Should We Use a Fixed Embedding Size? Customized Dimension Sizes for Knowledge Graph Embedding

Zhanpeng Guan[1,2], **Zhao Zhang**[1,2*], **Yiqing Wu**[1,2], **Fuwei Zhang**[3], **and Yongjun Xu**[1,2]
[1]Institute of Computing Technology, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences
[3]Institute of Artificial Intelligence, Beihang University
{guanzhanpeng22s, zhangzhao2021, wuyiqing20s, xyj}@ict.ac.cn
zhangfuwei@buaa.edu.cn

Abstract

Knowledge Graph Embedding (KGE) aims to project entities and relations into a low-dimensional space, so as to enable Knowledge Graphs (KGs) to be effectively used by downstream AI tasks. Most existing KGs (e.g. Wikidata) suffer from the data imbalance issue, i.e., the occurrence frequencies vary significantly among different entities. Current KGE models use a fixed embedding size, leading to overfitting for low-frequency entities and underfitting for high-frequency ones. A simple method is to manually set embedding sizes based on frequency, but this is not feasible due to the complexity and the large number of entities. To this end, we propose CustomizE, which customizes embedding sizes in a data-driven way, assigning larger sizes for high-frequency entities and smaller sizes for low-frequency ones. We use bilevel optimization for stable learning of representations and sizes. It is noteworthy that our framework is universal and flexible, which is suitable for various KGE models. Experiments on link prediction tasks show its superiority over state-of-the-art baselines.

1 Introduction

Knowledge Graphs (KGs) like Freebase (Bollacker et al., 2008), Yago (Suchanek et al., 2007), and Wikidata (Vrandečić and Krötzsch, 2014) are critical in AI-related applications, such as recommender systems (Guo et al., 2020; Xu et al., 2024), information retrieval (Su et al., 2022; Zhang et al., 2022a), and question answering (Ren et al., 2021; Jia et al., 2021). A fact in KGs is a triple (s, r, o), where s and o are entities, and r is the relation, e.g., $(\mathit{London}, \mathit{capital\_Of}, \mathit{UK})$. KGE models encode entities and relations in a low-dimensional space, which is crucial for knowledge completion, fusion, and inference. Given a input triple (s, r, o), KGE models output the representations of s, r, o, and a score for the triple's plausibility (Kazemi and Poole, 2018).

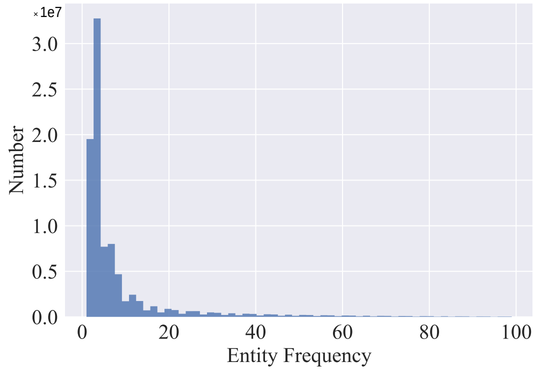


Figure 1: Entity Frequency Histogram of Wikidata.

However, real-world KGs suffer from the data imbalance issue, where various entities showcase significant differences in their occurrence frequencies. Statistics of a real-world KG Wikidata (Vrandečić and Krötzsch, 2014) is shown in Fig 1. The horizontal axis corresponds to the frequencies (number of occurrences) of entities, and the vertical axis represents the number of entities with a certain frequency. Only a small number of entities occur frequently, while most entities occur infrequently, highlighting an imbalance in real-world KGs. Typically, entities outnumber relations, with a more pronounced imbalance. In this paper, we focus on addressing the data imbalance of entities.

Existing KGE models use a fixed embedding size, leading to overfitting for low-frequency entities and underfitting for high-frequency ones. This raises the question: should we use a fixed embedding size? Related works in recommender systems (Zhao et al., 2021; Qu et al., 2022) and computer vision (Wan et al., 2020; Chavan et al., 2022) show the benefits of varying dimension sizes, mainly for reducing memory usage. GreenKGC (Wang et al., 2023) and HolmE (Zheng et al., 2024) focus on maintaining the performance using a unified low-dimensional embedding size

*Corresponding author: Zhao Zhang.

Proceedings of the 31st International Conference on Computational Linguistics, pages 9006–9012
January 19–24, 2025.

for large-scale KGs. In this paper, we focus on enhancing expressive capacity by adjusting sizes based on frequency.

To this end, we propose CustomizE, a novel KGE model that aims to assign smaller embedding sizes to infrequent entities, while customizing larger sizes to frequent ones. Specifically, we design a dimension customization framework, which consists of an embedding module, a dimension selection module, a dimension alignment module, and an application module. Inspired by DARTS (Liu et al., 2018), we use a bilevel optimization algorithm to update parameters, ensuring stable convergence (Zhaok et al., 2021). Unlike neural architecture search, which seeks a unified embedding size, our method customizes sizes for each entity.

To summarize, we highlight our key contributions as follows:

- In this paper, we propose a novel model CustomizE, which customizes different embedding sizes to various entities to address the data imbalance issue in KGE.
- The technique of CustomizE is general and flexible, which is applicable to numerous existing KGE models.
- We validate the effectiveness of CustomizE over state-of-the-art KGE models on benchmark datasets.

2 Preliminaries

In this section, we provide some basic definitions used in this paper.

Definition 1. ***Frequent/Infrequent Entities.*** *In a KG, the entities with top 20% frequencies are named as frequent/high-frequency entities, while the remaining 80% entities are infrequent/low-frequency entities.*

Definition 2. ***Frequent/Infrequent Triples.*** *For a triple, if both s and o are frequent entities, it is termed a frequent triple. Conversely, if both s and o are infrequent entities, it is labeled as an infrequent triple.*

3 Methodology

In this section, we first give an overview of the dimension customization framework. Subsequently, we introduce each part of the proposed framework and provide the training details for the entire framework. Finally, we apply them to KGE models and propose CustomizE.

3.1 Overview

Figure 2 illustrates the dimension customization framework, comprising four modules: embedding, dimension selection, dimension alignment, and application module. The embedding module contains multiple lookup tables with varying embedding sizes. For an entity e, it maps the entity to an embedding $\mathbf{e}^{d_i} \in \mathbb{R}^{d_i}$ from the i-th table $\mathbf{E}^{d_i} \in \mathbb{R}^{n \times d_i}$, where n is the number of entities and d_i is the dimension size. Given N lookup tables $\{\mathbf{E}^{d_1}, ..., \mathbf{E}^{d_N}\}$, we obtain a set of embeddings $\{\mathbf{e}^{d_1}, ..., \mathbf{e}^{d_N}\}$ with various dimensions. The subsequent subsections detail other modules.

3.2 Dimension Selection Module

3.2.1 Input and Output

As noted in Section 1, embedding sizes correlate with entity frequencies. **Input:** Frequency buckets, each representing a specific range. The bucket embedding serves as the input. **Output:** A one-hot vector $\hat{\mathbf{a}} \in \mathbb{R}^N$ indicating the selected dimension size, where N is the number of candidate embedding sizes.

3.2.2 Relaxation

We use a multilayer perceptron (MLP) to capture entity frequency information. To maintain differentiability, we use temperature softmax (Hinton et al., 2015) instead of standard softmax to approximate the dimension selection probability $\mathbf{a} \in \mathbb{R}^N$ to a discrete vector.

$$\mathbf{a}^i = \frac{\exp\left(\mathbf{h}^i/\tau\right)}{\sum_{k=1}^{N}\exp\left(\mathbf{h}^k/\tau\right)}, \quad i \in \{1, ..., N\}, \quad (1)$$

$\mathbf{a}^i$ is the i-th entry of $\mathbf{a}$. $\mathbf{h}$ is the MLP output logits. τ is the temperature hyperparameter, as $\tau \to 0$, the output approaches a one-hot vector. To bridge the gap between training (approximate) and inference (exact one-hot), we apply Straight-Through Estimator (STE) (Bengio et al., 2013) to $\mathbf{a}$. The final output is defined as:

$$\hat{\mathbf{a}} = \mathbf{a} + \text{stop\_gradient}(\text{setmax}(\mathbf{a}) - \mathbf{a}), \quad (2)$$

stop_gradient$(\cdot)$ prevents gradient back propagation. setmax$(\cdot)$ sets the maximum entry to 1 and others to 0. STE ensures $\hat{\mathbf{a}} = \text{setmax}(\mathbf{a})$ while maintaining differentiability (Bengio et al., 2013).

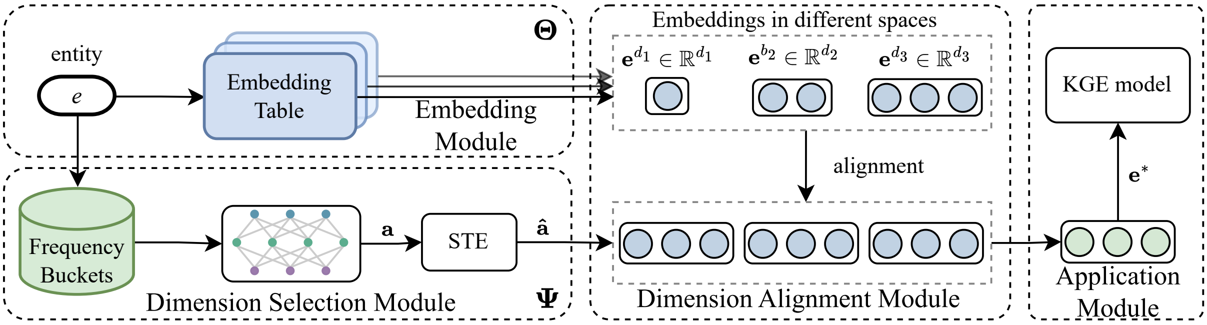


Figure 2: An overview of Dimension Customization Framework.

3.3 Dimension Alignment Module

After obtaining entity embeddings of different sizes, we need to align the embeddings because vectors with different dimensions cannot be directly applied to existing KGE models. To unify the embeddings, we present an alignment method that transforms embeddings with different sizes to the same size.

$$\hat{\mathbf{e}}^{d_i} = \text{LayerNorm}\left(\mathbf{W}_i \mathbf{e}^{d_i} + \mathbf{b}_i\right), \quad i \in \{1, ..., N\}. \tag{3}$$

$\mathbf{W}_i \in \mathbb{R}^{d_N \times d_i}$ and $\mathbf{b}_i \in \mathbb{R}^{d_N}$ represent the i-th weight matrix and bias vector. $\text{LayerNorm}(\cdot)$ is the layer normalization, which aims to make the network converge to appropriate weights faster. Finally, embeddings with different sizes are aligned to the same size.

3.4 Bilevel Optimization

Previous studies (Ren et al., 2018; Borsos et al., 2020) indicate that simultaneously learning embedding sizes and data point representations can lead to instability. Inspired by DARTS (Liu et al., 2018), we propose a bilevel optimization algorithm for alternate updates. We define Ψ as the dimension selection module parameter and Θ as the parameter for other modules. Specifically, we give the general form of bilevel optimization:

$$\min_{\Psi} \mathcal{L}_{\text{outer}}\left(\arg\min_{\Theta}\left(\mathcal{L}_{\text{inner}}\left(\Theta, \Psi^*\right)\right), \Psi\right). \tag{4}$$

Moreover, we employ an approximation scheme:

$$\begin{aligned} &\nabla_{\Psi} \mathcal{L}_{\text{outer}}\left(\Theta^*(\Psi), \Psi\right) \\ \approx &\nabla_{\Psi} \mathcal{L}_{\text{outer}}\left(\Theta - \delta \nabla_{\Theta} \mathcal{L}_{\text{inner}}\left(\Theta, \Psi\right), \Psi\right), \end{aligned} \tag{5}$$

δ is the step size for the dimension selection module parameters. Parameters with superscript $*$ indicate optimal values. The scheme approximates $\Theta^*(\Psi)$ through incremental updates to Θ, avoiding complete optimization of $\Theta^*(\Psi) = \arg\min_{\Theta} \mathcal{L}_{\text{inner}}\left(\Theta, \Psi^*\right)$.

3.5 Application to KGE models

The preceding subsections provide details of each module and the optimization algorithm. Importantly, the dimension customization framework is general and flexible, making it applicable to a variety of KGE models. It is worth mentioning that we empirically verify the flexibility of our framework in Appendix 4.3.3. We apply our framework to ComplEx (Trouillon et al., 2016), proposing CustomizE. ComplEx maps entities and relations to complex space. For a triple (s, r, o), the score function is:

$$\text{score}(s, r, o) = \text{Re}\left(< \mathbf{e}_s, \mathbf{v}_r, \bar{\mathbf{e}}_o >\right), \tag{6}$$

where $\mathbf{e}_s$, $\mathbf{v}_r$, $\mathbf{e}_o$ are representations of s, r, o. $\bar{\mathbf{e}}_o$ is $\mathbf{e}_o$'s conjugate. $\text{Re}(\cdot)$ is the real part. $< \cdot, \cdot, \cdot >$ is the inner product. The loss function is:

$$\begin{aligned} \min_{\Theta_{kge}} \sum_{(s,r,o)} \log\left(1 + \exp\left(-\mathbf{Y}_{sro} \cdot \text{score}(s, r, o)\right)\right) \\ + \gamma \|\Theta_{kge}\|_2^2, \end{aligned} \tag{7}$$

where $\Theta_{kge} \subseteq \Theta$ are embeddings of s, r, o, $\mathbf{Y}_{sro} \in \{0, 1\}$ indicates triple truth, $\gamma\|\Theta_{kge}\|_2^2$ is regularization. Substituting Eq. (7) into Eq. (4) yields specific inner and outer losses for bilevel optimization. CustomizE's training procedure at each iteration:

- Update Ψ by descending $\nabla_{\Psi} \mathcal{L}_{\text{outer}}\left(\Theta^*, \Psi\right)$ with approximation in Eq. (5) on S_O.
- Update Θ by descending $\mathcal{L}_{\text{inner}}\left(\Theta, \Psi^*\right)$ on S_I.

S_I and S_O are splits of the training set S_T ($S_I \cup S_O = S_T$), used for inner and outer loop training respectively.

4 Experiment

We answer the following research questions. **RQ 1**: Does CustomizE perform better than other state-of-the-art KGE models? **RQ 2**: How does CustomizE learn the dimension sizes for entities with different frequencies in KG? **RQ 3**: Does the dimension customization framework work on other KGE models?

4.1 Datasets, Metrics and Baselines.

We evaluate CustomizE with two benchmark datasets: FB15k-237 (Toutanova and Chen, 2015) and WN18RR (Dettmers et al., 2018). The above datasets are widely used benchmarks in KGE, and they both exhibit imbalanced data distributions.

Dataset	Entity	Relation	Train	Valid	Test
FB15k-237	14,541	237	272,115	17,535	20,466
WN18RR	40,943	11	86,835	3,034	3,134

Table 1: Dataset Statistics.

Following prior work (Zhang et al., 2021), we conduct experiments on the link prediction task, also known as the knowledge graph completion task. We compare CustomizE with other methods using two metrics: (i) Mean Reciprocal Rank (MRR, the mean of the reciprocals of predicted ranks); (ii) Hits@k (H@k, the proportion of ranks not larger than k). Results are reported under the "filtered" setting (Bordes et al., 2013).

In this paper, we compare the proposed method with the following baselines. We categorize them into five groups. **Distance-based models:** TransE (Bordes et al., 2013), RotatE (Sun et al., 2019), MuRP (Balazevic et al., 2019) and MuRE (Balazevic et al., 2019). **Tensor decomposition models:** DistMult (Yang et al., 2015), ComplEx (Trouillon et al., 2016), QutaE (Zhang et al., 2019) and BoxE (Abboud et al., 2020). **Neural network models:** ConvE (Dettmers et al., 2018), HypER (Balažević et al., 2019), rules-LNN (Sen et al., 2022) and M-DCN (Zhang et al., 2022b). **Data-imbalance-aware methods:** LSU (Zhang et al., 2021) and Mixup-ComplEx (Xie and Ge, 2024), which are two models that address the data imbalance issue with latent semantic units and data mixup methods, respectively. For the ablation study, we constructed three **variant of CustomizE:** (1) CustomizE-rule, a variant that allocates dimension sizes based on the frequency of entities. Specifically, higher frequencies are allocated larger sizes. (2) CustomizE-sim, a variant that abandons the bilevel training procedure. For CustomizE-sim, we train the embedding sizes and representations of entities simultaneously. (3) CustomizE-iter, a variant that alternately trains the representations and embedding sizes without the sophisticated gradient update method in bilevel optimization.

4.2 Experimental Details

During training, before starting each epoch, we randomly split 80% of the training set as the inner set S_I, and 20% as the outer set S_O. We use Adagrad as the optimizer to update all parameters. We set the learning rate and batch size to 0.15 and 512. We set the layer number of MLP to 2 with ReLU as the activation function. For FB15k-237 and WN18RR, we set the dimension candidate sets as $\{64, 128, 256, 512, 768\}$ and $\{32, 64, 128, 256, 512\}$, respectively. For temperature softmax, we set $\tau = \max(0.01, e^{-0.0003 \cdot t})$, where t is the training step.

4.3 Experimental Results

4.3.1 Main results (RQ1)

	FB15k-237			**WN18RR**		
	MRR	H@1	H@3	MRR	H@1	H@3
TransE†	0.294	-	-	0.226	-	-
RotatE‡	0.338	0.241	0.375	0.476	0.428	0.492
MuRP§	0.335	0.243	0.367	<u>0.481</u>	<u>0.440</u>	0.495
MuRE§	0.336	0.245	0.370	0.465	0.436	0.487
DistMult‡	0.241	0.155	0.263	0.430	0.390	0.440
ComplEx‡	0.247	0.158	0.275	0.440	0.410	0.460
QuatE§	0.311	0.221	0.342	<u>0.481</u>	0.436	<u>0.500</u>
BoxE§	0.337	-	-	0.451	-	-
ConvE‡	0.325	0.237	0.356	0.430	0.400	0.440
HypER§	0.341	0.252	-	0.465	0.436	-
rules-LNN§	0.307	-	0.342	0.473	-	0.497
M-DCN§	<u>0.345</u>	<u>0.255</u>	<u>0.380</u>	0.475	<u>0.440</u>	0.485
LSU$^{\diamondsuit}$	0.336	0.251	0.364	0.475	0.402	0.468
Mixup-ComplEx§	0.279	-	-	0.401	-	-
CustomizE-rule	0.322	0.236	0.353	0.448	0.425	0.458
CustomizE-sim	0.326	0.249	0.356	0.471	0.433	0.485
CustomizE-iter	0.341	0.251	0.370	0.472	0.436	0.488
CustomizE	**0.351***	**0.261***	**0.385***	**0.486***	**0.446***	**0.504***

Table 2: Evaluation results on FB15k-237 and WN18RR datasets. Superscripts †, ‡, ♯, and § indicate the results are taken from (Zhang et al., 2020), (Wang et al., 2021), (Rossi et al., 2021), and the original paper, respectively. $*$ denotes the improvement of CustomizE is statistically significant compared with the best baseline at p-value < 0.05 over paired t-test.

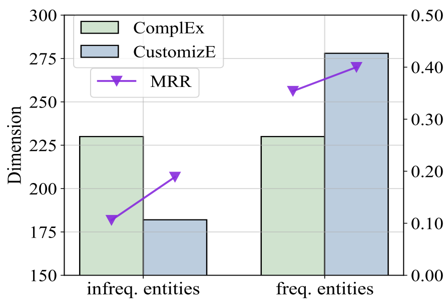
Comparing CustomizE with baselines (RQ1), Table 2 shows: (1) CustomizE outperforms all baselines, demonstrating its effectiveness. (2) CustomizE outperforms its base model ComplEx significantly, demonstrating the effectiveness of the dimension customization framework. (3) CustomizE-rule, CustomizE-sim, and CustomizE-iter under-perform CustomizE, highlighting the importance of our dimension selection module and bilevel optimization algorithm.

Furthermore, we find the proposed dimension customization framework can be successfully applied to various KGE methods. Due to space limitations, this analysis is presented in Appendix 4.3.3.

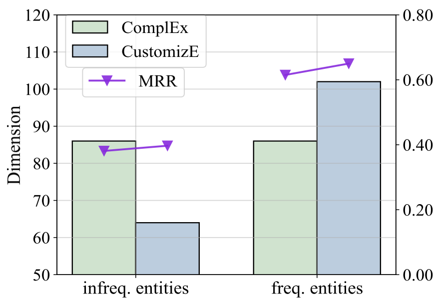
4.3.2 Dimensionality Analysis (RQ2)

Figure 3 shows average dimensions for infrequent and frequent entities (bars) and MRR scores for corresponding triples (line with triangles). To ensure a fair comparison, we set the dimension of ComplEx to the mean of that of CustomizE.

We find CustomizE effectively assigns smaller dimension sizes to infrequent entities and larger dimension sizes for frequent ones, which both result in enhanced performance on infrequent and frequent triples. The above results comprehensively demonstrate that CustomizE mitigates the overfitting phenomenon of infrequent entities and the underfitting phenomenon of frequent entities.



(a) FB15k-237



(b) WN18RR

Figure 3: Dimension size analysis.

4.3.3 Flexibility Analysis(RQ3)

We apply the **Dim**ension **C**ustomization framework (DimC) to other popular KGE models. Due to space limitation, we only report results on FB15k-237 in Table 3. The results provide comprehensive evidence for the effectiveness and flexibility of the framework.

	FB15k-237			
	MRR	H@1	H@3	H@10
TransE	0.294	-	-	0.465
TransE + DimC	**0.324**	**0.232**	**0.358**	**0.509**
DistMult	0.241	0.155	0.263	0.419
DistMult + DimC	**0.346**	**0.255**	**0.380**	**0.528**
ConvE	0.325	0.237	0.356	0.501
ConvE + DimC	**0.331**	**0.238**	**0.365**	**0.516**
ComplEx	0.247	0.158	0.275	0.428
CustomizE	**0.351**	**0.261**	**0.385**	**0.504**

Table 3: Results of dimension customization framework upon different representative KGE models.

5 Conclusion

In this paper, we propose CustomizE, a novel KGE model that customizes different embedding sizes to varying entities according to their frequencies. Specifically, CustomizE is devised with the dimension customization framework, equipped with a bilevel optimization algorithm crafted to steer the model toward optimal dimension customization in the training process. Particularly, CustomizE is capable of assigning larger embedding sizes to frequent entities, and smaller sizes to infrequent ones. Furthermore, the proposed framework is general and flexible, allowing its application to diverse existing KGE models. Finally, due to the appropriate customized embedding sizes, evaluation on two benchmark datasets demonstrates the effectiveness of CustomizE.

6 Limitations

CustomizE effectively addresses data imbalance issues in KGE by enabling entity-specific dimension learning, enhancing representation precision and model flexibility across diverse KG structures. In line with other KGE models and bilevel optimization approaches, our method encounters common challenges in the field: it incurs higher computational costs for large-scale KGs and complicates hyperparameter tuning due to its adaptive dimension selection mechanism. These aspects may impact scalability and optimization efficiency in practical applications.

Acknowledgements

The research work is supported by the National Natural Science Foundation of China under Grant No.62206266.

References

Ralph Abboud, Ismail Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. 2020. Boxe: A box embedding model for knowledge base completion. *Advances in Neural Information Processing Systems*, 33:9649–9661.

Ivana Balazevic, Carl Allen, and Timothy Hospedales. 2019. Multi-relational poincaré graph embeddings. *Advances in Neural Information Processing Systems*, 32.

Ivana Balažević, Carl Allen, and Timothy M Hospedales. 2019. Hypernetwork knowledge graph embeddings. In *Artificial Neural Networks and Machine Learning–ICANN 2019: Workshop and Special Sessions: 28th International Conference on Artificial Neural Networks, Munich, Germany, September 17–19, 2019, Proceedings 28*, pages 553–565. Springer.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

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

Zalán Borsos, Mojmir Mutny, and Andreas Krause. 2020. Coresets via bilevel optimization for continual learning and streaming. *Advances in Neural Information Processing Systems*, 33:14879–14890.

Arnav Chavan, Zhiqiang Shen, Zhuang Liu, Zechun Liu, Kwang-Ting Cheng, and Eric P Xing. 2022. Vision transformer slimming: Multi-dimension searching in continuous optimization space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4931–4941.

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

Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. 2020. A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Zhen Jia, Soumajit Pramanik, Rishiraj Saha Roy, and Gerhard Weikum. 2021. Complex temporal question answering on knowledge graphs. In *Proceedings of the 30th ACM international conference on information & knowledge management*, pages 792–802.

Seyed Mehran Kazemi and David Poole. 2018. Simple embedding for link prediction in knowledge graphs. *Advances in neural information processing systems*, 31.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. 2018. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*.

Liang Qu, Yonghong Ye, Ningzhi Tang, Lixin Zhang, Yuhui Shi, and Hongzhi Yin. 2022. Single-shot embedding dimension search in recommender system. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 513–522.

Hongyu Ren, Hanjun Dai, Bo Dai, Xinyun Chen, Michihiro Yasunaga, Haitian Sun, Dale Schuurmans, Jure Leskovec, and Denny Zhou. 2021. Lego: Latent execution-guided reasoning for multi-hop question answering on knowledge graphs. In *International Conference on Machine Learning*, pages 8959–8970. PMLR.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. Learning to reweight examples for robust deep learning. In *International conference on machine learning*, pages 4334–4343. PMLR.

Andrea Rossi, Denilson Barbosa, Donatella Firmani, Antonio Matinata, and Paolo Merialdo. 2021. Knowledge graph embedding for link prediction: A comparative analysis. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(2):1–49.

Prithviraj Sen, Breno WSR de Carvalho, Ryan Riegel, and Alexander Gray. 2022. Neuro-symbolic inductive logic programming with logical neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8212–8219.

Zhan Su, Zhicheng Dou, Yutao Zhu, and Ji-Rong Wen. 2022. Knowledge enhanced search result diversification. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1687–1695.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd workshop on* *continuous vector space models and their compositionality*, pages 57–66.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *The International Conference on Machine Learning*, pages 2071–2080.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Alvin Wan, Xiaoliang Dai, Peizhao Zhang, Zijian He, Yuandong Tian, Saining Xie, Bichen Wu, Matthew Yu, Tao Xu, Kan Chen, et al. 2020. Fbnetv2: Differentiable neural architecture search for spatial and channel dimensions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12965–12974.

Shen Wang, Xiaokai Wei, Cicero Nogueira Nogueira dos Santos, Zhiguo Wang, Ramesh Nallapati, Andrew Arnold, Bing Xiang, Philip S Yu, and Isabel F Cruz. 2021. Mixed-curvature multi-relational graph neural network for knowledge graph completion. In *Proceedings of the Web Conference 2021*, pages 1761–1771.

Yun Cheng Wang, Xiou Ge, Bin Wang, and C-C Jay Kuo. 2023. Greenkgc: A lightweight knowledge graph completion method. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10596–10613.

Tianyang Xie and Yong Ge. 2024. Enhance knowledge graph embedding by mixup. *IEEE Transactions on Knowledge & Data Engineering*, 36(02):569–580.

Caijun Xu, Fuwei Zhang, Zhao Zhang, Fuzhen Zhuang, and Rui Liu. 2024. Exploring high-order user preference with knowledge graph for recommendation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 4138–4142.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *International Conference on Learning Representations*.

Fuwei Zhang, Zhao Zhang, Xiang Ao, Dehong Gao, Fuzhen Zhuang, Yi Wei, and Qing He. 2022a. Mind the gap: Cross-lingual information retrieval with hierarchical knowledge enhancement. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 4345–4353.

Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019. Quaternion knowledge graph embeddings. *Advances in neural information processing systems*, 32.

Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2020. Learning hierarchy-aware knowledge graph embeddings for link prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3065–3072.

Zhao Zhang, Fuzhen Zhuang, Meng Qu, Zheng-Yu Niu, Hui Xiong, and Qing He. 2021. Knowledge graph embedding with shared latent semantic units. *Neural Networks*, 139:140–148.

Zhaoli Zhang, Zhifei Li, Hai Liu, and Neal N Xiong. 2022b. Multi-scale dynamic convolutional network for knowledge graph embedding. *IEEE Transactions on Knowledge & Data Engineering*, 34(05):2335–2347.

Xiangyu Zhao, Haochen Liu, Hui Liu, Jiliang Tang, Weiwei Guo, Jun Shi, Sida Wang, Huiji Gao, and Bo Long. 2021. Autodim: Field-aware embedding dimension searchin recommender systems. In *Proceedings of the Web Conference 2021*, pages 3015–3022.

Xiangyu Zhaok, Haochen Liu, Wenqi Fan, Hui Liu, Jiliang Tang, Chong Wang, Ming Chen, Xudong Zheng, Xiaobing Liu, and Xiwang Yang. 2021. Autoemb: Automated embedding dimensionality search in streaming recommendations. In *2021 IEEE International Conference on Data Mining (ICDM)*, pages 896–905. IEEE.

Zhuoxun Zheng, Baifan Zhou, Hui Yang, Zhipeng Tan, Arild Waaler, Evgeny Kharlamov, and Ahmet Soylu. 2024. Low-dimensional hyperbolic knowledge graph embedding for better extrapolation to under-represented data. In *European Semantic Web Conference*, pages 100–120. Springer.