k-237.

- The comparison between the results of RotatE and that of TransE, knowing that the first can represent symmetry while the latter can't, meanwhile both can represent composition and anti-symmetry patterns. This

  comparison combined to the two previous comparisons show that this dataset contains more symmetric relations than anti-symmetric relation, but the dominant pattern is the composition.

In our experiment on WN18RR and FB15k-237, our model surpasses all the models. This superiority is actually due to TransModE ability to represent complex relations unlike other models in the experiment. Both TransModE and RotatE can represent all simple relation patterns, but only TransModE can represent complex relation patterns. This gave TransModE an advantage over its competitors and this is evident in the results. Based on the inferred points of both datasets, we have the presence of anti-symmetry pattern is the weaker in both WN18RR and FB15k-237. This can be also proved through the close results of TransModE and ModE, knowing that our model can represent anti-symmetry pattern while ModE can't, meanwhile both can represent all other simple and complex relation patterns.

3.4 Conclusion

Knowledge graph embedding is among the most prominent link prediction methods that solve the problem of incompleteness of knowledge graph. In this chapter, we propose a new knowledge graph embedding model called TransModE, which represents relation between entities as translation in modulus space. This model has the ability to represent all patterns of simple and complex relations while preserving low level of complexity. Our experimental results show that TransModE outperforms the majority of the existing state-of-the art models on two large-scale benchmarks.

4 KEMA++: A Full Representative Knowledge-Graph Embedding Model

4.1 Introduction

Knowledge graph (KG) rises recently as one of the best ways for knowledge representation. We have seen the construction of many KGs of different sizes, domains, and coverage, like Freebase [9], Yago [138], and WordNet [15]. KG is a multi-relational graph built of nodes representing real world entities such as objects and events. These entities are connected by edges representing the relations and interactions between them. KG is represented by a set of triplets that shows the relations linking the entities. KG has proven to be effective in many real-world applications like recommender systems [139], natural language processing [141], and question-answering [140].

Although a KG may consist of a huge number of entities and relations, it is usually incomplete. It is impossible for a KG to cover every single entity or relation in whole world, no matter how huge this KG is. This is called *the completeness problem* of KG. This challenge represents one of the main issues facing KGs that researchers are working to solve. Link prediction emerges as efficient way to overcome the KG completeness problem. Subsequently, link prediction is used to predict not only the existence of a relation between two entities, but also the specific type of this relation. However, these predictions are infeasible using traditional methods. So the need for novel link prediction approaches like Knowledge graph embedding arises. Knowledge graph embedding (KGE) methods have proven to be very effective applied in link prediction. KGE embeds a KG into a continuous vector space while preserving certain information of the graph. Generally, KGE replaces any object (entity, relation, ..) with a vector of continuous numbers holding this object semantics. Mainly, KGE models differ in how these numeric vectors are used. and divided into three categories accordingly. The first one is the Translational that consider relations as a motion between entities. The second category is Tensor factorization that uses tensors for embedding vectors processing. The third category is the Neural network, in which the embedding vectors are fed to neural networks for training.

In this chapter, we introduce a novel knowledge graph embedding model based on the modulus operation called as KEMA. Indeed, KEMA adopts a new way for dealing with embedding vectors away from translational, tensor factorization, and neural networks. Our model relies on the modular arithmetic mathematical operation. Since modular arithmetic is an equivalence relation, it helps handling different types of knowledge graph relations such as symmetry, inversion, and composition. Moreover, KEMA can deal with relations of complex mapping patterns like one-to-many, many-to-one, and many-to-many. The limitation of KEMA is its inability to infer anti-symmetric relation. This issue was overcomed by KEMA++, the enhanced version of KEMA ,that is a full representative model inferring all simple and complex relation patterns.To prove the effectiveness of our proposed

models KEMA and KEMA++, we evaluate both on a set of knowledge graph benchmark datasets including FB15k-237 [146] and WN18RR [63], and we compared the results to state-of-the-art approaches.

The rest of the chapter is organized as follows. Section 4.2 introduces our embedding model KEMA, explains how it works . Section 4.3 introduces the enhanced version of KEMA: KEMA++, explains its enhancements and the improvements applied to KEMA. Section 4.4 shows the experimental results of KEMA and KEMA++, and analyses the obtained results. Finally, we conclude the chapter in section 4.5.

4.2 KEMA Embedding Model

The novel idea behind KEMA is to represent the relation between two entities through modular arithmetic operation. In other words, the tail embedding vector is considered to be the projection of the head embedding vector in the modular arithmetic space of the relation embedding vector, as shown in Equation (4.1).

$$t \; = \; h \; mod \; (r) \tag{4.1}$$

where h, t, and r represent the embedding vectors of the head, the tail and the relation respectively.

The second layer of our model is called KEMA embedding model (see Fig. 4.1), and it shows the way the embedding vectors are assigned for entities and relations of a given KG. KEMA starts by assigning random vectors for entities and relations. Then, it modifies these vectors in a way it satisfies the score function shown in Equation (4.1). First, every index $E_h[j]$ in the vector of the head entity E_h is subjected to modular arithmetic operation of modulus $r[j]$, the j-th index of relation r. Then, the vector of numbers obtained from this operation is assigned to the tail entity E_3 of the relation r.

4.2.1 KEMA: different relation pattern representation

Despite the simplicity of the calculation process used in KEMA, it has proved to be highly effective and accurate compared to other models. This simplicity can also be seen in the low complexity of both training and prediction processes. As well as simple relations, KEMA can effectively handle complex relations of KG such as 1-N and N-N.

4.2.1.1 KEMA: simple relations

According to the existing literature, three types of simple relation patterns are very important: symmetric, inverse, and composed patterns. All these patterns are covered by KEMA embedding model and mathematically proved as follows:

Let $a, b, n \in \mathbb{Z}$ such that:

$$\begin{aligned} & a \equiv b (mod \; n) \\ \Longrightarrow \; & a - b = kn, \text{ for some } k \in \mathbb{Z} \\ \Longrightarrow \; & b - a = (-k)n \text{ and } -k \in \mathbb{Z} \\ \Longrightarrow \; & b \equiv a (mod \; n) \end{aligned}$$

Thus KEMA covers symmetric relations.

Let $a, b, n \in \mathbb{Z}$ such that:

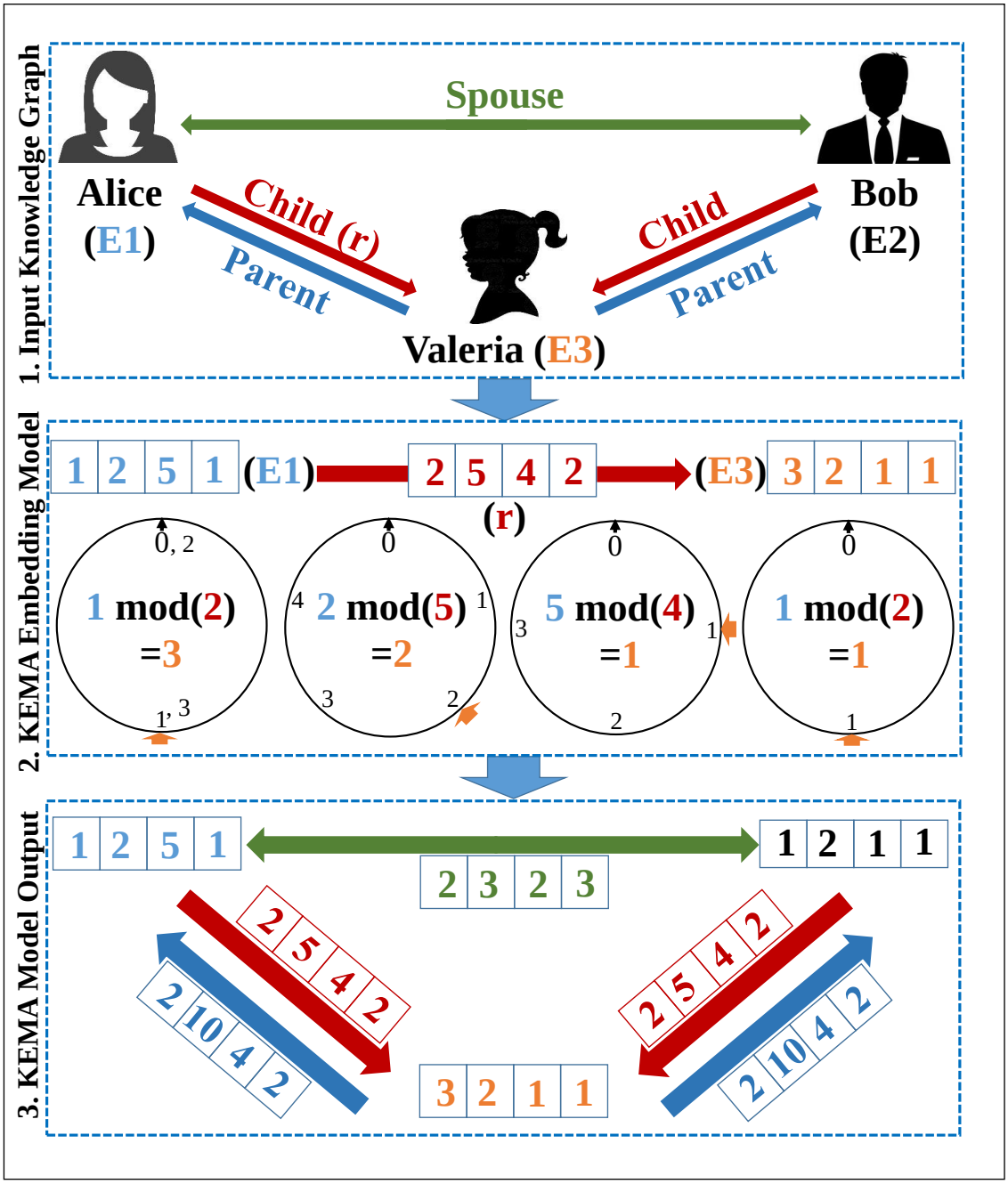


Figure 4.1 KEMA architecture..

$$
\begin{aligned}
& a \equiv b (mod\ n) \\
\implies & a - b = kn, \text{ for some } k \in \mathbb{Z} \\
\implies & b - a = (-n)k \text{ and } -n \in \mathbb{Z} \\
\implies & b \equiv a (mod\ k)
\end{aligned}
$$

mod(n) is inverse to mod(k).
Thus KEMA covers inverse relations.

Let a,b,n,$c \in \mathbb{Z}$, such that:

Table 4.1 Embedding vectors of Entities and relations.

Entity	Embedding vector	Relation	Embedding vector
Bob	[1,2,1,1]	*Spouse*	[2,3,2,3]
Alice	[1,2,5,1]	*Child*	[2,5,4,2]
Valeria	[3,2,1,1]	*Parent*	[2,4,10,2]

$a \equiv b(mod\ n)$ and $b \equiv c(mod\ n)$.
then $a = b + kn, k \in \mathbb{Z}$ and $b = c + hn, h \in \mathbb{Z}$.
$a = b + kn$
$\implies a = (c + hn) + kn$
$\implies a = c + (hn + kn)$
$\implies a = c + (h+k)n,\ h+k \in Z$.
Hence $a \equiv c(mod\ n)$.
Thus KEMA covers composed relations.

4.2.1.2 KEMA: Complex Relations

The majority of the models proposed in the literature can deal with simple relations between KG entities, i.e 1-to-1 relations. However, relying on simple relationships to build knowledge is impractical. Contrarily, KEMA has the ability to handle both simple and complex relationships.

Let $a, b, n, c \in \mathbb{Z}$ such that:
$a \equiv b(mod\ n)$

then $a = b + kn,\ \ k \in \mathbb{Z}$
$\implies a = b + (k - c + c)n$
$\implies a = b + cn + (k-c)n,\ \ k - c \in Z$.
Hence $a \equiv b + cn(mod\ n)$ and $a \equiv b(mod\ n)$

Thus KEMA covers 1-N relations.

Given the 1-N relation:

$a \equiv b + cn(mod\ n)$ and $a \equiv b(mod\ n)$

Since modular arithmetic is symmetric relation, then:

$b + cn \equiv a(mod\ n)$ and $b \equiv a(mod\ n)$

Thus KEMA covers N-1 relations.

By combining the 1-N and N-1 modular arithmetic relations, we conclude its ability to represent N-N relation, and thus modulus can represent all the complex relation patterns.

4.2.2 Analytical example

In this section, we illustrate an example to show the effectiveness of KEMA in terms of representing simple and complex relations. According to the input knowledge graph layer shown in figure 4.1, the sub-graph shows the relations between three entities. Table 4.1 shows the embedding vectors that KEMA assigned to every entity and relation.

Table 4.2 Symmetric relation Example.

Head	Relation	Tail (result)
[1,2,1,1]	[2,3,2,3]	[1,2,5,1]
[1,2,5,1]	[2,3,2,3]	[1,2,1,1]

Table 4.3 Inverse relations Example.

Head	Relation	Tail (result)
[1,2,5,1]	[2,4,10,2]	[3,2,1,1]
[3,2,1,1]	[2,4,5,2]	[1,2,5,1]

The relation "*Spouse*" is an example of the symmetric relation. In Fig. 4.1, the output layer of KEMA shows that this relation holds in both directions. Moreover, in Table 4.2, the first row shows that the tail of the relation "Spouse" with head entity "Bob" is "Alice". On the other hand, the second row represents the opposite direction, where the tail of the relation "Spouse" with head entity "Alice" is "Bob".

In Fig. 4.1, the relations "Child" and "Parent" show the inversion pattern of simple relations. In the first row of the Table 4.3, "Alice" is the head of the relation "Child" while "Valeria" is its tail. In the opposite direction, the second row shows the relation "Parent", where "Valeria" is the head and "Alice" is the tail. Then the relations "Parent" and "Child" are said to be inverse.

Furthermore, Fig. 4.1 shows that the relation "Spouse" is a composed relation. Fig. 4.1 shows that the relation "Child" of head "Alice" and tail "Valeria", followed by the relation "Parent" of head "Valeria" and tail "Bob", can be replaced by the relation "Spouse" having the same head as "Child", and the same tail as "Parent".

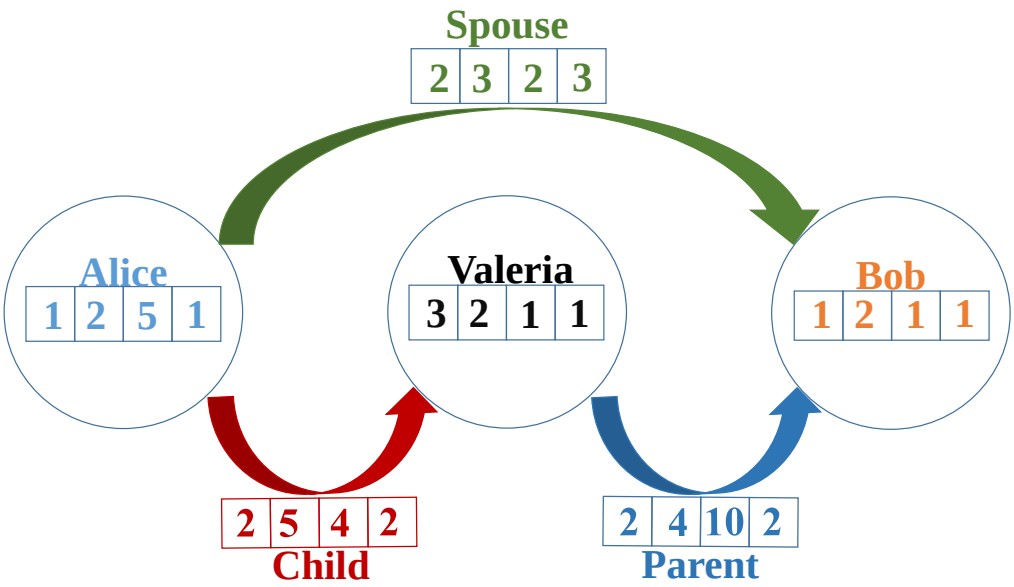


Figure 4.2 Composed relation example..

Indeed, the strength of KEMA in terms of representing complex relations is shown through the relations "Parent" and "Child" in Fig. 4.1. Since "Parent" has one head which is "Valeria", and two

tails which are "Bob" and "Alice", then it is a complex relation of $1-N$ mapping pattern. Whereas "Child" relation has two heads "Bob" and "Alice" connected to one tail "Valeria", representing N-1 pattern.

4.3 Enhanced KEMA: KEMA++

The most influential factor in the success of the existing knowledge graph embedding models is their ability to model various relation patterns, such as symmetry, antisymmetry, inversion, and composition. Unfortunately, only few existing models have the ability to model complex relation patterns along with the simple relation patterns mentioned above. For example, a relation "locatedIn" can have hundreds of heads, in the case of searching a gas station in a certain city. Meanwhile the same relationship can have hundreds of tails in the case of distribution of outlet branches across cities.

In this section, we introduce an enhanced version of KEMA called KEMA++. Indeed, KEMA model relies on representing the relation between two entities as modular arithmetic mathematical operation. Such model has the ability to model all simple and complex relation patterns except anti-symmetric. In this section, we are introducing KEMA++, an enhanced version of KEMA that is a full representative model inferring all simple and complex relation patterns.

To model the complex patterns like 1-N, N-1 or N-N, KEMA++ takes benefit from the advantages of modular arithmetic where the projection operation is used to enhance the accuracy of representations of simple relations, such as symmetric, anti-symmetric, inverse, transitive and composed. Therefore, the major enhancements of KEMA++ are:

- We combine the modular arithmetic operation used in KEMA with the projection operation. This step enhanced the modeling accuracy of simple and complex relation patterns.
- Unlike KEMA, KEMA++ can model and infer all simple and complex relation patterns, particularly the anti-symmetry pattern.
- We add a new pattern to the set of simple relation patterns called as transitive, and we mathematically prove the ability of KEMA++ to model and infer it.
- We test KEMA++ on a large scale knowledge graphs such as WN18 [45] and FB15K [45] in order to prove its scalability.
- Beside the analytical study, we add an experimental verification of KEMA++ to test its efficiency in inferring complex relations, i.e. 1-N, N-1, and N-N.
- We compare the results of KEMA++ to the results of up-to-date models like StructurE [147], QuatE [148], and ConvR [149]. The results show that KEMA++ gives:
  - The best scores in MR and Hits@1 tests for WN18RR dataset, and in Hits@1 for WN18.
  - The second best score in Mean Rank (MR) and Mean Reciprocal Rank (MRR) tests for WN18 and MRR for WN18RR. As well as, the second best results in Hits@1, Hits@10, and MRR for FB15K.

4.3.1 KEMA++ Core:

The objective of KEMA++ is to predict the missing links connecting the entities of a given knowledge graph. First, the targeted knowledge graph is given as an input to KEMA++ model. This model processes the knowledge graph components and embeds it to a low dimension continuous space.

Finally, KEMA++ returns a representative numerical vector for every entity and relation in the KG. These representative numerical vectors are then used for predicting links between KG entities.

KEMA++ is a knowledge graph embedding model that represents each entity by one embedding vector, and each relation by 2 embedding vectors: r_p the embedding vector used for projection of the head , and r_m the embedding vector defining the modular space. For a triplet (h,r,t), KEMA++ plots the result of head projection $(h \times r_p)$ in modular space of the relation r_m to obtain t, as Eq. (4.2) shows.

$$t \equiv (h \times r_p)\ mod(r_m) \Leftrightarrow t = (h \times r_p) + k \times r_m, \forall\ k \in \mathbb{Z} \tag{4.2}$$

We have $\times$ *and* $mod() : \mathrm{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^n$.
Let $\times$ denote the product between two vectors, that is: $[a \times b]i = [a]i \times [b]i$; and Let mod() denote the modulo of a vector in a modular space having another vector as modulus, that is: $a[i]mod(b[i]) = a[i] - k \times [b]i$, such that $k \in \mathbb{Z}$.

For a triplet (h,r,t) in KEMA++, the closer $(h \times r_p)$ is to t, the higher the triplet score will be. This implies that the score function in KEMA++ opposes the distance function in modular space. For this reason, KEMA++ uses the negation of the distance function in modulus space as a score function F:

$$\begin{gathered} F = -dist(h \times r_p, t) \\ F = -min((h \times r_p - t)mod(r_m), (t - (h \times r_p))mod(r_m)) \\ F = max((h \times r_p - t)mod(r_m), (t - (h \times r_p))mod(r_m)) \end{gathered} \tag{4.3}$$

KEMA++ starts by assigning random vectors for entities and relations. Then, it modifies these vectors in a way it satisfies the score function shown in Eq. (4.3). In Algorithm 1, the head is multiplied by r_p, the relation projection vector. Then the result is plotted to modular space of modulus r_m. The vector of numbers obtained from this operation is assigned to the tail entity t of the relation r.

Algorithm 1 KEMA++ Algorithm.

Require:
h: The embedding vector of the head entity
r_p: The projection embedding vector of the relation
r_m: The modulo embedding vector of the relation
Ensure: t: The embedding vector of the tail entity.
j = 0
while $j < h.length$ **do**
    $t[j] = h[j] \times r_p[j]mod(r_m[j])$
    j++
end while
return t

4.3.2 Types of Relations

Despite the simplicity of the calculation process used in KEMA++ , it has proved to be highly effective and accurate compared to other models. This simplicity can also be seen in the low complexity of both training and prediction processes. As well as simple relations, KEMA++ can effectively

handle complex relations of KG such as $1-N$, $N-1$, and $N-N$.

4.3.2.1 Simple Relations

In addition to the four simple relation patterns defined in TransModE chapter, we define another simple pattern called transitive. Then we mathematically prove that KEMA++ perfectly cover them all. Each of the following lemmas corresponds to a relation pattern proof:

- [Transitive Relation in KEMA++]

  For entities x, y, and z, a relation r is a transitive relation, if for any instances (x,r,y) and (y,r,z) of relation r, (x,r,z) is also an instance of r.

$$(x,r,y)+(y,r,z) \implies (x,r,z) \tag{4.4}$$

  Demostración. Transitive relation can be proven in KEMA++ as follows:

$$\begin{array}{rcl} t \equiv (h \times r_p)\ mod\ (r_m) & \text{and} & c \equiv (t \times r_p)\ mod\ (r_m) \\ t = h \times r_p + k \times r_m & \text{and} & c = t \times r_p + l \times r_m \\ c = h \times (r_p)^2 & + & r_m(k \times r_p + l) \\ c \equiv (h \times (r_p)^2) & & mod(r_m) \end{array}$$

  KEMA++ infers transitive pattern $\Leftrightarrow r_p = 1$

  this Lemma is proved and KEMA++ infers transitive pattern. ■

- [Symmetric Relation in KEMA++]

  Demostración. Symmetric relation can be proven in KEMA++ as follows:

$$\begin{array}{rcl} t & \equiv & (h \times r_p)\ mod(r_m) \\ t-(h \times r_p) & = & k \times r_m, \qquad \text{for some } k \in \mathbb{Z} \\ (h \times r_p)-t & = & -k \times r_m \quad \text{and} \quad -k \in \mathbb{Z} \\ h \times r_p & = & t-k \times r_m \\ h & = & \dfrac{1}{r_p} \times t - \dfrac{k}{r_p} \times r_m \end{array}$$

  for r_p is equal to 1, we obtain:

$$\begin{array}{rcl} h & \equiv & t \times mod(r_m) \\ r_p & = & 1 \end{array}$$

  Lemma 1 is proved and KEMA++ infers symmetric pattern. ■

- [Anti-Symmetric Relation in KEMA++]:

  Demostración. Anti-Symmetric relation can be proven in KEMA++ as folows:

  Basing on the proof of lemma 1 we obtain:

KEMA++ infers Anti-symmetric$\Leftrightarrow r_p \neq 1$.

Lemma 2 is proved and KEMA++ infers composed pattern. ■

- [Composed Relation in KEMA++]

Demostración. Composed relation can be proven in KEMA++ as follows:

$$\begin{aligned} c &\equiv (t \times r_{p1})\ mod(r_{m1})\ \ and\ \ t \equiv (h \times r_{p2})\ mod(r_{m2}) \\ \text{For}\quad k &\in \mathbb{Z}, \text{ we obtain: .} \\ c &= t \times r_{p1} + k \times r_{m1},\quad and\ \ t = h \times r_{p2} + k \times r_{m2} \\ c &= (h \times\ r_{p2} + k \times r_{m2}) \times r_{p1} + k \times r_{m1} \\ c &= h\ \times\ (r_{p2} \times r_{p1}) + (k \times r_{m2} \times r_{p1}) + k \times r_{m1} \\ c &= h \times (r_{p2} \times r_{p1}))\ mod\ ((r_{m2} \times r_{p1}) + r_{m1}) \end{aligned}$$

The later is a relation composed of :

$$c \equiv (t \times r_{p1}) mod(r_{m1}) \quad and \quad t \equiv (h \times r_{p2})\ mod(r_{m2}).$$

Lemma 3 is proved and KEMA++ infers composed pattern. ■

- [Inverse Relation in KEMA++]

Demostración. Inverse relation can be proven in KEMA++ as follows:

$$\begin{aligned} t &\equiv (h \times r_p)\ mod\ (r_m) \\ t &= (h \times r_p) + kr_m, \quad \text{for some } k \in \mathbb{Z} \\ h &= (\frac{1}{r_p}) \times t - (\frac{k}{r_p}) \times r_m \\ h &\equiv t \times (\frac{1}{r_p})\ mod\ (\frac{r_m}{r_p}) \end{aligned}$$

The later equation is inverse of : t$\equiv\ (h \times r_p)\ mod(r_m)$

Lemma 4 is proved and KEMA++ infers inverse pattern. ■

- [Transitive Relation in KEMA++]

Demostración. Transitive relation can be proven in KEMA++ as follows:

$$\begin{aligned} t \equiv (h \times r_p)\ mod\ (r_m) \quad &\text{and} \quad c \equiv (t \times r_p)\ mod\ (r_m) \\ t = h \times r_p + k \times r_m \quad &\text{and} \quad c = t \times r_p + l \times r_m \\ c = h \times (r_p)^2 \quad &+ \quad r_m(k \times r_p + l) \\ c \equiv (h \times (r_p)^2) \quad &\quad mod(r_m) \end{aligned}$$

KEMA++ infers transitive pattern $\Leftrightarrow r_p = 1$

Lemma 5 is proved and KEMA++ infers transitive pattern. ■

4.3.2.2 Complex Relations

In contrast to the majority of the existing models, KEMA++ has the ability to perfectly cover all complex relationships. Here is proves for KEMA++ ability to represent all complex patterns, i.e $1-N, N-1, N-N$:

- [$1 \rightarrow N$ Relation in KEMA++]

  Demostración. $1 \rightarrow N$ relation can be proven in KEMA++ as follows:

$$\begin{aligned} t &\equiv (h \times r_p)\ mod(r_m) \\ t &= (h \times r_p) + k \times r_m \\ t &= (h \times r_p) + r_m \times (k + c - c) \\ t + (c \times r_m) &= (h \times r_p) + (k + c) \times r_m \\ t + (c \times r_m) &= (h \times r_p)\ mod\ (r_m) \end{aligned}$$

  Hence we obatin:

$$\begin{aligned} t + (c \times r_m) &\equiv (h \times r_p) mod(r_m) \\ \text{and} \quad t &\equiv (h \times r_p) mod(r_m) \end{aligned}$$

  Lemma 6 is proved and KEMA++ infers $1 \rightarrow N$ complex pattern. ■

- [$N \rightarrow 1$ Relation in KEMA++]

  Demostración. $N \rightarrow 1$ relation can be proven in KEMA++ as follows:

$$\begin{aligned} t &\equiv (h \times r_p)\ mod(r_m) \\ t &= (h \times r_p) + k \times r_m \\ t &= (h \times r_p) + r_m \times (k + c - c) \\ t &= (h \times r_p) - (c \times r_m) + (k + c) r_m \\ t &= ((h \times r_p) - (c \times r_m)) mod(r_m) \end{aligned}$$

  Hence we obtain:

$$\begin{aligned} t &\equiv (h \times r_p)\ mod(r_m) \quad \text{and} \\ t &\equiv ((h \times r_p) - (c \times r_m)) mod(r_m) \end{aligned}$$

  Lemma 7 is proved and KEMA++ infers $N \rightarrow 1$ complex pattern. ■

- [$N \rightarrow N$ Relation in KEMA++]

  $N-N$ complex relation can have several heads and several tails at once, joining both $1-N$ and $1-N$ patterns. By combining the results and the proves of Lemma 6 and Lemma 7, we conclude the ability of KEMA++ to represent $N \rightarrow N$ relation.

4.3.3 Illustrative example

In this section, we illustrate an example to show the effectiveness of KEMA++ in terms of representing simple and complex relations, and how it is used to predict missing links in a knowledge graph. As shown in Fig.4.3, the given knowledge graph is built of 5 entities which are: *Father*, *Mother*, *Child*1, *Child*2, *Child*3; and 4 relations which are:

- **"***ParentOf***"** relation: is an anti-symmetric relation, in which head and tail can not be switched. It also follows the inverse pattern having "*ChildOf*" as inverse. Moreover, it is a complex relation following the patterns: $1 \rightarrow N$, having more than one tail (*Child*1, *Child*2, and *Child*3) for one head (*Father* or *Mother*); and $N \rightarrow N$ having more than one tail (*Child*1, *Child*2, and *Child*3) for more than one head (*Father* and *Mother*).

- **"***ChildOf***"** relation: is an anti-symmetric relation, in which head and tail can not be switched. It also follows the inverse pattern having "*ParentOf*" as inverse. Moreover, it is a complex relation following the patterns: $N \rightarrow 1$, having more than one head (*Child*1, *Child*2, and *Child*3) for one tail (*Father* or *Mother*); and $N \rightarrow N$ having more than one tail (*Father* and *Mother*) for more than one head (*Child*1, *Child*2, and *Child*3).

- **"***SisterOf***"** relation: is a symmetric relation, in which head and tail can be switched. It also follows the transitive pattern. Moreover, it is a complex relation following the patterns: $N \rightarrow 1$, having more than one head (*Child*1 and *Child*2) for one tail (*Child*3); $1 \rightarrow N$, having more than one tail (*Child*2 and *Child*3) for one head (*Child*1); and $N \rightarrow N$ having more than one tail (*Child*1, *Child*2, and *Child*3) for more than one head (*Child*1, *Child*2, and *Child*3).

- **"***MarriedTo***"** relation: This is a symmetric relation, in which head and tail can be switched. It also follows the Composed pattern, it can be broken down to: *ParentOf* + *Child*(1/2/3) + *ChildOf*.

  After the knowledge graph is given to our model, KEMA++ assigns two random vectors r_p and r_m for each relation, and a single random vector for each entity. Then KEMA++ keep modifying these vectors till it satisfy its function f which is:

$$t = (h \times r_p) mod (r_m) \tag{4.5}$$

  So that for a triplet (h,r,t) in a modular space of modulus r_m, the result of multiplying the head vector h and the relation project vector r_p must be congruent to the tail vector t.

  In this example, we fixed the embedding vector size to 2. Table 4.4 shows a single embedding vector assigned by KEMA++ for each entity in the given knowledge graph. Whereas Table 4.5 shows the projection and the modulo embedding vectors of the relations in the knowledge graph.

  In Table 4.4, the embedding vector of each entity was chosen while taking into account the pattern of the connected relations, and the position of this entity in these relation (head/tail).

  For example, *Father* embedding vector was determined by KEMA++ in a way that all the relations connected to this entity holds. The entity *Father* is:

  - the tail of 3 *ChildOf* relations of different heads(*Child* 1, *Child* 2, *Child* 3),
  - the head of 3 *ParentOf* relations of different tails(*Child* 1, *Child* 2, *Child* 3),
  - the head/tail of the symmetric relation *MarriedTo*.

Every single connection mentioned in the list above must hold when applied to KEMA++ score function.

In the Table 4.5, we see that the relations *MarriedTo* and *SisterOf* have project vectors of '1's, that's due to the fact that they are symmetric and transitive, which is consistent with $lemma_{sym}$[KEMA++] and $lemma_{trans}$[KEMA++] in 4.3.2 above. By having project vector of '1's, KEMA++ have the functionality of KEMA model [150], which only assigns modulo vector for each relation, and this prevents it from representing anti-symmetric relations. The project and modulo embedding vectors of each relation was chosen while taking into account the pattern of the relations, and all heads and tails of this relation instances. For example, *ParentOf* project and modulo vectors were determined by KEMA++ such that it must hold for every single connection when applied to KEMA++ score function. The relation *ParentOf* has one head (*Father*/*Mother*) and three tails(*Child* 1, *Child* 2, *Child* 3)

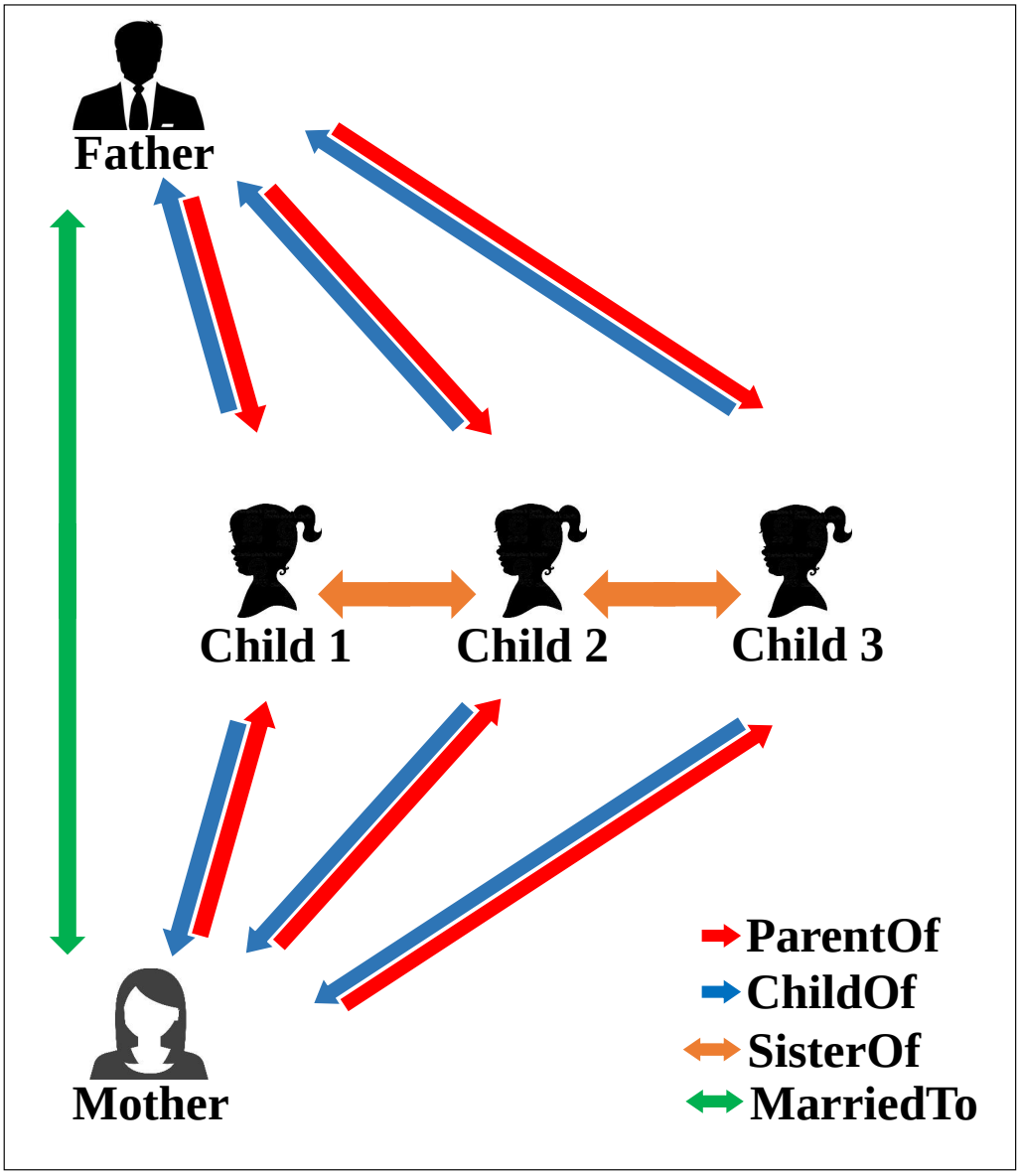


Figure 4.3 Illustrative knowledge graph example..

Table 4.4 Embedding vectors of Entities .

Entity	Embedding vector
$Father$	[5,1]
$Mother$	[35,1]
$Child1$	[12,25]
$Child2$	[48,25]
$Child3$	[66,25]

Table 4.5 Embedding vectors of relations .

Relation	Project vector	Modulo vector
$MarriedTo$	[1,1]	[2,3]
$SisterOf$	[1,1]	[6,5]
$ParentOf$	[6,8]	[18,7]
$ChildOf$	[5,2]	[5,7]

Finally, the obtained embedding vectors shown in Table 4.4 and Table 4.5 are used in predicting missing links. Unlike the other links in Fig.4.3, the faded orange arrow connecting *Child* 1 to *Child* 3 was not fed to KEMA++ model, this is a missing link that we predict according to the connections in the given knowledge graph. In this case, we have a missing triplet $(Child\ 1, SisterOf, Child\ 3)$ that we need to test its plausibility. here, we use the embedding vectors of $SisterOf$ and *Child* 1 in KEMA++ score function, if the returned result is congruent to *Child* 3 embedding vector, then this triplet holds and its true according to KEMA++ , otherwise the triplet is false and there is no $SisterOf$ relation connecting *Child* 1 and *Child* 3.

4.4 Simulation Results

In this section, we will show the experiment setting to implement our model followed by the discussion of the obtained results.

4.4.1 Experimental setting

To evaluate our model, we implemented KEMA++ using Ampligraph [142] python library. Then, we compare it to the state-of-the-art models on commonly used benchmark datasets: WN18, FB15K, WN18RR, and FB15K-237.

- WN18 is extracted from WordNet [15], which contains semantic knowledge of English lexical relations and has 40943 entities and 18 relations.
- FB15K is extracted from the large-scale knowledge database Freebase [9], which describes the general facts in the real world. It has 592,213 entities and 1,345 relations.
- WN18RR is a subset of WordNet, a KG that clusters words into synonym groups and features lexical relationships between words. It consists of 40,943 entities, 11 relation types, and 93003 triples. WN18RR contains symmetric, anti-symmetric and composed relation patterns. The main pattern is the symmetric since almost each word has a symmetric relation in WN18RR, e.g., $also-see$ and $similar-to$ [59].
- FB15K-237 is a subset of Freebase, a large knowledge graph that stores general knowledge facts. It consists of 14951 entities, 237 relation types, and 310116 triples. The

main patterns of the relation in FB15K are symmetric, anti-symmetric and composed [59].

We use Adam optimizer[143] to tune the hyperparameters on the training set. We calculate the loss using multiclass-nll loss function with *margin* 1, 2, 6, 9, 12, 18, 24. The ranges of the hyperparameters during the training process are set as follows: embedding dimension k : 200, 300, 350, 500, 600, 700, 900, batch size b : 64, 52, 96, 128, 512, 1024, 2048, number of negative samples per triplet eta : 20, 30, 50, 70, 100, 150. And finally we use regularization with $lambda$ 0.0001, 0.05 to prevent the model from overfitting. The embedding initialization follows the Xavier strategy [144], and we generate the negative samples following $< subject, object >$ corruption strategy. That is, for a triple, we randomly replace the entity in the subject or the object position by another, but not both at once. This is what is called the local closed world assumption.

We applied three tests to evaluate the performance of link prediction of KEMA++ . These tests rely on ranking each positive test triple against all its generated negatives according to its score. We compute the ranks of the triplets following the filtered scenario, in which the valid entities outscoring the target one are not taken into consideration while ranking. Rank computation also requires defining a tie-breaking strategy to apply when multiple entities obtain the same score as the target one. In our evaluation, we choose to use the most strict policy in which we assign the worst rank to the target entity The first test is Mean Rank (MR) which is calculated as follow:

$$mean(rank_t)\ \forall t \in T \tag{4.6}$$

with T is the set of positive test triples, and $rank_t$ is the rank of triple t against its negatives.

The second evaluation test is Mean Reciprocal Rank (MRR). It is similar to MR, but it uses the reciprocal rank of a triple instead of its rank, what make it less sensitive to outliers [52]:

$$mean(\frac{1}{rank_t})\ \forall t \in T \tag{4.7}$$

The last evaluation test is Hits@N, which counts the test triples having a rank less than or equal to N.

$$\sum t_N,\ where\ t_N \in T, rank_{t_N} >= N \tag{4.8}$$

4.4.2 Main Results

In order to evaluate the link prediction task in the KG, we conducted a set of extensive experiments on four common datasets , i.e. mentioned above. Tables 4.6 and 4.7 show the main results of our proposed model KEMA++ along with the results of several state of the art methods. The following observations are eminent based on the results shown on Table 4.6:

– Regarding WN18RR, KEMA++ obtains the best scores with respect to MR and Hits@1, which is better than the recent state of the art RotatE, Rot-Pro, ReflectE, StructurE, and QuatE. Moreover, KEMA++ obtains the second best score for the MRR test. Hits@1 is improved by 18 % compared to state of the art, and 1.7 % with respect to KEMA. In addition, our model improve MR test by 18.2 % compared to state of the art and 0.3 % compared to KEMA.

– Regarding FB15K-237, KEMA++ shows important results compared to all state of the art models in terms all the tests, i.e. MR, MRR, H@1, H@3 and H@10. Particularly, KEMA++ applied on FB1K-237 outperforms all results of KEMA according to the following percentages: 20% on MR, 9.8% on MRR, 10% on Hits@1, 10.7% on Hits@3, and 9.1% for Hits@10. This is due to the fact that 95% of FB15K-237 relations are anti-symmetry [151] where KEMA++ can work perfectly.

Table 4.7 shows the results of our experiments on two large scales datasets, i.e. WN18 and FB15K. The test leakage, which counts for the triples existing in both the training and testing sets, accounts for almost 81% of the triples in the test set of FB15K and 94% WN18 [146]. The observations shown in Table 4.7 are as follows:

– Regarding FB15K, KEMA++ obtains the second best scores for each of MRR, Hits@1 and Hits@3 with a small difference compared to those obtained with the model having highest score. Similarly, KEMA++ achieved a high MRR score (i.e. 0.801) compared to StructurE that achieved a near results with a difference less than 0.004.Furthermore, KEMA++ and StructurE outperform all other models in terms of Hits@1 with close scores of 0.747 and 0.750 respectively. Lastly, KEMA++ achieves a high Hits@10 score, i.e. 0.893, just after the QuatE model that achieved the best score, i.e. 0.900.
– Regarding WN18, KEMA++ achieved the best Hits@1 score (i.e. 0.950) followed by ConvR model, with a percentage of 0.32%. Whereas, KEMA++ achieved the second best score with respect to MRR compared to the best model, i.e. ConvR, with a small difference of 0.001.

Table 6 shows the scores of KEMA++ for different complex relation patterns in FB15K. For the head prediction process, the results show that KEMA++ achieved a competitive results compared to other models as follows:

– It achieved the second best scores in both Hits@10 and MRR for *N*-*N* pattern.
– It achieved the second best Hits@10 score for both 1-*N* and *N*-1 patterns.
– It achieved the second best Hits@10 score for 1-1 pattern.

Similarly, KEMA++ achieved a competitive results for the tail prediction process as follows:

– The best Hits@10 score for 1-*N* pattern that improves the result by 0.2%.
– The second best Hits@10 score for 1-1 pattern.

4.4.3 Further Analysis

Table 7 shows the score functions used in each model while comparing their complexities in terms of the variable number needed for each model. By analyzing the complexity in ascending order, the results show the following:
1) KEMA and TransE have the lowest complexity compared to other models, because both of them assign one embedding vector for each entity and one embedding vector each relation. Subsequently, the complexity of KEMA is less than that of TransE.

2) KEMA++ complexity is a bit more than that of KEMA and TransE models. This is due to the fact that KEMA++ assigns one embedding vector for each entity, but two embedding

Table 4.6 Results of models evaluation on WN18RR and FB15K-237 datasets. The best obtained scores are made in bold and the second best results are underlined.

	WN18RR					FB15K-237				
Model	MR	MRR	Hits@1	Hits@3	Hits@10	MR	MRR	Hits@1	Hits@3	Hits@10
TransE	3384	0.226	-	-	0.501	357	0.294	-	-	0.465
DisMult	5110	0.43	0.39	0.44	0.49	254	0.241	0.155	0.263	0.419
ComplEx	5261	0.44	0.41	0.46	0.51	339	0.247	0.158	0.275	0.428
RotatE	3340	0.476	0.428	0.492	0.571	177	0.338	0.241	0.375	0.533
Rot-Pro	2815	0.457	0.397	0.482	0.577	201	0.344	0.246	0.383	0.540
ReflectE	-	**0.488**	**0.450**	0.501	0.559	-	**0.358**	**0.263**	**0.396**	0.546
StructurE	2865	0.479	0.425	0.500	**0.585**	160	0.351	0.252	0.390	0.546
QuatE	2314	**0.488**	0.438	**0.508**	0.582	**87**	0.348	0.248	0.382	**0.550**
KEMA	1898	0.479	0.442	0.486	0.543	244	0.311	0.223	0.342	0.486
KEMA++	**1891**	0.487	**0.450**	0.488	0.553	197	0.345	0.248	0.383	0.535

Table 4.7 Results of models evaluation on WN18 and FB15K datasets. The best obtained scores are made in bold and the second best results are underlined.

	WN18					FB15K				
Model	MR	MRR	Hits@1	Hits@3	Hits@10	MR	MRR	Hits@1	Hits@3	Hits@10
TransE	-	0.495	0.113	0.888	0.943	-	0.463	0.297	0.578	0.749
ComplEx	-	0.941	0.936	0.945	0.947	-	0.692	0.599	0.759	0.840
RotatE	309	0.949	0.944	0.952	0.959	40	0.797	0.746	0.830	0.884
StructurE	249	**0.951**	0.945	0.953	**0.960**	44	**0.805**	**0.750**	**0.843**	0.893
QuatE	**162**	0.950	0.945	0.954	0.959	**17**	0.782	0.711	0.835	**0.900**
ConvR	-	**0.951**	0.947	**0.955**	0.958	-	0.782	0.720	0.826	0.887
KEMA++	175	0.950	**0.950**	0.953	0.955	41	0.801	0.747	0.833	0.893

Table 4.8 Experimental results of different complex relation patterns in FB15K.

		Head Prediction				Tail Prediction			
Test	Models	$1-1$	$1-N$	$N-1$	$N-N$	$1-1$	$1-N$	$N-1$	$N-N$
4*Hits@10	ComplEx	0.939	0.969	0.692	0.893	0.938	0.823	0.952	0.910
	RotatE	0.922	0.967	0.602	0.893	0.923	0.713	0.961	0.922
	StructurE	0.919	0.971	0.630	0.897	0.918	0.805	0.964	0.928
	KEMA++	0.930	0.968	0.652	0.896	0.928	0.811	0.966	0.920
4*MRR	ComplEx	0.832	0.914	0.543	0.787	0.826	0.661	0.869	0.800
	RotatE	0.878	0.934	0.465	0.803	0.872	0.611	0.909	0.832
	StructurE	0.762	0.931	0.511	0.810	0.761	0.685	0.899	0.839
	KEMA++	0.820	0.930	0.505	0.804	0.822	0.606	0.898	0.828

Table 4.9 Comparison of knowledge graph embedding models in terms of scoring functions and complexity.

Model	Score Function	Complexity
TransE	|h + r - t|	O(|E|d + |R|d)
Complex	$h^T R\,\bar{t}$	$O(2\|E\|d+2\|R\|d)$
RotatE	-$\|\|h \odot r - t\|\|$	$O(2\|E\|d+2\|R\|d)$
ConvR	$g(W\ g([h] \circledast w_r)+b)^T t$	$O(\|E\|d_e+\|R\|d_r+ Td_r+Td_e(2d_{e_m}-m_w+1) \times(d_{e_n}-n_w+1))$
QuatE	$h \otimes \frac{r}{\|r\|}.t$	$O(2\|E\|d+2\|R\|d)$
KEMA	$min((h-t)\,\%r,(t-h)\,\%r)$	$O(\|E\|d+\|R\|d)$
KEMA++	$min(((h\times r_p)-t)\,\%r_m,(t-(h\times r_p))\,\%r_m)$	$O(\|E\|d+2\|R\|d)$

vectors for each relation. However, such increasing of complexity has allowed KEMA++ to outperform the performance of both TransE and KEMA models.
3) ComplEx, RotatE, and QuatE share the same complexity levels. The main difference between those models is that complEx is an aged model with limited performance unlike RotatE and QuatE that achieve competitive results.
4) ConvR has the highest complexity compared to other models, because it is based on deep learning architecture that have more variable to be tuned.
According to this analysis, we conclude that a model must be chosen according to three factors: dataset, complexity and performance. We strongly believe that KEMA++ is the best model since it can deal with all types of relation while preserving a low complexity.

4.5 Conclusion

Link prediction is among the most prominent methods that solve the problem of incompleteness of Knowledge graph. It is used to predict the existence and the type of a relation connecting two entities. The more the knowledge graph is well represented, the more the predictions are accurate. In this chapter, We proposed a novel knowledge graph embedding model (KEMA++) that relies on both projection and modular space to represent relations between entities. The main strength of our model lies in its ability to represent all simple and complex patterns: symmetric, anti-symmetric, inverse, transitive, composed, $1-N$, $N-1$, and $N-N$ relations. The results of our experiments show that KEMA++ achieves good results compared to the state-of-the-art models in representation accuracy while preserving low level of complexity.

5 Conclusion and Perspectives

This thesis investigates the methods exploiting different features from the KGs for KGC: Link Prediction in KGs including head and tail prediction, triple classification, and Entity Type Prediction. The embeddings of entities and relations represented by low-dimensional vector have proven to be beneficial for KGC as shown in literature review presented in Chapters 3 shows. However, the powerful features of KG are still not being used to their best extent. In this thesis, several embedding-based models leveraging these features are proposed to predict the missing links in the KG. The contributions made in this thesis are summarized in this chapter and the prospective areas for future research.

5.0.1 Conclusions

In this thesis we have tackled several problems related to knowledge graph embedding models.

First, Chapter 1 introduced the concepts of knowledge graph after a brief introduction to graphs. Chapter 1 defines and give examples of some basic concepts of knowledge graph domain such as triplets, relations, and literals. Then it mentions some knowledge graph applications, and shows the graph dependency of some public use applications. Later the chapter discusses the challenges facing Knowledge graphs, before defining the types of knowledge graphs with examples of each type. The chapter ends with the section explaining the general component of a knowledge graph embedding model, and listing the tasks this model can handle.

Chapter 3 introduce TransModE, an embedding model relying on transition in modular space. Before explaining the core of the model, the chapter lists and defines all the patterns of simple and complex relations. Then an overview on the modular arithmetic operation is given, to recall its rules of negation, distancing, and others. In the core of TransModE model, the chapter explains the details of the scoring function this model relies on. Moreover, the chapter mathematically proves the ability of TransModE to represent all simple and complex patterns. Last, the simulation results are shown in tables proving the efficiency of the model in comparison to the models existing in the literature.

Chapter 4 proposes another knowledge graph model called KEMA++. This model is an enhanced version of the model KEMA, knowledge graph embedding model using modular arithmetic. KEMA is a low complexity, high efficiency model. The score function of KEMA takes the advantage of the modular arithmetic operation to guide the head of a relation to its missing tail. KEMA scores a promising results, knowing that it can represent all simple and complex patterns except anti-symmetric. Beside its high efficiency, an important feature of KEMA is it simplicity, reflecting low time complexity. To overcome the shortage of inability

of KEMA to represent anti-symmetric relations, KEMA++ is introduced through upgrading the scoring function. KEMA++ preserves the low time complexity meanwhile it improves the representation ability. Mathematical equations show that all simple and complex patterns are perfectly represented by KEMA++. Lastly, the simulation results are shown in tables proving the efficiency of the model in comparison to the models existing in the literature.

5.0.2 Open issues and future work

This dissertation will encourage different opportunities to pursue open challenges that have not been addressed so far. This section discusses the open issues of this thesis and possible directions for future work.

This thesis focuses on predicting the missing links within a KG. However, link prediction can be more generalized, so it can be performed across multiple KGs to predict the missing links between the two same entities across KGs. This task is known as Entity Alignment, and it relies on predicting links between entities of different KGs that each has its own point of view of a certain domain. Entity Alignment can be achieved by learning the embedding of the entities and the relations of the KGs separately then learning a supervised model to align the entities and the relations. Next the KGs are joined into the unified space. The proposed KG embedding models in this thesis TransModE, KEMA and KEMA++ can be used as a base model to embed the KGs.

Our proposed KGE models focus on learning the embedding vectors of the entities and relations of a knowledge graph depending on the graph itself. However, a vast amount of data sources can be incorporated beside the KG, what may enhance the quality of the data in the KG, and help obtaining more accurate embeddings, and therefore predict more missing links.

Another limitation of the thesis is that the proposed models are evaluated on open source datasets. In future, it would be interesting to test the performance of our proposed link prediction models on enterprise KGs, where the quality of the data is better and the performance of the models for sure will be perfect.

In conclusion, it is anticipated that the contributions from this thesis will lead to rapid advancement in the techniques used for KG-based representation and its applications in different domains.

Publications

1 Published Journal

[1] Hussein Baalbaki, Hussein Hazimeh, Hassan Harb, Rafael Angarita. *KEMA++: A Full Representative Knowledge-Graph Embedding Model (036)*. International Journal of Software Engineering and Knowledge Engineering, Vol. 32, No. 11n12, pages 1619–1641, 2022.

2 Published Conferences

[2] Hussein Baalbaki, Hussein Hazimeh, Hassan Harb, Rafael Angarita. *KEMA: Knowledge-graph embedding using modular arithmetic*. The 34th International Conference on Software Engineering and Knowledge Engineering (KESE 2022), Pittsburgh, USA, July 1-10, 2022. Best paper award (second place).

[3] Hussein Baalbaki, Hussein Hazimeh, Hassan Harb, Rafael Angarita. *TransModE: Translational Knowledge Graph Embedding Using Modular Arithmetic*. Th 26th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES 2022), Verona, Italy, September 7-9, 2022.

Bibliography

[1] R. F. Simmons, *Synthetic language behavior*. System Development Corporation, 1963.

[2] E. W. Schneider, "Course modularization applied: The interface system and its implications for sequence control and data analysis." 1973.

[3] A. Singhal *et al.*, "Introducing the knowledge graph: things, not strings," *Official google blog*, vol. 5, no. 16, p. 3, 2012.

[4] H. Paulheim, "Knowledge graph refinement: A survey of approaches and evaluation methods," *Semantic web*, vol. 8, no. 3, pp. 489–508, 2017.

[5] D. Vrandečić and M. Krötzsch, "Wikidata: a free collaborative knowledgebase," *Communications of the ACM*, vol. 57, no. 10, pp. 78–85, 2014.

[6] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum, "Yago2: A spatially and temporally enhanced knowledge base from wikipedia," *Artificial intelligence*, vol. 194, pp. 28–61, 2013.

[7] N. Nakashole, M. Theobald, and G. Weikum, "Scalable knowledge harvesting with high precision and high recall," in *Proceedings of the fourth ACM international conference on Web search and data mining*, 2011, pp. 227–236.

[8] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "Dbpedia: A nucleus for a web of open data," in *international semantic web conference*. Springer, 2007, pp. 722–735.

[9] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 2008, pp. 1247–1250.

[10] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: A large ontology from wikipedia and wordnet," *Journal of Web Semantics*, vol. 6, no. 3, pp. 203–217, 2008.

[11] Y. Raimond, T. Ferne, M. Smethurst, and G. Adams, "The bbc world service archive prototype," *Journal of web semantics*, vol. 27, pp. 2–9, 2014.

[12] A. Callahan, J. Cruz-Toledo, P. Ansell, and M. Dumontier, "Bio2rdf release 2: improved coverage, interoperability and provenance of life science linked data," in *The Semantic Web: Semantics and Big Data: 10th International Conference, ESWC 2013, Montpellier, France, May 26-30, 2013. Proceedings 10*. Springer, 2013, pp. 200–212.

[13] R. Pittman, A. Srivastava, S. Hewavitharana, A. Kale, and S. Mansour, "Cracking the code on conversational commerce. ebay blog," *Library Catalog: www. ebayinc. com. 6th Apr*, 2017.

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

[15] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[16] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang, "Knowledge vault: A web-scale approach to probabilistic knowledge fusion," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 601–610.

[17] A. Bordes and E. Gabrilovich, "Constructing and mining web-scale knowledge graphs: Kdd 2014 tutorial," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 1967–1967.

[18] H. Paulheim, "Towards profiling knowledge graphs." in *PROFILES@ ISWC*, 2017.

[19] R. West, E. Gabrilovich, K. Murphy, S. Sun, R. Gupta, and D. Lin, "Knowledge base completion via search-based question answering," in *Proceedings of the 23rd international conference on World wide web*, 2014, pp. 515–526.

[20] P. H. Kanani and A. K. McCallum, "Selecting actions for resource-bounded information extraction using reinforcement learning," in *Proceedings of the fifth ACM international conference on Web search and data mining*, 2012, pp. 253–262.

[21] J. Paul Vargheese, P. Travers, J. Pan, K. Vincent, C. Wallace, and A. Kabedeva, "Constructing social media knowledge graphs with social scientists," in *Proceedings of the 30th International BCS Human Computer Interaction Conference 30*, 2016, pp. 1–3.

[22] A. Sadeghi, C. Lange, M.-E. Vidal, and S. Auer, "Integration of scholarly communication metadata using knowledge graphs," in *Research and Advanced Technology for Digital Libraries: 21st International Conference on Theory and Practice of Digital Libraries, TPDL 2017, Thessaloniki, Greece, September 18-21, 2017, Proceedings 21*. Springer, 2017, pp. 328–341.

[23] G. Gawriljuk, A. Harth, C. A. Knoblock, and P. Szekely, "A scalable approach to incrementally building knowledge graphs," in *Research and Advanced Technology for Digital Libraries: 20th International Conference on Theory and Practice of Digital Libraries, TPDL 2016, Hannover, Germany, September 5–9, 2016, Proceedings 20*. Springer, 2016, pp. 188–199.

[24] S. Choudhury, K. Agarwal, S. Purohit, B. Zhang, M. Pirrung, W. Smith, and M. Thomas, "Nous: Construction and querying of dynamic knowledge graphs," in *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*. IEEE, 2017, pp. 1563–1565.

[25] T. Steiner, R. Verborgh, R. Troncy, J. Gabarro, and R. Van de Walle, "Adding realtime coverage to the google knowledge graph," in *11th International Semantic Web Conference (ISWC 2012)*, vol. 914. Citeseer, 2012, pp. 65–68.

[26] P. Zeng, Q. Tan, X. Meng, H. Zhang, and J. Xu, "Modeling complex relationship paths for knowledge graph completion," *ieice Transactions on Information and Systems*, vol. 101, no. 5, pp. 1393–1400, 2018.

[27] B. Shi and T. Weninger, "Open-world knowledge graph completion," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[28] J. Ding, S. Ma, W. Jia, and M. Guo, "Jointly modeling structural and textual representation for knowledge graph completion in zero-shot scenario," in *Web and Big Data: Second International Joint Conference, APWeb-WAIM 2018, Macau, China, July 23-25, 2018, Proceedings, Part I 2*. Springer, 2018, pp. 369–384.

[29] P. Zhao, C. Aggarwal, and G. He, "Link prediction in graph streams," in *2016 IEEE 32nd international conference on data engineering (ICDE)*. IEEE, 2016, pp. 553–564.

[30] I. Chami, A. Wolf, D.-C. Juan, F. Sala, S. Ravi, and C. Ré, "Low-dimensional hyperbolic knowledge graph embeddings," *arXiv preprint arXiv:2005.00545*, 2020.

[31] T. Ebisu and R. Ichise, "Toruse: Knowledge graph embedding on a lie group," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[32] J. Guo and S. Kok, "Bique: Biquaternionic embeddings of knowledge graphs," *arXiv preprint arXiv:2109.14401*, 2021.

[33] H. Xiao, M. Huang, and X. Zhu, "From one point to a manifold: Knowledge graph embedding for precise link prediction," *arXiv preprint arXiv:1512.04792*, 2015.

[34] Z. Cao, Q. Xu, Z. Yang, X. Cao, and Q. Huang, "Dual quaternion knowledge graph embeddings," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 8, 2021, pp. 6894–6902.

[35] I. Balazevic, C. Allen, and T. Hospedales, "Multi-relational poincaré graph embeddings," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[36] S. Liang, "Collaborative, dynamic and diversified user profiling," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 4269–4276.

[37] ——, "Unsupervised semantic generative adversarial networks for expert retrieval," in *The world wide web conference*, 2019, pp. 1039–1050.

[38] M. Nickel and D. Kiela, "Poincaré embeddings for learning hierarchical representations," *Advances in neural information processing systems*, vol. 30, 2017.

[39] Y. Wu, B. Lu, L. Tian, and S. Liang, "Learning to co-embed queries and documents," *Electronics*, vol. 11, no. 22, p. 3694, 2022.

[40] W. H. Greub, *Linear algebra*. Springer Science & Business Media, 2012, vol. 23.

[41] J. H. Wilkinson, F. L. Bauer, and C. Reinsch, *Linear algebra*. Springer, 2013, vol. 2.

[42] H. Lu, H. Hu, and X. Lin, "Dense: An enhanced non-commutative representation for knowledge graph embedding with adaptive semantic hierarchy," *Neurocomputing*, vol. 476, pp. 115–125, 2022.

[43] Y. Chen, J. Liu, Z. Zhang, S. Wen, and W. Xiong, "M\"{o} biuse: Knowledge graph embedding on m\"{o} bius ring," *arXiv preprint arXiv:2101.02352*, 2021.

[44] A. A. Ungar, "Hyperbolic trigonometry and its application in the poincaré ball model of hyperbolic geometry," *Computers & Mathematics with Applications*, vol. 41, no. 1-2, pp. 135–147, 2001.

[45] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," *Advances in neural information processing systems*, vol. 26, pp. 1–9, 2013.

[46] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 28, no. 1, 2014.

[47] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 29, no. 1, 2015.

[48] M. Nickel, V. Tresp, and H.-P. Kriegel, "A three-way model for collective learning on multi-relational data," in *Icml*, 2011.

[49] I. Balažević, C. Allen, and T. M. Hospedales, "Tucker: Tensor factorization for knowledge graph completion," *arXiv preprint arXiv:1901.09590*, 2019.

[50] S. Amin, S. Varanasi, K. A. Dunfield, and G. Neumann, "Lowfer: Low-rank bilinear pooling for link prediction," in *International Conference on Machine Learning*. PMLR, 2020, pp. 257–268.

[51] H. Liu, Y. Wu, and Y. Yang, "Analogical inference for multi-relational embeddings," in *International conference on machine learning*. PMLR, 2017, pp. 2168–2178.

[52] M. Nickel, L. Rosasco, and T. Poggio, "Holographic embeddings of knowledge graphs," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016.

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

[54] S. M. Kazemi and D. Poole, "Simple embedding for link prediction in knowledge graphs," *Advances in neural information processing systems*, vol. 31, 2018.

[55] Q. Zhang, R. Wang, J. Yang, and L. Xue, "Knowledge graph embedding by reflection transformation," *Knowledge-Based Systems*, vol. 238, p. 107861, 2022.

[56] Y. Peng and J. Zhang, "Lineare: Simple but powerful knowledge graph embedding for link prediction," in *2020 IEEE international conference on data mining (ICDM)*. IEEE, 2020, pp. 422–431.

[57] Q. Zhang, R. Wang, J. Yang, and L. Xue, "Knowledge graph embedding by translating in time domain space for link prediction," *Knowledge-Based Systems*, vol. 212, p. 106564, 2021.

[58] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," in *International Conference on Machine Learning*. PMLR, 2016, pp. 2071–2080.

[59] Z. Sun, Z.-H. Deng, J.-Y. Nie, and J. Tang, "Rotate: Knowledge graph embedding by relational rotation in complex space," *arXiv preprint arXiv:1902.10197*, 2019.

[60] L. Gao, H. Zhu, H. H. Zhuo, and J. Xu, "Dual quaternion embeddings for link prediction," *Applied Sciences*, vol. 11, no. 12, p. 5572, 2021.

[61] S. Wang, K. Fu, X. Sun, Z. Zhang, S. Li, and L. Jin, "Hierarchical-aware relation rotational knowledge graph embedding for link prediction," *Neurocomputing*, vol. 458, pp. 259–270, 2021.

[62] X. Ge, Y.-C. Wang, B. Wang, and C. J. Kuo, "Core: A knowledge graph entity type prediction method via complex space regression and embedding," *Pattern Recognition Letters*, vol. 157, pp. 97–103, 2022.

[63] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2d knowledge graph embeddings," in *Thirty-second AAAI conference on artificial intelligence*, 2018.

[64] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15*. Springer, 2018, pp. 593–607.

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

[66] T. Yao, Y. Wang, K. Zhang, and S. Liang, "Improving the expressiveness of k-hop message-passing gnns by injecting contextualized substructure information," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023, pp. 3070–3081.

[67] S. Vashishth, S. Sanyal, V. Nitin, and P. Talukdar, "Composition-based multi-relational graph convolutional networks," *arXiv preprint arXiv:1911.03082*, 2019.

[68] D. Marcheggiani and I. Titov, "Encoding sentences with graph convolutional networks for semantic role labeling," *arXiv preprint arXiv:1703.04826*, 2017.

[69] A. Bordes, J. Weston, R. Collobert, and Y. Bengio, "Learning structured embeddings of knowledge bases," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 25, no. 1, 2011, pp. 301–306.

[70] L. Yao, C. Mao, and Y. Luo, "Kg-bert: Bert for knowledge graph completion," *arXiv preprint arXiv:1909.03193*, 2019.

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

[72] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[73] G. Li, Z. Sun, W. Hu, G. Cheng, and Y. Qu, "Position-aware relational transformer for knowledge graph embedding," *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[74] J. Feng, M. Huang, Y. Yang, and X. Zhu, "Gake: Graph aware knowledge embedding," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 641–651.

[75] T. Safavi, D. Koutra, and E. Meij, "Evaluating the calibration of knowledge graph embeddings for trustworthy link prediction," *arXiv preprint arXiv:2004.01168*, 2020.

[76] P. Tabacof and L. Costabello, "Probability calibration for knowledge graph embedding models," *arXiv preprint arXiv:1912.10000*, 2019.

[77] E. Burnaev and A. Bernstein, "Manifold modeling in machine learning," *Journal of Communications Technology and Electronics*, vol. 66, no. 6, pp. 754–763, 2021.

[78] R. Xie, Z. Liu, J. Jia, H. Luan, and M. Sun, "Representation learning of knowledge graphs with entity descriptions," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, no. 1, 2016.

[79] R. Xie, Z. Liu, M. Sun *et al.*, "Representation learning of knowledge graphs with hierarchical types." in *IJCAI*, vol. 2016, 2016, pp. 2965–2971.

[80] Z. Zhang, F. Zhuang, M. Qu, F. Lin, and Q. He, "Knowledge graph embedding with hierarchical relation structure," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 3198–3207.

[81] L. Guo, Z. Sun, and W. Hu, "Learning to exploit long-term relational dependencies in knowledge graphs," in *International conference on machine learning*. PMLR, 2019, pp. 2505–2514.

[82] Y. Zhang, Q. Yao, and L. Chen, "Interstellar: Searching recurrent architecture for knowledge graph embedding," *Advances in Neural Information Processing Systems*, vol. 33, pp. 10 030–10 040, 2020.

[83] Z. Li, X. Liu, X. Wang, P. Liu, and Y. Shen, "Transo: a knowledge-driven representation learning method with ontology information constraints," *World Wide Web*, vol. 26, no. 1, pp. 297–319, 2023.

[84] T. Lacroix, N. Usunier, and G. Obozinski, "Canonical tensor decomposition for knowledge base completion," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2863–2872.

[85] J. Hao, M. Chen, W. Yu, Y. Sun, and W. Wang, "Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 1709–1719.

[86] Z. Huang, D. Wang, B. Huang, C. Zhang, J. Shang, Y. Liang, Z. Wang, X. Li, C. Faloutsos, Y. Sun *et al.*, "Concept2box: Joint geometric embeddings for learning two-view knowledge graphs," *arXiv preprint arXiv:2307.01933*, 2023.

[87] R. Xie, Z. Liu, H. Luan, and M. Sun, "Image-embodied knowledge representation learning," *arXiv preprint arXiv:1609.07028*, 2016.

[88] N. Bourbaki, *Commutative algebra: chapters 1-7*. Springer Science & Business Media, 1998, vol. 1.

[89] S. Lang, *Algebra*. Springer Science & Business Media, 2012, vol. 211.

[90] J. J. Rotman, "A first course in abstract algebra: with applications," *(No Title)*, 2006.

[91] C. Faith, *Algebra II Ring Theory: Vol. 2: Ring Theory*. Springer Science & Business Media, 2012, vol. 191.

[92] H. Georgi and S. L. Glashow, "Unity of all elementary-particle forces," *Physical Review Letters*, vol. 32, no. 8, p. 438, 1974.

[93] G. d. B. Robinson, "On the representations of the symmetric group," *American Journal of Mathematics*, pp. 745–760, 1938.

[94] C. Xu and R. Li, "Relation embedding with dihedral group in knowledge graph," *arXiv preprint arXiv:1906.00687*, 2019.

[95] T. Yang, L. Sha, and P. Hong, "Nage: Non-abelian group embedding for knowledge graphs," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 1735–1742.

[96] J. Chai and G. Shi, "Module: Module embedding for knowledge graphs," *arXiv preprint arXiv:2203.04702*, 2022.

[97] W. M. Goldman, "varieties of representations," in *Geometry of Group Representations: Proceedings of the AMS-IMS-SIAM Joint Summer Research Conference Held July 5-11, 1987 with Support from the National Science Foundation*, vol. 74. American Mathematical Soc., 1988, p. 169.

[98] R. Fitzpatrick, "Euclid's elements of geometry," 2008.

[99] Y. Tang, J. Huang, G. Wang, X. He, and B. Zhou, "Orthogonal relation transforms with graph context modeling for knowledge graph embedding," *arXiv preprint arXiv:1911.04910*, 2019.

[100] C. Gao, C. Sun, L. Shan, L. Lin, and M. Wang, "Rotate3d: Representing relations as rotations in three-dimensional space for knowledge graph embedding," in *Proceedings of the 29th ACM international conference on information & knowledge management*, 2020, pp. 385–394.

[101] X. Huang, J. Tang, Z. Tan, W. Zeng, J. Wang, and X. Zhao, "Knowledge graph embedding by relational and entity rotation," *Knowledge-Based Systems*, vol. 229, p. 107310, 2021.

[102] T. Le, N. Huynh, and B. Le, "Knowledge graph embedding by projection and rotation on hyperplanes for link prediction," *Applied Intelligence*, vol. 53, no. 9, pp. 10 340–10 364, 2023.

[103] T. Le, H. Tran, and B. Le, "Knowledge graph embedding with the special orthogonal group in quaternion space for link prediction," *Knowledge-Based Systems*, vol. 266, p. 110400, 2023.

[104] L. Chao, J. He, T. Wang, and W. Chu, "Pairre: Knowledge graph embeddings via paired relation vectors," *arXiv preprint arXiv:2011.03798*, 2020.

[105] L. Yu, Z. Luo, H. Liu, D. Lin, H. Li, and Y. Deng, "Triplere: Knowledge graph embeddings via tripled relation vectors," *arXiv preprint arXiv:2209.08271*, 2022.

[106] B. Wang, Q. Meng, Z. Wang, H. Zhao, D. Wu, W. Che, S. Wang, Z. Chen, and C. Liu, "Interht: Knowledge graph embeddings by interaction between head and tail entities," *arXiv preprint arXiv:2202.04897*, 2022.

[107] R. Li, J. Zhao, C. Li, D. He, Y. Wang, Y. Liu, H. Sun, S. Wang, W. Deng, Y. Shen *et al.*, "House: Knowledge graph embedding with householder parameterization," in *International Conference on Machine Learning*. PMLR, 2022, pp. 13 209–13 224.

[108] A. S. Householder, "Unitary triangularization of a nonsymmetric matrix," *Journal of the ACM (JACM)*, vol. 5, no. 4, pp. 339–342, 1958.

[109] X. Zhang, Q. Yang, and D. Xu, "Trans: Transition-based knowledge graph embedding with synthetic relation representation," *arXiv preprint arXiv:2204.08401*, 2022.

[110] X. Ge, Y.-C. Wang, B. Wang, and C.-C. J. Kuo, "Compounde: Knowledge graph embedding with translation, rotation and scaling compound operations," *arXiv preprint arXiv:2207.05324*, 2022.

[111] ——, "Knowledge graph embedding with 3d compound geometric transformations," *arXiv preprint arXiv:2304.00378*, 2023.

[112] T. Ma, M. Li, S. Lv, F. Zhu, L. Huang, and S. Hu, "Conte: contextualized knowledge graph embedding for circular relations," *Data Mining and Knowledge Discovery*, vol. 37, no. 1, pp. 110–135, 2023.

[113] S. Wang, R. Liu, L. Shen, and A. M. Khattak, "Stke: Temporal knowledge graph embedding in the spherical coordinate system," in *International Conference on Artificial Intelligence and Security*. Springer, 2022, pp. 292–305.

[114] F. Sala, C. De Sa, A. Gu, and C. Ré, "Representation tradeoffs for hyperbolic embeddings," in *International conference on machine learning*. PMLR, 2018, pp. 4460–4469.

[115] Z. Pan and P. Wang, "Hyperbolic hierarchy-aware knowledge graph embedding for link prediction," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, 2021, pp. 2941–2948.

[116] Z. Sun, M. Chen, W. Hu, C. Wang, J. Dai, and W. Zhang, "Knowledge association with hyperbolic knowledge graph embeddings," *arXiv preprint arXiv:2010.02162*, 2020.

[117] K. Wang, Y. Liu, D. Lin, and Q. Z. Sheng, "Hyperbolic geometry is not necessary: Lightweight euclidean-based models for low-dimensional knowledge graph embeddings," *arXiv preprint arXiv:2103.14930*, 2021.

[118] W. Zheng, W. Wang, F. Qian, S. Zhao, and Y. Zhang, "Hyperbolic hierarchical knowledge graph embeddings for link prediction in low dimensions," *arXiv preprint arXiv:2204.13704*, 2022.

[119] S. Wang, X. Wei, C. N. Nogueira dos Santos, Z. Wang, R. Nallapati, A. Arnold, B. Xiang, P. S. Yu, and I. F. Cruz, "Mixed-curvature multi-relational graph neural network for knowledge graph completion," in *Proceedings of the Web Conference 2021*, 2021, pp. 1761–1771.

[120] B. Xiong, S. Zhu, M. Nayyeri, C. Xu, S. Pan, C. Zhou, and S. Staab, "Ultrahyperbolic knowledge graph embeddings," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 2130–2139.

[121] C. Gregucci, M. Nayyeri, D. Hernández, and S. Staab, "Link prediction with attention applied on multiple knowledge graph embedding models," in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 2600–2610.

[122] H. Xiao, X. Liu, Y. Song, G. Y. Wong, and S. See, "Complex hyperbolic knowledge graph embeddings with fast fourier transform," *arXiv preprint arXiv:2211.03635*, 2022.

[123] J. W. Harris and H. Stöcker, *Handbook of mathematics and computational science*. Springer Science & Business Media, 1998.

[124] A. Suzuki, Y. Enokida, and K. Yamanishi, "Riemannian transe: Multi-relational graph embedding in non-euclidean space," 2018.

[125] X. Lv, L. Hou, J. Li, and Z. Liu, "Differentiating concepts and instances for knowledge graph embedding," *arXiv preprint arXiv:1811.04588*, 2018.

[126] Y. Dong, X. Guo, J. Xiang, K. Liu, and Z. Tang, "Hypersphere: An embedding method for knowledge graph completion based on hypersphere," in *International Conference on Knowledge Science, Engineering and Management*. Springer, 2021, pp. 517–528.

[127] S. He, K. Liu, G. Ji, and J. Zhao, "Learning to represent knowledge graphs with gaussian embedding," in *Proceedings of the 24th ACM international on conference on information and knowledge management*, 2015, pp. 623–632.

[128] H. Xiao, M. Huang, Y. Hao, and X. Zhu, "Transg: A generative mixture model for knowledge graph embedding," *arXiv preprint arXiv:1509.05488*, 2015.

[129] G. Wan and B. Du, "Gaussianpath: A bayesian multi-hop reasoning framework for knowledge graph reasoning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 5, 2021, pp. 4393–4401.

[130] S. Liao, S. Liang, Z. Meng, and Q. Zhang, "Learning dynamic embeddings for temporal knowledge graphs," in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 2021, pp. 535–543.

[131] F. Wang, Z. Zhang, L. Sun, J. Ye, and Y. Yan, "Dirie: knowledge graph embedding with dirichlet distribution," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 3082–3091.

[132] M. Nayyeri, B. Xiong, M. Mohammadi, M. M. Akter, M. M. Alam, J. Lehmann, and S. Staab, "Knowledge graph embeddings using neural itô process: From multiple walks to stochastic trajectories," in *Findings of the Association for Computational Linguistics: ACL 2023*, 2023, pp. 7165–7179.

[133] M. Xu, "Understanding graph embedding methods and their applications," *SIAM Review*, vol. 63, no. 4, pp. 825–853, 2021.

[134] M. Nayyeri, C. Xu, F. Hoffmann, M. M. Alam, J. Lehmann, and S. Vahdati, "Knowledge graph representation learning using ordinary differential equations," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 9529–9548.

[135] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," *Advances in neural information processing systems*, vol. 31, 2018.

[136] Z. Han, Z. Ding, Y. Ma, Y. Gu, and V. Tresp, "Temporal knowledge graph forecasting with neural ode," *arXiv preprint arXiv:2101.05151*, 2021.

[137] I. Kobyzev, S. J. Prince, and M. A. Brubaker, "Normalizing flows: An introduction and review of current methods," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 11, pp. 3964–3979, 2020.

[138] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: a core of semantic knowledge," in *Proceedings of the 16th international conference on World Wide Web*, 2007, pp. 697–706.

[139] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, "Collaborative knowledge base embedding for recommender systems," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 353–362.

[140] Y. Hao, Y. Zhang, K. Liu, S. He, Z. Liu, H. Wu, and J. Zhao, "An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 221–231.

[141] B. Yang and T. Mitchell, "Leveraging knowledge bases in lstms for improving machine reading," *arXiv preprint arXiv:1902.09091*, 2019.

[142] L. Costabello, S. Pai, C. Van, R. McGrath, N. McCarthy, and P. Tabacof, "Ampligraph: a library for representation learning on knowledge graphs," *Retrieved October*, vol. 10, p. 2019, 2019.

[143] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[144] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[145] Z. Zhang, J. Cai, Y. Zhang, and J. Wang, "Learning hierarchy-aware knowledge graph embeddings for link prediction," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 03, 2020, pp. 3065–3072.

[146] K. Toutanova and D. Chen, "Observed versus latent features for knowledge base and text inference," in *Proceedings of the 3rd workshop on continuous vector space models and their compositionality*, 2015, pp. 57–66.

[147] Q. Zhang, R. Wang, J. Yang, and L. Xue, "Structural context-based knowledge graph embedding for link prediction," *Neurocomputing*, vol. 470, pp. 109–120, 2022.

[148] S. Zhang, Y. Tay, L. Yao, and Q. Liu, "Quaternion knowledge graph embeddings," *Advances in neural information processing systems*, vol. 32, 2019.

[149] J. Burstein, C. Doran, and T. Solorio, "Proceedings of the 2019 conference of the north american chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019.

[150] H. Baalbaki, H. Hazimeh, H. Harb, and R. Angarita, "Kema: Knowledge-graph embedding using modular arithmetic," in *The 34th International Conference on Software Engineering and Knowledge Engineering*, 2022.

[151] A. Rossi, D. Barbosa, D. Firmani, A. Matinata, and P. Merialdo, "Knowledge graph embedding for link prediction: A comparative analysis," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 15, no. 2, pp. 1–49, 2021.