CogKGE: A Knowledge Graph Embedding Toolkit and Benchmark for Representing Multi-source and Heterogeneous Knowledge

Zhuoran Jin[*1,2], **Tianyi Men**[*1,2], **Hongbang Yuan**[*1,2], **Zhitao He**[1,2], **Dianbo Sui**[1,2], **Chenhao Wang**[1,2], **Zhipeng Xue**[1], **Yubo Chen**[1,2], **Jun Zhao**[1,2]

[1] National Laboratory of Pattern Recognition, Institute of Automation, CAS, Beijing, China

[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China

{zhuoran.jin, dianbo.sui, yubo.chen, jzhao}@nlpr.ia.ac.cn

{mentianyi2022, yuanhongbang2022, hezhitao2021}@ia.ac.cn

Abstract

In this paper, we propose CogKGE, a knowledge graph embedding (KGE) toolkit, which aims to represent the **multi-source** and **heterogeneous** knowledge. For multi-source knowledge, unlike existing methods that mainly focus on entity-centric world knowledge, CogKGE also supports the representations of event-centric world knowledge, commonsense knowledge and linguistic knowledge. For heterogeneous knowledge, besides structured triple facts, CogKGE leverages additional unstructured information, such as text descriptions, node types and temporal information, to enhance the meaning of embeddings. Moreover, CogKGE aims to provide a unified programming framework for KGE tasks and a series of knowledge representations for downstream tasks. As a research framework, CogKGE consists of five parts, including core, data, model, knowledge and adapter module. As a knowledge discovery toolkit, CogKGE provides pre-trained embedders to discover new facts, cluster entities and check facts. Furthermore, we construct two new benchmark datasets for further research on multi-source heterogeneous KGE tasks: EventKG240K and CogNet360K. We also release an online system [1] to discover knowledge visually. Source code, datasets and pre-trained embeddings are publicly available at GitHub [2], with a short instruction video [3].

1 Introduction

In recent years, knowledge graphs (KGs) have experienced rapid development. A large number of KGs, such as FrameNet (Baker et al., 1998), Wikidata (Vrandečić and Krötzsch, 2014), DBpedia (Lehmann et al., 2015) and ConceptNet (Speer et al., 2017), have been built and successfully applied to many real-world applications. Most

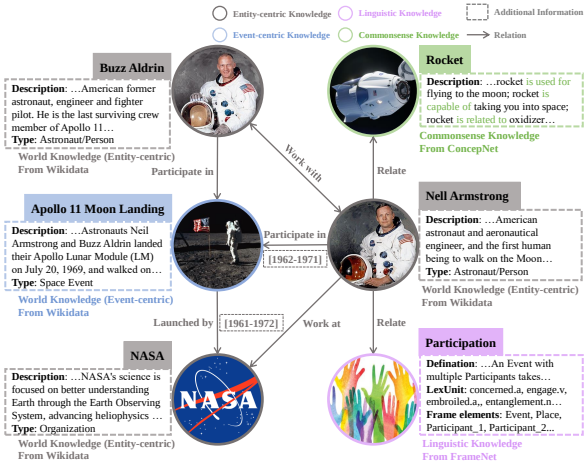


Figure 1: An example of a multi-source heterogeneous KG. Grey, blue, green and purple denote entity-centric world knowledge, event-centric world knowledge, commonsense knowledge and linguistic knowledge, respectively. The dotted boxes show additional information.

KGs are originally organized in the form of triples (h, r, t), where h and t indicate head and tail entities, and r indicates the relation between h and t. However, a KG is a symbolic system that cannot be directly applied to large-scale deep learning frameworks. To this end, a series of knowledge graph embedding (KGE) models have been proposed to represent the entities and relations into continuous spaces (Bordes et al., 2013; Wang et al., 2014; Lin et al., 2015; Sun et al., 2018; Abboud et al., 2020).

To facilitate the development of KGE models, some remarkable KGE toolkits, such as OpenKE (Han et al., 2018), Graphvite (Zhu et al., 2019), LibKGE (Broscheit et al., 2020), PyKEEN (Ali et al., 2021) and Pykg2vec (Yu et al., 2021) have been released, providing easy-to-use frameworks for a series of KGE models. However, most of them perform the embedding task solely based on entity-related triple facts, so they are still limited to two critical challenges in practical applications: **multi-source** challenge and **heterogeneous** challenge.

As to the multi-source challenge, real-world

[*] These authors contribute equally to this work.

[1] http://cognlp.com/cogkge/

[2] https://github.com/jinzhuoran/CogKGE/

[3] https://youtu.be/BiA2Rm9JYKs/

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
System Demonstrations, pages 166 - 173

KGs involve not only world knowledge (including entity-centric knowledge and event-centric knowledge), but also linguistic knowledge and commonsense knowledge. In various practical applications, we need to use multi-source knowledge simultaneously. For example, as shown in Figure 1, to understand an article about *"Neil Armstrong"*, we need (1) entity-centric world knowledge, e.g., *"Neil Armstrong worked at NASA"* from Wikidata; (2) event-centric world knowledge, e.g., *"Neil Armstrong is a participator of the Apollo 11 Moon Landing"* from Wikidata; (3) linguistic knowledge, e.g., the linguistic frame of *"Participation"* from FrameNet; (4) commonsense knowledge, e.g., *"rocket is used for flying to the moon"* from ConceptNet. However, most existing toolkits only focus on representing world knowledge, especially entity-centric knowledge, while ignoring other knowledge, like commonsense knowledge and linguistic knowledge. Therefore, developing a toolkit that can represent multi-source knowledge is essential.

As to the heterogeneous challenge, real-world KGs involve not only triple facts, but also additional information, such as text descriptions, node types and temporal information. In many practical applications, we should use these heterogeneous knowledge together. Likewise, as shown in Figure 1, to understand an article about *"Neil Armstrong"*, besides structured triple facts, we also need (1) text descriptions, e.g., *"Neil Armstrong was the first human being to walk on the Moon"*; (2) node types, e.g., *"Neil Armstrong is an astronaut"*; (3) temporal information, e.g., *"Nell Armstrong participated in Apollo 11 Moon Landing from 1962 to 1971"*. All these heterogeneous knowledge can be used for obtaining the embeddings, but conventional KGE models cannot take full advantage of the additional information mentioned above. Therefore, it is highly desirable to have a toolkit that can bridge these heterogeneous knowledge by plug-and-play knowledge adapters.

To solve the above two problems, we propose CogKGE, a knowledge graph embedding toolkit that aims to represent multi-source and heterogeneous knowledge. The toolkit consists of five parts, including core module, data module, model module, adapter module and knowledge module. CogKGE currently supports 17 models, 11 datasets, five evaluation metrics, four knowledge adapters, four loss functions, three samplers and three built-in data containers. Besides, we also construct two large-scale benchmark datasets to promote the research on KGE. In summary, the main features and contributions are as follows:

- **Multi-source and heterogeneous knowledge representation.** CogKGE explores the unified representation of knowledge from diverse sources. Moreover, our toolkit not only contains the triple fact-based embedding models, but also supports the fusion representation of additional information, including text descriptions, node types and temporal information.
- **Comprehensive models and benchmark datasets.** CogKGE has implemented 17 classic KGE models of four categories, including translation distance models, semantic matching models, graph neural network-based models and transformer-based models. Besides nine built-in public datasets, we also release two new large benchmark datasets for further evaluating KGE methods, called EventKG240K and CogNet360K.
- **Extensible and modularized framework.** CogKGE provides a programming framework for KGE tasks. Based on the extensible architecture, CogKGE can meet the requirements of module extension and secondary development, and pre-trained knowledge embeddings can be directly applied to downstream tasks.
- **Open source and online demo.** Besides the toolkit, we also release an online CogKGE demo to discover knowledge visually. Source code, datasets and pre-trained embeddings are publicly available at GitHub.

2 System Architecture

The overall system architecture of CogKGE is presented in Figure 2. The top part is composed of the core module and data module. The former is the basis of the toolkit, while the latter provides fundamental data containers, loaders and processors. The bottom part is built upon the top part, the model module contains lots of built-in models, the knowledge module integrates multi-source and heterogeneous knowledge, and the adapter module acts as a bridge between the two. In the following, we will cover these five modules in detail.

2.1 Core Module

In the core module, we develop an extensible framework and various ready-to-use components.

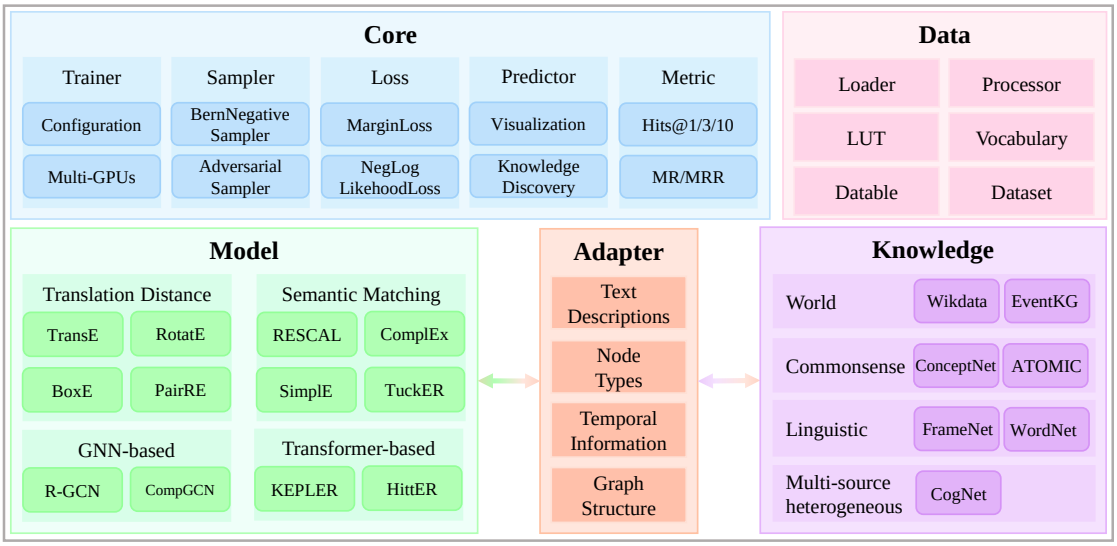


Figure 2: The main architecture of CogKGE.

Trainer, Evaluator and Predictor. Since training, evaluation, and prediction are the core processes in the deep learning pipeline, we design `Trainer` class, `Evaluator` class and `Predictor` class to implement them, respectively. To improve the efficiency of our toolkit, we also involve some functions to support multi-GPUs training, breakpoints resume, logs record and result visualization as shown in Appendix A.

Loss and Sampler. All models in CogKGE are trained by minimizing `MarginLoss` function or `NegLogLikehoodLoss` function. Both of these loss functions need to construct false triples as negative samples. We encapsulate the efficient process of constructing negative samples in `Sampler` class, including `UniSampler` class, `BernSampler` class and `AdversarialSampler` class.

Metric. KGE models are usually evaluated on link prediction, which aims to predict the missing entities in triples $(?, r, t)$ or $(h, r, ?)$. In CogKGE, the `Metric` class computes the ratio of answers ranked top-k (Hits@1/3/10), the mean rank of the answers (MR) and the mean reciprocal rank of the answers (MRR), both raw and filtered results are available.

2.2 Data Module

A primary design principle of CogKGE is to support unified KGE tasks. For this purpose, the data module is based on easy-to-use data containers, such as `LUT` class for looking up items in table form, `Vocabulary` class for converting labels to indexes. To improve reusability, CogKGE includes built-in `Loader` and `Processor` class for many benchmarking datasets and is compatible with multi-source heterogeneous KGs with additional information.

2.3 Model Module

`BaseModel` class is the base class of all models in CogKGE. `BaseModel` class organizes code into three basic sections: (1) `forward` function for training, (2) `embedding` function for getting the embedding of entities and relations and (3) `scoring` function for computing the score of triples. The model module consists of four parts, which are: translation distance models, semantic matching models, graph neural network-based models and transformer-based models.

Translation Distance Models. The translation distance models use distance-based measures to compute the similarity score for a pair of entities and their relationships. In CogKGE, the similarity score function of translation distance models is generally defined as:

$$f_r(h,t) = \|g_h(\mathbf{h}) + \mathbf{r} - g_t(\mathbf{t})\|_{\ell_1/\ell_2}^{1/2}, \quad (1)$$

where $\mathbf{h}$, $\mathbf{r}$, $\mathbf{t}$ are the embedding representations of h, r, t, $g_h(\cdot)$ and $g_t(\cdot)$ are the transformation functions. The translation-based models aim to find a vector representation of entities with relation to the translation of the entities. In CogKGE, we implement several translational distance models, including TransE (Bordes et al., 2013), TransH (Wang et al., 2014), TransR (Lin et al., 2015), TransD (Ji

et al., 2015), TransA (Xiao et al., 2015), RotatE (Sun et al., 2018), BoxE (Abboud et al., 2020) and PairRE (Chao et al., 2020).

Semantic Matching Models. Compared with the distance-based score function of translation distance models, semantic matching models use the similarity-based score function. They measure the plausibility of facts by matching latent semantics of entities and relations embodied in their vector space representations. RESCAL (Nickel et al., 2011), DistMult (Yang et al., 2015), ComplEx (Trouillon et al., 2016), SimplE (Kazemi and Poole, 2018) and TuckER (Balažević et al., 2019) have been built into CogKGE.

Graph Neural Network-based Models. Graph neural network (GNN) has recently been shown to be quite successful in modelling graph-structured data. Considering that KG itself happens to be a kind of graph-structured data, GNN can integrate the topological structure and node feature, then provides a more refined vector representation. We implement R-GCN (Schlichtkrull et al., 2018) and CompGCN (Vashishth et al., 2019) to represent the multi-relational data.

Transformer-based Models. Transformer has been widely used in pre-trained language models, and its deep network architecture can learn contextual representations of entities and relations in a KG jointly by aggregating information from graph neighbourhoods. Besides, transformer-based models can also utilize the text descriptions in KGs, encoding the texts and facts into a unified semantic space. We have implemented KEPLER (Wang et al., 2021b) and HittER (Chen et al., 2021).

2.4 Knowledge Module

The knowledge module mainly integrates three kinds of knowledge representation, namely world, commonsense and linguistic knowledge.

World Knowledge. Encyclopedia KGs such as Freebase, DBpedia and Wikidata mainly focus on explicit world knowledge, containing facts about specific instances, e.g., (*Neil Armstrong, Work at, NASA*). Besides entity-centric knowledge, event-centric knowledge is also an essential kind of knowledge, which conveys dynamic and procedural knowledge, e.g., (*Neil Armstrong, Participate in, Apollo 11 Moon Landing*). In CogKGE, we implement entity-centric knowledge representation based on Wikidata and event-centric knowledge representation based on EventKG (Gottschalk and Demidova, 2018). World knowledge representations have been widely used in knowledge-enhanced pre-trained language models, entity disambiguation and event extraction.

Commonsense Knowledge. Commonsense knowledge tries to capture implicit general facts and regular patterns in our daily life. Nodes in commonsense KG are semantically rich natural language phrases rather than entities. CogKGE supports the commonsense knowledge representation of ConceptNet, which can be helpful for commonsense completion and reasoning.

Linguistic Knowledge. Linguistic knowledge includes considerable information about lexical, conceptual and predicate argument semantics. For example, *"participation"* has hyponymy relation to *"engagement"* in WordNet, while *"take part"* can evoke the *"Participation"* frame in FrameNet. In CogKGE, the knowledge representation of FrameNet can be applied for downstream tasks, such as word sense disambiguation and machine reading comprehension.

2.5 Adapter Module

Almost all of the models in Section 2.3 embed KGs to a specific feature space only based on the triple facts (h, r, t). In practice, as shown in Section 2.4, multi-source and heterogeneous knowledge representation is more realistic and valuable. There is a lot of additional information in KGs that can further enhance and refine the knowledge representation. Inspired by the adapter pattern in the design patterns, we leverage plug-and-play knowledge adapters to build a bridge between KGE models and multi-source heterogeneous data.

Text Descriptions Adapter. As shown in Figure 1, there are text descriptions of entities in KGs, containing abundant semantic information about them. The challenge of KGE with text description is to embed both structured fact knowledge and unstructured textual information in the same space. According to KEPLER (Wang et al., 2021b), we adopt RoBERTa (Liu et al., 2019) as the encoder to generate the entity embeddings based on text descriptions. For a triple (h, r, t), we have:

$$\begin{aligned} \mathbf{h} &= \text{Encoder}(h_d) \\ \mathbf{t} &= \text{Encoder}(t_d), \end{aligned} \tag{2}$$

where h_d and t_d are the text descriptions for h and t. Users can replace the traditional embedding matrixes with the text descriptions adapter without modifying `scoring` function of models. Models with the text description adapters can generate embeddings from their descriptions for those entities invisible during the training stage.

Node Types Adapter. In most KGs, nodes are represented with hierarchical types or categories. For example, *"Neil Armstrong"* belongs to *"Astronaut"* and *"Person"* category. To implement the node types adapter, we use type-specific entity projections based on TKRL (Xie et al., 2016), which is defined as:

$$\begin{aligned} g_h(\mathbf{h}) &= \mathbf{M}_{ch}\mathbf{h} \\ g_t(\mathbf{t}) &= \mathbf{M}_{ct}\mathbf{t}, \end{aligned} \tag{3}$$

where $\mathbf{M}_{ch}$ and $\mathbf{M}_{ct}$ are the projection matrixes of h and t belonging to category c.

Temporal Information Adapter. KG facts are usually time-sensitive, different events and actions cause entities and relations to change over time. For example, Figure 1 illustrates *"Nell Armstrong"* paticipated in *"Apollo 11 Moon Landing"* from *1962* to *1971*. A fact with temporal information in KGs is represented as a quadruple $(h, r, t, [\tau_b, \tau_e])$, where τ_b and τ_e respectively denote the start and end time of the fact. We implement diachronic embedding (DE) (Goel et al., 2020) as the temporal information adapter in CogKGE.

3 System Usage

Our goal of designing CogKGE is to provide a unified research framework for KGE tasks and pre-trained knowledge representations for downstream tasks. In this section, we show a detailed guideline on how to use our toolkit.

3.1 Pre-trained Embedder for Knowledge Discovery

CogKGE provides a series of pre-trained knowledge representations, such as EventKG, CogNet (Wang et al., 2021a) and other KGs. `Predictor` class serves as the pre-trained embedder, whose model and dataset can be selected by users. As shown in Figure 3, `Predictor` class implements the following functions: similar nodes query, head query according to tail and relation, relation query according to head and tail, etc. Pre-trained embedders can be applicable for knowledge discovery.

```
import cogkge
predictor = cogkge.Predictior(model='BoxE', data='EventKG')
# Fuzzy query nodes by keywords
fuzzy_nodes = predictor.fuzzy_query_node('Copa Colombia')
# Fuzzy query relations by keywords
fuzzy_relations = predictor.fuzzy_query_relation('sport')
# Query similar nodes
similar_nodes = predictor.predict_similar_node(node_id=1)
# Given head node and relation, query the tail node
tails = predictor.predcit_tail(head_id=1, relation_id=2)
# Given head node and tail node, query the relation
relations = predictor.predict_relation(head_id=1, tail_id=2)
```

Figure 3: An example of pre-trained embedder.

```
# Load and process data
loader = EVENTKG2MLoader()
train_data, valid_data, test_data = loader.load_all_data()
node_lut, relation_lut = loader.load_all_lut()
processor = EVENTKG2MProcessor()
train_dataset = processor.process(train_data)
valid_dataset = processor.process(valid_data)
test_dataset = processor.process(test_data)
# Initialize components
model = TransE(embedding_dim=200)
loss = MarginLoss()
optimizer = torch.optim.Adam(model.parameters())
negative_sampler = UniNegativeSampler()
metric = Link_Prediction()
lr_scheduler = torch.optim.lr_scheduler.ReduceLROnPlateau()
# Train and evaluate model
trainer = Trainer(train_dataset, valid_dataset, model, loss,
↪  optimizer, negative_sampler, metric, lr_scheduler)
trainer.train()
evaluator = Evaluator(test_dataset, model, metric)
evaluator.evaluate()
```

Figure 4: An example of programming framework.

3.2 Programming Framework for Training Models

As a unified programming framework, CogKGE supports researchers to use various off-the-shelf components to implement new models quickly. Figure 4 shows the sample code of training models. To do this, users need to use `Loader` class to load the lookup tables and process datasets by `Processor` class. Then, `Model`, `Loss`, `Metric`, `Optimizer`, `Sampler` class should be initialized before added to `Trainer` class. And finally, `Trainer` and `Evaluator` class can automatically train and validate the model.

3.3 Online System for Visualization

In addition to this toolkit, we also release an online system as shown in Figure 5. We implement high-performance KGE models for large-scale KGs and deploy pre-trained knowledge embedders for online access. The online system can be directly used for querying nodes and relations in various forms, and in the meantime, dimensionality reduction and visualization of nodes are supported.

4 Evaluation Benchmark

To evaluate the KGE models on large-scale multi-source heterogeneous KGs, we construct two new benchmark datasets: EventKG240K and

Model	EventKG240K					CogNet360K				
	Hits@1	Hits@3	Hits@10	MR	MRR	Hits@1	Hits@3	Hits@10	MR	MRR
RESCAL	6.3	14.3	29.4	1644.8	13.7	1.0	2.6	7.7	1734.9	4.0
TransE	6.2	16.1	34.7	1019.1	15.1	0.7	2.8	8.6	1167.0	4.1
TransH	6.7	15.9	32.5	1109.4	15.0	0.6	2.6	8.3	2077.9	4.0
DistMult	7.1	15.1	31.2	1113.6	14.8	1.4	3.7	10.4	923.9	5.1
ComplEx	8.4	19.7	41.1	1513.5	18.4	0.7	2.2	7.4	1167.2	3.8
RotatE	8.3	**22.3**	**45.6**	**717.3**	**19.8**	**1.9**	**4.7**	**12.2**	**230.0**	**6.0**
SimplE	**9.2**	20.6	42.8	2354.5	19.2	1.3	3.3	9.0	2973.3	4.7
BoxE	8.3	17.5	34.5	1871.8	16.5	1.4	3.8	10.1	355.5	5.1
PairRE	7.7	20.3	39.5	1051.0	17.7	1.3	4.1	11.3	810.6	5.4

Table 1: Link prediction results on EventKG240K and CogNet360K (% except MR). Under the raw evaluation setting, we do not remove the corrupted triples before ranking. The embedding dimension is 50.

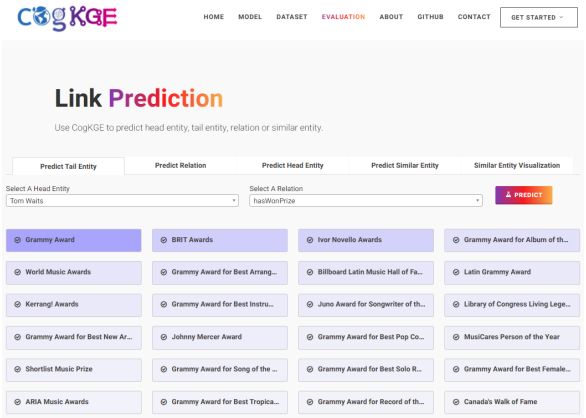


Figure 5: An example of online system.

CogNet360K. In this section, we introduce our datasets and conduct evaluations for classic models included in CogKGE.

4.1 EventKG240K

EventKG is an event-centric temporal knowledge graph. To our best knowledge, EventKG240K is the first event-centric KGE dataset. We use EventKG V3.0 data to construct the dataset. First, we filter entities and events based on their degrees. Then, we select the triple facts when both nodes' degrees are greater than 10. At last, we add text descriptions and node types for nodes and translate triples to quadruples by temporal information. The whole dataset contains 238,911 nodes, 822 relations and 2,333,986 triples.

4.2 CogNet360K

CogNet is a multi-source heterogeneous KG dedicated to integrating linguistic, world and commonsense knowledge. To build a subset, we count the number of occurrences for each node. Then, we sort frame instances by the minimum occurrences of their connected nodes. After the sorted frame instances, we filter the triple facts according to the preset frame categories. The final dataset contains 360,637 nodes, 45 relations and 1,470,488 triples.

4.3 Performance

To assess the challenges of EventKG240K and CogNet360K, we benchmark several popular KGE models on our dataset and select Hits@1/3/10, MR and MRR as the metrics. Table 1 shows the performance of KGE models on EventKG240K and CogNet360K, and the evaluation result shows that both datasets are more challenging due to their multi-source and heterogeneous features. For EventKG240K, traditional KGE models can not distinguish events and entities well. For CogNet360K, it is difficult for vanilla KGE methods to represent multiple kinds of knowledge uniformly. The results advocate for more efforts towards large-scale multi-source heterogeneous KGE tasks.

5 Conclusion

In this paper, we propose CogKGE, a knowledge graph embedding toolkit and benchmark for representing multi-source and heterogeneous knowledge. For multi-source knowledge, CogKGE explores the unified representation of world, commonsense and linguistic knowledge. For heterogeneous knowledge, CogKGE incorporates the structured and unstructured knowledge to enhance the meaning of embeddings. So far, we have implemented 17 classic KGE models. Besides nine public datasets, we also release two new benchmark datasets for further evaluating KGE models. Moreover, owing to the extensible and modularized architecture,

CogKGE is not only a KGE research framework, but also a knowledge discovery library. Besides the toolkit, we also release an online system to discover knowledge visually. In the future, more models, benchmark datasets, and knowledge adapters will be incorporated into CogKGE.

Acknowledgements

We thank the anonymous reviewers for their constructive comments. This work is supported by the National Key Research and Development Program of China (No.2020AAA0106400), the National Natural Science Foundation of China (No.61976211 and No.62176257).

References

Ralph Abboud, Ismail Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. 2020. Boxe: A box embedding model for knowledge base completion. *Proc. of NIPS*.

Mehdi Ali, Max Berrendorf, Charles Tapley Hoyt, Laurent Vermue, Sahand Sharifzadeh, Volker Tresp, and Jens Lehmann. 2021. Pykeen 1.0: A python library for training and evaluating knowledge graph embeddings. *Journal of Machine Learning Research*.

Collin F Baker, Charles J Fillmore, and John B Lowe. 1998. The berkeley framenet project. In *Proc. of ACL*.

Ivana Balažević, Carl Allen, and Timothy Hospedales. 2019. Tucker: Tensor factorization for knowledge graph completion. In *Proc. of EMNLP*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proc. of NIPS*.

Samuel Broscheit, Daniel Ruffinelli, Adrian Kochsiek, Patrick Betz, and Rainer Gemulla. 2020. Libkge: A knowledge graph embedding library for reproducible research. In *Proc. of EMNLP: System Demonstrations*.

Linlin Chao, Jianshan He, Taifeng Wang, and Wei Chu. 2020. Pairre: Knowledge graph embeddings via paired relation vectors. *ArXiv:2011.03798*.

Sanxing Chen, Xiaodong Liu, Jianfeng Gao, Jian Jiao, Ruofei Zhang, and Yangfeng Ji. 2021. HittER: Hierarchical transformers for knowledge graph embeddings. In *Proc. of EMNLP*.

Rishab Goel, Seyed Mehran Kazemi, Marcus Brubaker, and Pascal Poupart. 2020. Diachronic embedding for temporal knowledge graph completion. In *Proc. of AAAI*.

Simon Gottschalk and Elena Demidova. 2018. Eventkg: A multilingual event-centric temporal knowledge graph. In *Proc. of ESWC*.

Xu Han, Shulin Cao, Xin Lv, Yankai Lin, Zhiyuan Liu, Maosong Sun, and Juanzi Li. 2018. Openke: An open toolkit for knowledge embedding. In *Proc. of EMNLP: System Demonstrations*.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proc. of ACL*.

Seyed Mehran Kazemi and David Poole. 2018. Simple embedding for link prediction in knowledge graphs. In *Proc. of NIPS*.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proc. of AAAI*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv:1907.11692*.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *Proc. of ICML*.

Michael Sejr Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *Proc. of ESWC*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proc. of AAAI*.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2018. Rotate: Knowledge graph embedding by relational rotation in complex space. In *Proc. of ICLR*.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proc. of ICML*.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. 2019. Composition-based multi-relational graph convolutional networks. In *Proc. of ICLR*.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*.

Chenhao Wang, Yubo Chen, Zhipeng Xue, Yang Zhou, and Jun Zhao. 2021a. Cognet: Bridging linguistic knowledge, world knowledge and commonsense knowledge. In *Proc. of AAAI*.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021b. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proc. of AAAI*.

Han Xiao, Minlie Huang, Yu Hao, and Xiaoyan Zhu. 2015. Transa: An adaptive approach for knowledge graph embedding. *ArXiv:1509.05490*.

Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2016. Representation learning of knowledge graphs with hierarchical types. In *Proc. of IJCAI*.

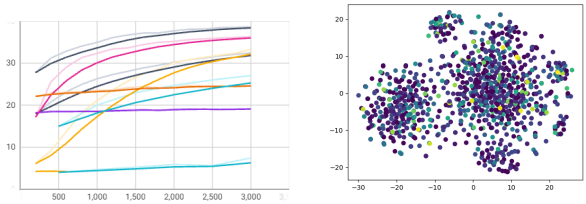
Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *Proc. of ICLR*.

Shih-Yuan Yu, Sujit Rokka Chhetri, Arquimedes Canedo, Palash Goyal, and Mohammad Abdullah Al Faruque. 2021. Pykg2vec: A python library for knowledge graph embedding. *Journal of Machine Learning Research*.

Zhaocheng Zhu, Shizhen Xu, Jian Tang, and Meng Qu. 2019. Graphvite: A high-performance cpu-gpu hybrid system for node embedding. In *Proc. of WWW*.

A Visualization in CogKGE

As shown in Figure 6, CogKGE plots training loss and commonly metrics by Tensorboard. To visualize the high-dimensional embeddings, we use t-SNE dimensionality reduction.



(a) Training Plot. (b) Embedding Plot.

Figure 6: Examples of visualization in CogKGE.

B EventKG240K Statistics

Type	Train	Validation	Test
Nodes	238,911	28,844	28,848
Relations	822	289	301
Event-Event	219,128	1,389	1,427
Event-Entity	1,121,106	9,715	9,731
Entity-Entity	953,774	8,874	8,842
All Triples	2,294,008	19,978	20,000

Table 2: The statistics of EventKG240K.

In this section, we provide more details of our EventKG240K. As shown in Table 2, EventKG240K contains various event-centirc knowledge, especially event-event triples and event-entity triples.

C CogNet360K Statistics

In this section, we provide more details of our CogNet360K. As shown in Table 3, CogNet360K contains rich multi-source and heterogeneous knowledge.

Type	Train	Validation	Test
Nodes	360,637	12,989	13,044
Frames	1,273	253	258
Mini_frames	7,673	0	0
Micro_frames	12,188	1,556	1,558
Synset_frames	5,271	1,189	1,179
Frame_elements	5,642	256	246
Fers	1,419	424	420
Fis	254,384	5,611	5,655
Entitys	72,787	3,700	3,728
Frame-Frame	5,762	250	247
Fe-Frame	5,294	0	0
Fe-Fe	14,819	230	225
Fer-Fer	6,440	368	386
Fer-Micro_frame	5,375	462	444
Micro_frame-Micro_frame	63,202	9,108	9,088
Micro_frame-Frame	24,075	0	0
Mini_frame-Frame	15,346	0	0
Mini_frame-Micro_frame	47,548	0	0
Mini_frame-Synset_frame	21,084	0	0
Synset_frame-Frame	23,133	78	77
Synset_frame-Synset_frame	62,215	8,267	8,316
Synset_frame-Micro_frame	129,773	16,283	16,224
Fi-Fer	220,157	16	13
Fi-Entity	488,789	6,047	6,086
Fi-Micro_frame	255,035	112	114
All Triples	1,388,047	41,221	41,220

Table 3: The statistics of CogNet360K.