Hierarchy Knowledge Graph for Parameter-Efficient Entity Embedding

Hepeng Gao, **Funing Yang***, **Yongjian Yang** and **Ying Wang**
Jilin University
gaohepeng13@hotmail.com, {yfn, yyj, wangying2010}@jlu.edu.cn

Abstract

Traditional knowledge graphs (KGs) provide each entity with a unique embedding as a representation, which contains a lot of redundant information. Meanwhile, the space complexities of the KGs are positively related to the number of entities. In this work, we propose a hierarchical representation learning method, namely HRL, which is a parameter-efficient model where the number of model parameters is independent of dataset scales. Specifically, we propose a hierarchical model comprising a **Meta Encoder** and a **Context Encoder** to generate the representation of entities and relations. The Meta Encoder captures the common representations shared across entities, while the Context Encoder learns entity-specific representations. We further provide a theoretical analysis of model design by constructing a **structural causal model** (SCM) when completing a knowledge graph. The SCM outlines the relationships between nodes, where entity embeddings are conditioned on both common and entity-specific representations. Note that our model is designed to reduce model scale while maintaining competitive performance. We evaluate HRL on the knowledge graph completion task using three real-world datasets. The results demonstrate that HRL significantly outperforms existing parameter-efficient baselines, as well as traditional state-of-the-art baselines of similar scale.

1 Introduction

Knowledge Graphs (KGs) have been widely used across various domains, including recommendation systems [Yang *et al.*, 2023; Du *et al.*, 2022; Zhang *et al.*, 2023], interpretability analysis [Ma *et al.*, 2023; Bing *et al.*, 2023], and zero-shot learning [Liu *et al.*, 2020; Wang *et al.*, 2018], among others. KGs, as a form of structured human knowledge, are composed of triples, i.e., (*head entity*, *relation*, *tail entity*), or (*h*, *r*, *t*) for short, where entities represent objects or abstract concepts, and relations denote the relationships between entities.

*Corresponding Author.

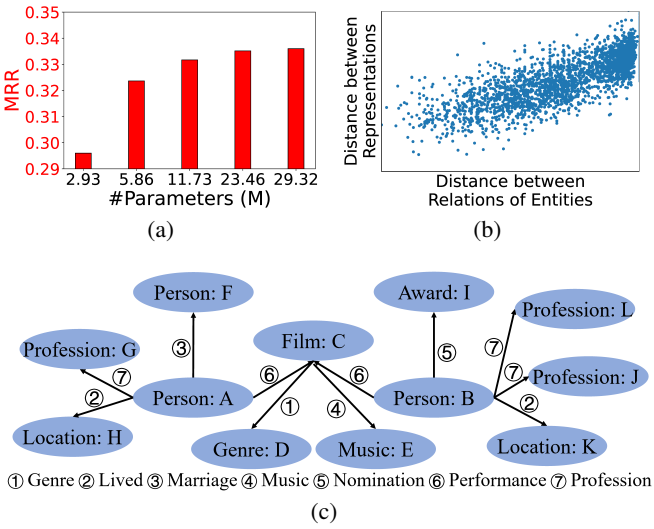


Figure 1: (a) RotatE Performance. (b) Correlation of Entity Representation Distance and Entities' Relations on the FB15k-237. The distance between representations denotes the Euclidean distance between RotatE's entity representations. The distance between relations of entities is calculated as the Euclidean distance between relation vectors. Each vector position is binary, with a value of 1 indicating that the entity interacts with the relation type corresponding to the position at least once. (c) Illustration of a knowledge graph. It shows blue ellipses as entities and arrows as relations.

In traditional KGs, each entity corresponds to a unique embedding that is independently trained. It results in the following issues: 1) The number of model parameters is correlated with the dataset scale, leading to significant hardware resource consumption during training and deployment. The space complexity of these methods is theoretically $O(|E| + |R|)$ [Galkin *et al.*, 2021] where $|E|$ is the number of entities, and $|R|$ is the number of relations. Since the number of entities is typically much larger than the number of relations (e.g., YAGO3-10: 123k entities vs. 37 relations and OGBL-Wikikg2: 2,500k entities vs. 535 relations) in real-world KGs, $|E|$ is the dominant factor in space complexity. Our analysis focused on $|E|$, following prior work (e.g., EARL [Chen *et al.*, 2023]). 2) Since entity embeddings are not shared between entities, the only way to limit model

scales in traditional methods is by reducing the embedding dimensions. In traditional methods, reducing embedding dimensions too much can lead to excessive compression, which results in information loss and degraded model performance. As shown in Figure 1(a), the performance of RotatE [Sun *et al.*, 2018], a state-of-the-art knowledge graph model, decreases dramatically when entity embedding dimensions are reduced. In summary, simply reducing the embedding dimensions is not an effective manner to limiting model scale.

Several works [Galkin *et al.*, 2021; Chen *et al.*, 2023] have been developed to address the above issues. These works primarily focus on reducing the number of entity embeddings to limit space complexity. These works first randomly sample a fixed proportion of entities to serve as anchors, each with a corresponding unique embedding. Specifically, NodePiece [Galkin *et al.*, 2021] generates entity embeddings by utilizing the nearest k anchors, while EARL generates embeddings by incorporating relation types and passing messages from these anchors. However, the methods still fall short due to the following challenges: 1) **Space Complexity**. Despite the significant reduction in model scales, the space complexity of existing methods remains $O(|E| + |R|)$. This is due to selecting a fixed proportion of entities as anchors, though the proportion is low. 2) **Sampling Strategy**. The random sampling of entities in existing methods adversely affects model performance and stability. The uneven positional distribution of sampled anchors leads to suboptimal entity representations.

To address the aforementioned challenges, we propose HRL, a hierarchical representation learning method, which assumes entity embeddings are affected by two factors: common representations shared across entities and entity-specific representations distinguishing individual entities. Inspired by human cognitive processes, HRL begins with capturing the common characteristics of entities within a category and then delves into the individual features of each specific entity. For example, as illustrated in Figure 1(c), entities such as *Person A* and *Person B* belong to the same category and share common representations relevant to their category, which constrains their relation types, whereas *Film C* does not. It is logical for entities in the *Person* category to have relations such as *lived* or *profession*, while relations like *genre* or *music* are illogical for the *Person* category. Meanwhile, it is not possible to distinguish among entities belonging to the same category merely by common representations. It necessitates incorporating entity-specific representations. For example, (*Person: A*, *nomination*, *Award: I*) and (*Person: B*, *nomination*, *Award: I*) cannot be distinguished by common representations, as both *A* and *B* fall into the *Person* category. Distinguishing between them requires knowing the structural information relevant to *A* and *B* in the KG. *B*'s *profession J* and *Award I* belong to the same domain according to (*Person: B*, *Profession*, *Profession: J*), making the latter more likely to be the positive instance.

Our framework incorporates a Meta Encoder to capture common representations using a memory mechanism, and a Context Encoder to learn entity-specific representations from structural information. This combination allows us to efficiently manage model parameters while maintaining robust entity differentiation. Then, we integrate common and entity-specific representations as entity embeddings and follow the traditional KG training regime to optimize the parameters. This design model eliminates the need to assign a unique embedding to each entity, thereby reducing the model parameters while obviating the need for anchor selection and precomputation. Finally, we illustrate the causalities about the entity embedding generation and knowledge graph complete task with SCM. The contributions of this paper are summarized as follows:

- We propose a HRL that contains a Meta Encoder and a Context Encoder that can capture common and entity-specific representations of entities. Significantly, the number of parameters in the model is independent of the dataset scales, achieving a space complexity of $O(|R|)$.
- We introduce a novel causal perspective for analyzing the knowledge graph completion process, offering a theoretical expression for our model design. To our knowledge, this is the first successful application of structural causal models to guide entity embedding generation.
- We conduct extensive experiments to evaluate our model on three real-world graph datasets. Our model maintains competitive performance compared with state-of-the-art parameter-efficient models on knowledge graph completion tasks while utilizing fewer parameters.

2 Related Work

2.1 Causal Learning

Causal learning is widely used in numerous scenarios. Traditional methods often focus on correlation rather than underlying causality. KGCR [Wei *et al.*, 2022] enhances recommendation systems by learning fine-grained user preferences using causal KGs. CAL [Sui *et al.*, 2022] introduces a causal perspective to graph neural networks, improving feature exploitation while avoiding shortcuts. STNSCM [Deng *et al.*, 2023] uses a causal graph for traffic prediction, analyzing relationships among factors to improve performance by extrapolating from factual to counterfactual scenarios. This paper analyzes the entity embedding generation from the perspective of causality, offering a theoretical explanation.

2.2 Parameter-Efficient Model

Current models mainly pursue the enhancement of performance with increasing model size, which complicates training and deployment. To address this, methods like quantization and knowledge distillation are commonly used. Quantization reduces the precision of weights and activations to minimize accuracy loss, as seen in TS-CL [Sachan, 2020] and LightKG [Wang *et al.*, 2021a]. Knowledge distillation transfers knowledge from larger models to smaller ones, as exemplified by MulDE [Wang *et al.*, 2021b] and DualDE [Zhu *et al.*, 2022]. These methods require training large models first. Recently, there has been the emergence of a new perspective, which involves directly reducing model scales. NodePiece [Galkin *et al.*, 2021] selects a subset of entities to serve as anchors that are assigned unique embeddings. The embeddings for the other entities are then generated based on these anchors and the relations that belong to the other entities. The

number of anchors is $10 - 100\times$ smaller than the total number of entities. EARL [Chen *et al.*, 2023] constructs entity embeddings using connected relations, k-nearest anchor entities (similar to NodePiece), and multi-hop neighbors. Our model also falls into the above category and is compared with NodePiece and EARL as the same type.

3 Preliminaries

A knowledge graph consists of an entity set E, a relation set R, and a triple set T. $|\cdot|$ denotes the number of elements of the set. Further, a knowledge graph is also denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X}, \mathcal{R})$, where $\mathcal{V} = \{v_i \mid 0 \leq i < |E|\}$ is a node set corresponding to entities and $\mathcal{E} = \{e_{ij} \mid 0 \leq i, j < |E|\}$ denotes the set of directed edges based on the relation set R. $\mathcal{X} = \{x_i \in \mathbb{R}^d \mid 0 \leq i < |E|\}$ and $\mathcal{R} = \{r_{ij} \in \mathbb{R}^{d'} \mid 0 \leq i, j < |E|\}$ represent the entity and relation embeddings, respectively. e_{ij} represents a directed edge from v_i to v_j, and its embedding is r_{ij}.

Problem Statement. Given an entity set E, a relation set R, and a triple set T, the goal is to complete the missing triples.

4 Methodology

4.1 Meta Encoder

The Meta Encoder significantly reduces the need for unique embeddings by generalizing across entities within the same category. Specifically, the Meta Encoder learns the patterns of entities by interactions between entities and relations. As shown in Figure 1(b), entity representations are correlated with the interacted relation types, which means that entities sharing the same relation types can be grouped together. This fashion leads to more compact and efficient representations, making the model require fewer parameters.

The Meta Encoder leverages memory mechanisms and consists of a memory bank and a query network. The memory bank, serving as a representation space, stores representations for various entity categories. The query network, using relation embeddings as input to effectively retrieve, generates queries to access and retrieve relevant entity embeddings from the memory bank.

As shown in Figure 2(b), the memory bank $M \in \mathbb{R}^{\eta \times d}$ is a learnable matrix that is randomly initialized. We generate key vectors K and query vectors Q_i to compute the relevance weights of each embedding vector in the memory bank for the entity i. The K and Q_i are expressed as:

$$K = f_K(M), \tag{1}$$

$$Q_i = f_Q(m_i), \tag{2}$$

where f_K and f_Q are stacking nonlinear layers, $K, Q_i \in \mathbb{R}^{1\times\eta}$. $m_i \in \mathbb{R}^d$ is an entity embedding based on relation types where i is the index of entities.

To generate entity-related queries, we design a hypergraph structure, where entities and relations are represented as nodes and hyperedges, respectively. Entities sharing the same relation type are connected to the same hyperedge. The relation type embeddings are learnable parameters. The embeddings of entities are progressively refined through a message-passing process over the hypergraph. The message passing process is formulated as:

$$\bar{x}_i^{t+1} = \rho_v(\{\bar{e}_k^t \mid k \in \mathcal{N}_i^v\}), \tag{3}$$

$$\bar{e}_k^{t+1} = \rho_e(\{\bar{x}_j^t \mid j \in \mathcal{N}_k^e\}), \tag{4}$$

where $\mathcal{N}_i^v$ is a relation set connected by entity i and $\mathcal{N}_k^e$ is an entity set belonging to relation k. $\bar{x}$ and $\bar{e}$ are the entity and relation embeddings, respectively. ρ_v and ρ_e are aggregation functions that are set as pooling. After multiple iterations of message passing, the final entity embedding are used as m_i. This design ensures an efficient and expressive mechanism for capturing relational and structural information within the knowledge graph.

Next, we calculate the weight of each embedding vector in the memory bank and generate common representations, which are formulated as:

$$E_i^C = softmax(Q_i + K) \otimes M, \tag{5}$$

where $E_i^C \in \mathbb{R}^{1\times d}$, $\otimes$ is the matrix product, and $softmax(Q_i + K)$ is a weight that measures the relatedness of the entity v_i to the embedding vectors in M. The $softmax$ normalizes the weights, assigning higher values to more relevant vectors. This results in more distinctive common representations for entities from different categories, enhancing the model's ability to differentiate.

4.2 Context Encoder

The Meta Encoder generates common representations that can capture the characteristics of entities in one category. However, while the common representations capture common characteristics, they lack the specificity needed to differentiate between entities within the same category. As shown in Figure 1(c), common representations can effectively identify incorrect instances like (*Person: A, genre, Genre: D*), but distinguishing entities of the same category, such as (*Person: A, nomination, Award: I*) and (*Person: B, nomination, Award: I*), remains challenging due to the overlap in their common representations. To that end, we propose a Context Encoder that learns entity-specific representations, which are essential for distinguishing between entities that share common representations.

The Context Encoder learns entity-specific representations by leveraging graph structural information, which helps differentiate entities based on their unique topological relationships within the KG. For entities within the same category, their topological structure provides distinguishing information. The *Award I* and *Profession J* are in the same domain, from which it can be inferred that (*Person: B, nomination, Award: I*) could be a positive instance, rather than (*Person: A, nomination, Award: I*). Since KGs are heterogeneous graphs, the Context Encoder aggregates both entity and relation embeddings, which is expressed as:

$$h_i^{t+1} = \phi(h_i^t, u_i^{t+1}, v_i^{t+1}), \tag{6}$$

$$u_i^{t+1} = \psi_u(\{h_k^t || r_{ik}^t \mid k \in \mathcal{N}_{i\cdot}\}), \tag{7}$$

$$v_i^{t+1} = \psi_v(\{h_k^t || r_{ki}^t \mid k \in \mathcal{N}_{\cdot i}\}), \tag{8}$$

$$r_{ik}^t = \varphi_u(r_{ik}^{t-1}), \tag{9}$$

$$r_{ki}^t = \varphi_v(r_{ik}^{t-1}), \tag{10}$$

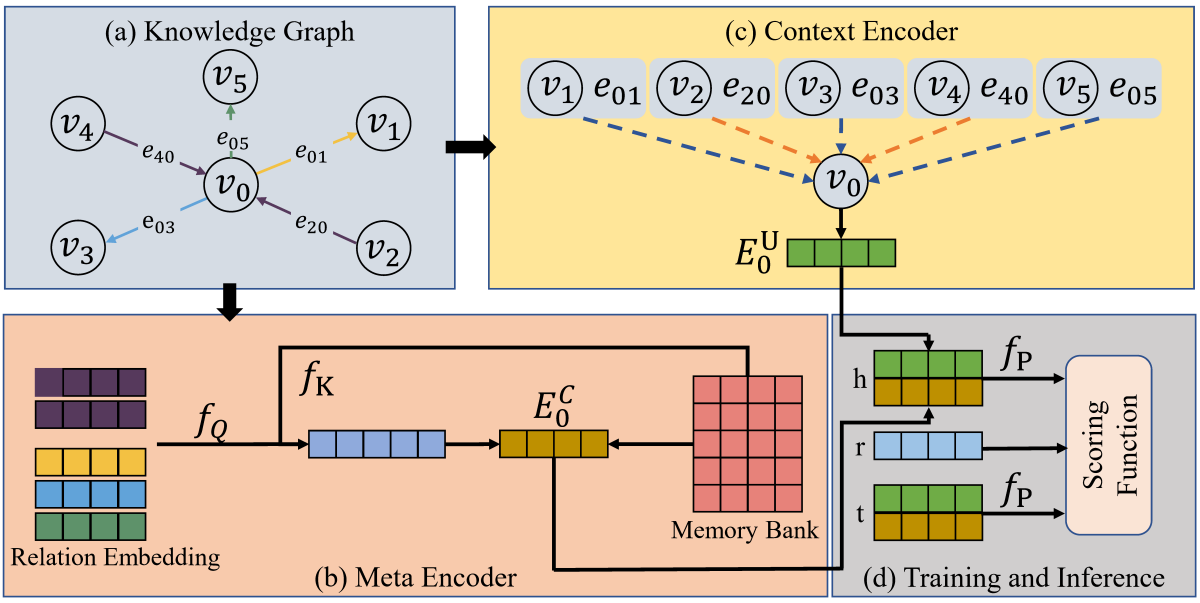


Figure 2: An overall architecture of HRL. (a) The colors of arrows indicate relation types. (b) The memory bank is a fixed-size matrix. (c) The direction and colors of dotted arrows indicate message passing and relation direction in the KG, respectively. (d) The scoring function is to measure the plausibility of (h, r, t).

where ϕ is an update function that updates each entity embedding by aggregating its current embedding h_i^t with its neighboring embeddings, u_i^{t+1} and v_i^{t+1}. ψ_u and ψ_v are message functions which are nonlinear layers. $\mathcal{N}_{i\cdot}$ and $\mathcal{N}_{\cdot i}$ are the sets of tail and head entities connected to the entity v_i, respectively. r_{ik}^t and r_{ki}^t are the relation embeddings of (v_i, v_k) and (v_k, v_i). $||$ is the concatenation operation. As shown in Figure 2(c), the entity v_0 integrates tail entities, i.e., (v_1, e_{01}), (v_3, e_{03}), and (v_5, e_{05}), and head entities, i.e., (v_2, e_{20}), and (v_4, e_{40}). φ_u and φ_v are nonlinear layers to update relation embeddings r_{ik}^t and r_{ki}^t. h_i^0 is initialized with the common embedding E_i^C and r^0 is a relation embedding r. The output is the entity-specific representation $E_i^U \in \mathbb{R}^{1\times d}$.

In our model, we aggregate information about neighboring entities and relations multiple times to enlarge receptive fields, which can enhance the distinguishability of entity-specific representations. However, frequent message passing can lead to over-smoothing embeddings, where the embeddings of adjacent nodes become too similar to distinguish between them. Therefore, the number of stacking layers is carefully chosen to balance between capturing sufficient context and avoiding over-smoothing.

4.3 Training and Inference

In our model, parameters are trained in the same manner as traditional models. We design a projection layer to integrate the common and entity-specific representations into entity embeddings, which is expressed as:

$$x_i = f_P(E_i^C || E_i^U), \tag{11}$$

where f_P denotes a projection layer that is a linear layer and x_i is the embedding of the entity v_i.

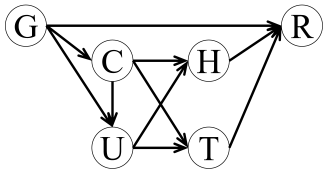


Figure 3: SCM. It shows a structure causal graph for KG completion where G, C, U, H, T, and R are KG information, common representation, entity-specific representation, head entity embedding, tail entity embedding, and relation embedding, respectively.

The HRL can adaptively integrate the scoring function and training method used in existing KGs. In HRL, we adopt the widely-used RotatE's scoring function and the self-adversarial negative sampling strategy for training.

4.4 Theoretical Analysis of Model Design

We first analyze the causal relationships among knowledge graph information, entity embeddings, and relation embeddings using a SCM, as depicted in Figure 3. In real-world scenarios, the complete knowledge graph is unavailable, therefore, the KG can be divided into two parts: a training set that is the observed part of the KG and a test set that is the unobserved part of the KG. We assume entity embeddings H and T consist of both common (C) and entity-specific (U) representations which are derived from the knowledge graph (G). R is affected by H, T, and G when it is inferred.

We need to model the C and U effects on R for parameter-efficient embedding and robustness. However, since knowledge graphs are complex, observed relations are directly used to infer unobserved relationships, which introduce noise. In Figure 1(c), we believe J and I belong to the same domain,

but L and I are uncorrelated. When (*Person: B, profession, Profession: J*) and (*Person: B, profession, Profession: L*) are all or partially observable, it is possible that the inference of (*Person: B, nomination, Award: I*) is negatively affected. Therefore, we quantify the impacts of the common and entity-specific representations on KG completion.

We cannot apply backdoor adjustments based on the path $\mathsf{C},\mathsf{U} \leftarrow \mathsf{G} \rightarrow \mathsf{R}$ because G is only partially observable, leading to incomplete adjustments for confounding effects. Additionally, we apply the frontdoor criterion according to the path $\mathsf{G} \rightarrow \mathsf{C},\mathsf{U} \rightarrow \mathsf{H},\mathsf{T} \rightarrow \mathsf{R} \leftarrow \mathsf{G}$, cutting off the link $\mathsf{G} \rightarrow \mathsf{C},\mathsf{U}$. Formally, we formulate:

$$
\begin{aligned}
&P(\mathsf{R}|do(\mathsf{C},\mathsf{U})) \\
=&\sum_{\mathsf{H},\mathsf{T}} P(\mathsf{R}|do(\mathsf{H},\mathsf{T}))P(\mathsf{H},\mathsf{T}|do(\mathsf{C},\mathsf{U})) \\
=&\sum_{\mathsf{H},\mathsf{T}} \sum_{\mathsf{C}',\mathsf{U}'} P(\mathsf{R}|\mathsf{H},\mathsf{T},\mathsf{C}',\mathsf{U}')P(\mathsf{C}',\mathsf{U}')P(\mathsf{H},\mathsf{T}|\mathsf{C},\mathsf{U}) \\
=&\sum_{\mathsf{H},\mathsf{T}} \sum_{\mathsf{C}',\mathsf{U}'} P(\mathsf{R}|\mathsf{H},\mathsf{T},\mathsf{C}',\mathsf{U}')P(\mathsf{U}'|\mathsf{C}')P(\mathsf{C}')P(\mathsf{H},\mathsf{T}|\mathsf{C},\mathsf{U}),
\end{aligned}
\tag{12}
$$

where $P(\mathsf{R}|do(\mathsf{C},\mathsf{U}))$ provides a causal perspective on how specific interventions in the C and U influence the R within KGs, thereby offering valuable insights into the underlying causal relationships. It denotes the probability distribution of R resulting from such interventions. $do(\mathsf{C},\mathsf{U})$ indicates an intervention in the causal model where C and U are fixed to specific values. Additionally, $P(\mathsf{C}')$ represents the distribution of common representations and $P(\mathsf{U}'|\mathsf{C}')$ represents the distribution of entity-specific representations given common representations. $P(\mathsf{H},\mathsf{T}|\mathsf{C},\mathsf{U})$ is a projection layer that generates entity embedding H and T based on common and entity-specific representations. $P(\mathsf{R}|\mathsf{H},\mathsf{T},\mathsf{C}',\mathsf{U}')$ denotes the distribution of the relation between the head entity and the tail entity, i.e., the feature extraction and the scoring function.

4.5 Space Complexity Analysis

We theoretically analyze the space complexity of our model and compare it with baseline models. We assume the number of entities and relations are $|E|$ and $|R|$, respectively. Since the number of entities $|E|$ is typically much larger than the number of relations $|R|$, the space complexity due to $|R|$ is often negligible in comparison to $|E|$.

In most traditional methods, the space complexity is dominated by $|E|$ entity embeddings, leading to a space complexity of $O(|E|)$. While $|R|$ relation embeddings contribute to the overall complexity, $|E|$ is typically much larger, so the $O(|R|)$ term in $O(|E| + |R|)$ is ignored and the complexity is considered to be actually $O(|E|)$. This means the number of parameters increases linearly as the growth of the dataset scales. To reduce the number of parameters, existing methods often use entity sampling techniques. They sample a proportion p of entities as anchors and generate embeddings for the remaining entities based on these anchors, reducing the number of unique entity embeddings required. However, the resulting space complexity still includes $p|E|$ for entity embeddings, $|R|$ for relation embeddings, and additional parameters for other components. The methods relieve the issue, but the space complexities are still $O(|E|)$.

In our model, parameters include $|R|$ relation embeddings, a Meta Encoder, a Context encoder, and a projection layer. The parameters of the Meta Encoder include a memory bank with a fixed size, a query network utilizing a pooling layer, and several linear layers. Since the scale of these components does not depend on the number of entities $|E|$, their combined space complexity is $O(1)$. The parameters of the Context Encoder include several stacking nonlinear layers. Given that the number of these layers is fixed and does not scale with $|E|$, their space complexity is $O(1)$. The projection layer also consists of linear layers, similarly, its space complexity is $O(1)$. In conclusion, the space complexity of HRL is $O(1)$ when $|R|$ is ignored, indicating that the number of parameters remains constant and does not increase with dataset scales.

5 Experiments

In this section, we perform KG completion tasks to train and evaluate our model. Note that the core goal of our model is to **achieve parameter efficiency, reducing the model scale while maintaining reasonable performance, rather than solely aiming to surpass traditional methods with high space complexity**. In summary, we demonstrate the following questions: **RQ1**. Can our model achieve reasonable performance with fewer parameters? **RQ2**. What is the influence of various components in the HRL on different datasets? **RQ3**. Can the Meta Encoder effectively capture the common representations as our assumption? **RQ4**. What is the impact of different settings on HRL?

5.1 Datasets

We evaluate our model on three real-world knowledge graph datasets: FB15k-237 [Toutanova *et al.*, 2015], WN18RR [Dettmers *et al.*, 2018], and CoDEx-L [Safavi and Koutra, 2020]. The datasets are well-established KGs commonly used in the field. Detailed statistics and descriptions of these datasets are provided in Table 2.

5.2 Baselines

We compare HRL with two categories of models: 1) Traditional Models. We select RotatE without constraint on the number of parameters, which serves as an upper bound of performance. For a fair comparison, we adjust the embedding dimension of RotatE and compGCN [Vashishth *et al.*, 2020] to approximately match the number of other baselines. 2) Entity-efficient Models. We compare with NodePiece and EARL, which use similar scoring as RotatE.

5.3 Experimental Settings

Implementation Details

For the Meta Encoder, the memory bank dimension is searched within the range $\{128, 256, 512\}$. For the Context Encoder, the number of layers is set to 2 for FB15k-237 and CoDEx-L, and 3 for WN18RR. We employ the Adam [Diederik, 2014] optimizer with a learning rate of $1e^{-3}$ and a weight decay of $5e^{-5}$. The batch size and the number of negative samples are set to 1024 and 256, respectively. Due

	Model	RotatE	RotatE	compGCN	NodePiece	EARL	HRL	△	w/o CM.	w/o CT.	w/o PJ.
FB15k-237	Dim	1000	100	100	100	150	180	-	180	180	180
	#P(M)	29.3	2.9	9.4	3.2	1.8	0.9	-50.0%	0.8	0.5	0.7
	MRR	0.356	0.293	0.345	0.258	0.301	0.306	1.7%	0.296	0.217	0.297
	Effi	0.012	0.101	0.037	0.081	0.167	0.340	103.3%	0.370	0.434	0.424
WN18RR	Dim	500	50	100	100	200	240	-	240	240	240
	#P(M)	40.6	4.1	12.0	4.4	3.8	1.9	-50.0%	1.7	0.8	1.5
	MRR	0.514	0.414	0.494	0.402	0.425	0.433	1.9%	0.407	0.243	0.410
	Effi	0.013	0.101	0.041	0.091	0.112	0.228	103.8%	0.239	0.304	0.273
CoDEx-L	Dim	500	25	100	100	100	190	-	190	190	190
	#P(M)	78	3.8	15.8	3.6	2.1	1.1	-47.6%	0.9	0.5	0.8
	MRR	0.262	0.197	0.286	0.193	0.232	0.235	1.3%	0.194	0.162	0.202
	Effi	0.003	0.052	0.018	0.054	0.110	0.214	93.4%	0.216	0.324	0.253

Table 1: Knowledge graph completion results of HRL and baseline models.

Dataset	#Ent	#Rel	#Train	#Valid	#Test
FB15k-237	14,505	237	272,115	17,526	20,438
WN18RR	40,559	11	86,835	2,824	2,924
CoDEx-L	77,951	69	551,193	30,622	30,622

Table 2: The statistics of datasets.

to the RotatE's training strategy, the margin γ is chosen from $\{8, 9, 10, 11, 12\}$. The temperature factor α is set to 1.

Metrics

We utilize a widely-used metric, MRR in the filtered setting [Bordes *et al.*, 2013] on KG completion tasks to evaluate our model. In addition, we report the number of parameters #P to compare our model scale with baselines. Meanwhile, we use Effi [Chen *et al.*, 2023] to quantify the efficiency of models, which is calculated as $\frac{\text{MRR}}{\#\text{P}}$.

5.4 Performance Results (RQ1)

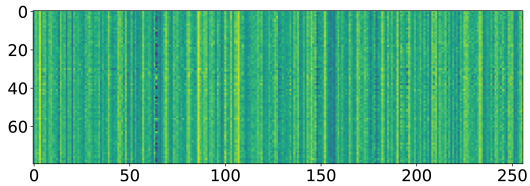
To demonstrate the effectiveness of HRL, we compare it with baselines on three benchmark datasets. The performances are summarized in Table 1. We set different entity dimensions for the three datasets and report the HRL and the baseline methods performances.

By analyzing the results, we have the following findings: 1) In the case where RotatE's parameter scale is unconstrained, it achieves optimal performance in MRR. However, its parameter scale significantly exceeds that of parameter-efficient models and HRL, and its Effi is substantially lower compared to other models. Additionally, when reducing the parameter scale of RotatE to be comparable to other models, its performance in MRR and Effi falls short compared to baseline methods. 2) NodePiece and EARL significantly reduce model scale by selecting a subset of entities as anchors and assigning unique embeddings only to these entities, rather than to all entities. Meanwhile, they leverage structural information within the knowledge graph to improve performance on MRR. In terms of Effi, these parameter-efficient models outperform traditional models. 3) HRL demonstrates a significant reduction in model scale compared to other methods, with decreases ranging from 47.6% to 50%. Despite this substantial decrease in model scale, HRL outperforms the baselines in performance metrics, with improving ranging from 1.3% to 1.7% in MRR across the different datasets. HRL demonstrates superior performance in terms of Effi compared to baseline methods, indicating its ability to better balance model performance with space complexity. This balance underscores the ability to optimize computational resources without sacrificing performance.

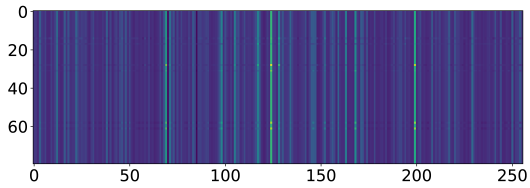
5.5 Ablation Study (RQ2)

This subsection describes ablation studies of HRL to validate the effectiveness of key components, as shown in Table 1. We consider the following variants of our base model: 1) w/o CM.: We remove the Meta Encoder. In this variant, the query network's input is used as the common representations to initialize the entity-specific representations and the entity-specific representations are treated as the entity embedding. 2) w/o CT.: We remove the Context Encoder, and the common representations are considered as the entity embedding to optimize parameters. 3) w/o PJ.: We remove the projection layer and concatenate the common and entity-specific representations as entity embeddings.

The performances of the competing baselines and HRL are summarized on the datasets shown in Table 1. From this table, we can have the following findings: 1) In terms of the number of parameters, the Context Encoder is more than the Meta Encoder and the projection layer. As the dimension of the entity embeddings becomes wider, the number of parameters in the Context Encoder grows the fastest. 2) The performance of w/o CT. on WN18RR drops dramatically compared to its performance on FB15k-237. We believe the number of relation types results in the phenomenon. Due to $P(U'|C')$, the Q is not a good prior distribution for a Context Encoder on WN18RR. 3) The projection layer is crucial for enhancing entity representations. A simple concatenation of common and entity-specific representations cannot capture the distribution of $P(H, T|C, U)$ effectively. 4) While the partial variants improve the efficiency (Effi), they lead to decreases in MRR and compromise the model's robustness. This suggests that these variants' performance is more sensitive to the characteristics

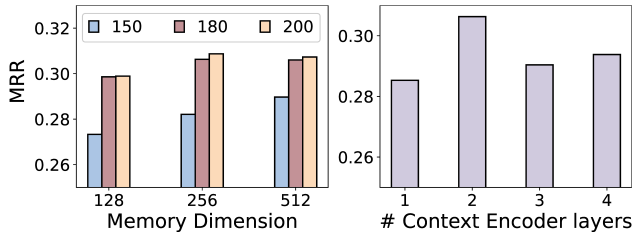


(a) Embedding Weight Heat-map of FB15k-237

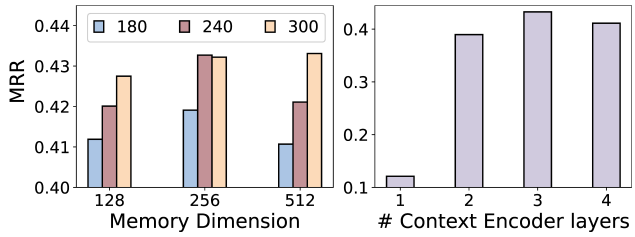


(b) Embedding Weight Heat-map of WN18RR

Figure 4: In (a) and (b), the horizontal axis indicates the output of the query network $softmax(K + Q)$, which ranges from 0 to 256 (memory bank dimension). The vertical axis indicates the entity ID. The larger the value, the brighter the color of the corresponding position, and conversely, the duller the color.



(a) FB15k-237



(b) WN18RR

Figure 5: The left figures in (a) and (b) shown HRL performances under various memory bank and embedding dimensions. The colors of the bars denote embedding dimensions. The right figures in (a) and (b) shown HRL performances under various numbers of Context Encoder layers.

of datasets and the components are necessary for the HRL.

5.6 Case Study (RQ3)

We further present a case study to verify our assumption that the Meta Encoder effectively captures common representations. We randomly sample 80 entities from one category to analyze their embedding weights, computed as $softmax(K + Q)$ for the memory bank. As shown in Figure 4, the embedding weights of entities within the same category exhibit a similar distribution. Therefore, the Meta Encoder can capture the common representation of entities in one category. Additionally, the results of the w/o CT. demonstrate that common representations for different categories can be effectively distinguished.

5.7 Parameter Study (RQ4)

Impact of memory bank dimension

As shown on the left in Figure 5(a) and (b), the memory bank dimension affects the HRL performance. In general, with the increase in memory bank dimension, the performance of HRL starts to increase at the beginning, then plateaus or declines. Initially, a larger memory bank dimension increases the embedding space, which enhances the model's learning capacity. However, excessively large dimensions result in overfitting, diminishing the model's generalization ability and leading to performance degradation.

Impact of the number of Context Encoder layers

We also conduct experiments on different numbers of Context Encoder layers. The experiment results are displayed on the right in Figure 5(a) and (b). Initially, MRR improves as the number of Context Encoder layers increases. The increased layers enlarge receptive fields, making entity-specific representations to obtain rich structural information. It enhances the distinguishability of the entity-specific representations, as well as model performances. Then, the MRR declines with the number of Context Encoder layers increasing. Layer stacking makes message passing frequent, which results in the representations being too similar to distinguish. However, as the number of layers continues to increase, frequent message passing can make over-smoothing embeddings, leading to diminished performance.

6 Conclusion

In this paper, we propose a hierarchical representation learning method, a parameter-efficient model designed to reduce the number of parameters while maintaining competitive performance. HRL leverages a combination of the Meta Encoder and the Context Encoder to capture both common and entity-specific representations. HRL achieves a significant reduction in model scale compared to existing models, while still delivering robust performance across three benchmark datasets. A critical aspect of HRL involves the use of SCM to analyze the causal relationships between entity embeddings, relation embeddings, and knowledge graph information. SCM provides a theoretical analysis for understanding how common and entity-specific representations influence relations within KGs. Finally, we conduct numerous experiments to demonstrate the effectiveness of our model.

Acknowledgments

This work was supported by National Natural Science Foundation of China (No. 62072209) and the Science and Technology Development Program of Jilin Province (No. 20240302093GX).

References

[Bing *et al.*, 2023] Qingyu Bing, Qiannan Zhu, and Zhicheng Dou. Cognition-aware knowledge graph reasoning for explainable recommendation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pages 402–410, 2023.

[Bordes *et al.*, 2013] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26, 2013.

[Chen *et al.*, 2023] Mingyang Chen, Wen Zhang, Zhen Yao, Yushan Zhu, Yang Gao, Jeff Z. Pan, and Huajun Chen. Entity-agnostic representation learning for parameter-efficient knowledge graph embedding. In *Thirty-Seventh AAAI Conference on Artificial Intelligence*, 2023.

[Deng *et al.*, 2023] Pan Deng, Yu Zhao, Junting Liu, Xiaofeng Jia, and Mulan Wang. Spatio-temporal neural structural causal models for bike flow prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 4242–4249, 2023.

[Dettmers *et al.*, 2018] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[Diederik, 2014] P Kingma Diederik. Adam: A method for stochastic optimization. 2014.

[Du *et al.*, 2022] Yuntao Du, Xinjun Zhu, Lu Chen, Ziquan Fang, and Yunjun Gao. Metakg: Meta-learning on knowledge graph for cold-start recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[Galkin *et al.*, 2021] Mikhail Galkin, Etienne Denis, Jiapeng Wu, and William L Hamilton. Nodepiece: Compositional and parameter-efficient representations of large knowledge graphs. In *International Conference on Learning Representations*, 2021.

[Liu *et al.*, 2020] Lu Liu, Tianyi Zhou, Guodong Long, Jing Jiang, and Chengqi Zhang. Attribute propagation network for graph zero-shot learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 4868–4875, 2020.

[Ma *et al.*, 2023] Ting Ma, Longtao Huang, Qianqian Lu, and Songlin Hu. Kr-gcn: Knowledge-aware reasoning with graph convolution network for explainable recommendation. *ACM Transactions on Information Systems*, 41(1):1–27, 2023.

[Sachan, 2020] Mrinmaya Sachan. Knowledge graph embedding compression. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2681–2691, 2020.

[Safavi and Koutra, 2020] Tara Safavi and Danai Koutra. Codex: A comprehensive knowledge graph completion benchmark. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8328–8350, 2020.

[Sui *et al.*, 2022] Yongduo Sui, Xiang Wang, Jiancan Wu, Min Lin, Xiangnan He, and Tat-Seng Chua. Causal attention for interpretable and generalizable graph classification. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1696–1705, 2022.

[Sun *et al.*, 2018] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*, 2018.

[Toutanova *et al.*, 2015] Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1499–1509, 2015.

[Vashishth *et al.*, 2020] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. Composition-based multi-relational graph convolutional networks. In *International Conference on Learning Representations*, 2020.

[Wang *et al.*, 2018] Xiaolong Wang, Yufei Ye, and Abhinav Gupta. Zero-shot recognition via semantic embeddings and knowledge graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6857–6866, 2018.

[Wang *et al.*, 2021a] Haoyu Wang, Yaqing Wang, Defu Lian, and Jing Gao. A lightweight knowledge graph embedding framework for efficient inference and storage. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 1909–1918, 2021.

[Wang *et al.*, 2021b] Kai Wang, Yu Liu, Qian Ma, and Quan Z Sheng. Mulde: Multi-teacher knowledge distillation for low-dimensional knowledge graph embeddings. In *Proceedings of the Web Conference 2021*, pages 1716–1726, 2021.

[Wei *et al.*, 2022] Yinwei Wei, Xiang Wang, Liqiang Nie, Shaoyu Li, Dingxian Wang, and Tat-Seng Chua. Causal inference for knowledge graph based recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[Yang *et al.*, 2023] Yuhao Yang, Chao Huang, Lianghao Xia, and Chunzhen Huang. Knowledge graph self-supervised rationalization for recommendation. In *Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining*, pages 3046–3056, 2023.

[Zhang *et al.*, 2023] Xiaoyu Zhang, Xin Xin, Dongdong Li, Wenxuan Liu, Pengjie Ren, Zhumin Chen, Jun Ma, and Zhaochun Ren. Variational reasoning over incomplete knowledge graphs for conversational recommendation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pages 231–239, 2023.

[Zhu *et al.*, 2022] Yushan Zhu, Wen Zhang, Mingyang Chen, Hui Chen, Xu Cheng, Wei Zhang, and Huajun Chen. Dualde: Dually distilling knowledge graph embedding for

faster and cheaper reasoning. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 1516–1524, 2022.