Disentangle-based Continual Graph Representation Learning

Xiaoyu Kou[1]*, Yankai Lin[2], Shaobo Liu[2], Peng Li[2], Jie Zhou[2], Yan Zhang[1]
[1]Key Laboratory of Machine Perception (MOE),
Department of Machine Intelligence, Peking University, Beijing, China
[2]Pattern Recognition Center, WeChat AI, Tencent Inc., China
{kouxiaoyu,zhyzhy001}@pku.edu.cn
{yankailin,patrickpli,withtomzhou}@tencent.com

Abstract

Graph embedding (GE) methods embed nodes (and/or edges) in graph into a low-dimensional semantic space, and have shown its effectiveness in modeling multi-relational data. However, existing GE models are not practical in real-world applications since it overlooked the streaming nature of incoming data. To address this issue, we study the problem of *continual graph representation learning* which aims to continually train a graph embedding model on new data to learn incessantly emerging multi-relational data while avoiding catastrophically forgetting old learned knowledge. Moreover, we propose a disentangle-based continual graph representation learning (DiCGRL) framework inspired by the human's ability to learn procedural knowledge. The experimental results show that DiCGRL could effectively alleviate the catastrophic forgetting problem and outperform state-of-the-art continual learning models.

1 Introduction

Multi-relational data represents relationships between entities in the world, which is usually denoted as a multi-relational graph with nodes and edges connecting them. It is widely used in real-world NLP applications such as *knowledge graphs* (KGs) (e.g., Freebase (Bollacker et al., 2008) and DBpedia (Lehmann et al., 2015)) and *information networks* (e.g., social network and citation network). Therefore, modeling multi-relational graph with graph embeddings (Bordes et al., 2013; Tang et al., 2015a; Sun et al., 2019a; Bruna et al., 2014) has been attracting intensive attentions in both academia and industry. Graph embedding (GE), aiming to embed nodes and/or edges in the graph into a low-dimensional semantic space to enable neural models to effectively and efficiently utilize multi-relational data, has demonstrated remarkable effectiveness in various downstream NLP tasks such as question answering (Bordes et al., 2014) and dialogue system (Moon et al., 2019).

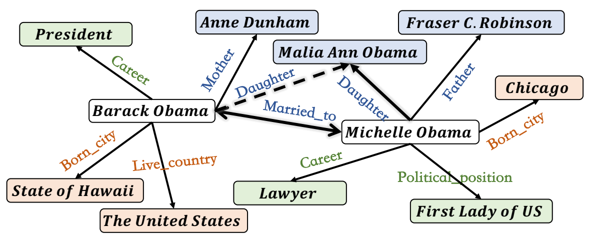


Figure 1: An example of edges that compressed into multiple components, which represented in different colors. The dotted line represents the inferred procedural knowledge from two **bold** edges.

Nevertheless, most existing graph embedding works overlook the streaming nature of the incoming data in real-world scenarios. In consequence, these models have to be retrained from scratch to reflect the data change, which is computationally expensive. To tackle this issue, we propose to study the problem of *continual graph representation learning* (CGRL) in this work.

The goal of continual learning is to alleviate catastrophically forgetting old data while learning new data. There are two mainstream continual learning methods in NLP: (1) consolidation-based methods (Kirkpatrick et al., 2017; Zenke et al., 2017; Liu et al., 2018; Ritter et al., 2018) which consolidate the important model parameters of old data when learning new data; and (2) memory-based methods (Lopez-Paz and Ranzato, 2017; Shin et al., 2017; Chaudhry et al., 2019) which remember a few old examples and learn them with new data jointly. Despite the promising results these methods have achieved on classification tasks, their effectiveness has not been validated on graph representation learning. Unlike the classification problem where instances are generally independent

*This work is done when Xiaoyu Kou was interning at Pattern Recognition Center, WeChat AI, Tencent Inc, China and can be operated individually, nodes and edges in multi-relational data are correlated, making it sub-optimal to directly deploy existing continuous learning methods on the multi-relational data.

认知心理学

In cognitive psychology (Solso et al., 2005), procedural knowledge refers to a set of operational steps. Its smallest unit is production, where multiple productions can complete a series of cognitive activities. When learning new procedural knowledge, humans would update cognitive results by only updating a few related productions and leave the rest intact. Intuitively, such a process can be mimicked to learn constantly growing multi-relational data by regarding each new data as a new procedural knowledge. For example, as illustrated in Figure 1, the relational triplets of *Barack Obama* and *Michelle Obama* are related to three concepts: "family", "occupation" and "location". When a new relational triplet (*Michelle Obama*, `Daughter`, *Malia Ann Obama*) appears, we only need to update the "family"-related information in *Barack Obama*. Consequently, we can further infer that the triplet (*Barack Obama*, `Daughter`, *Malia Ann Obama*) also holds.

Inspired by procedural knowledge learning, we propose a disentangle-based continual graph representation learning framework DiCGRL in this work. Our proposed DiCGRL consists of two modules: (1) **Disentangle module.** It decouples the relational triplets in the graph into multiple independent components according to their semantic aspects, and leverages two typical GE methods including Knowledge Graph Embedding (KGE) and Network Embedding (NE) to learn disentangled graph embeddings; (2) **Updating module.** When new relational triplets arrive, it selects the relevant old relational triplets and only updates the corresponding components of their graph embeddings. Compared with memory-based continual learning methods which save a fixed set of old data, DiCGRL could dynamically select important old data according to new data to fine-tune the model, which makes DiCGRL better model the complex multi-relational data stream.

We conduct extensive experiments on both KGE and NE settings based on the real-world scenarios, and the experimental results show that DiCGRL effectively alleviates the catastrophic forgetting problem and significantly outperforms existing continual learning models while remaining efficient.

2 Related Work

2.1 Graph Embedding

Graph embedding (GE) methods are critical techniques to obtain a good representation of multi-relational data. There are mainly two categories of typical multi-relational data in the real-world, *knowledge graphs* (KGs) and *information networks*. GE handles them via Knowledge Graph Embedding (KGE) and Network Embedding (NE) respectively, and our DiCGRL framework can adapt to the above two typical GE methods, which demonstrates the generalization ability of our model.

KGE is an active research area recently, which can be mainly divided into two categories to tackle link prediction task (Ji et al., 2020). One line of work is reconstruction-based models, which reconstruct the head/tail entity's embedding of a triplet using the relation and tail/head embeddings, such as TransE (Bordes et al., 2013), RotatE (Sun et al., 2019a), and ConvE (Dettmers et al., 2018). Another line of work is bilinear-based models, which consider link prediction as a semantic matching problem. They take head, tail and relation's embeddings as inputs, and measure a semantic matching score for each triplet using bi-linear transformation (e.g., DistMult (Yang et al., 2015), ComplEx (Trouillon et al., 2016), ConvKB (Nguyen et al., 2018)). Besides KGE, NE is also widely explored in both academia and industry. Early works (Perozzi et al., 2014; Tu et al., 2016; Tang et al., 2015b) focus on learning static node embeddings on information graphs. More recently, graph neural networks (Bruna et al., 2014; Henaff et al., 2015; Veličković et al., 2018) have been attracting considerable attention and achieved remarkable success in learning network embeddings. However, most existing GE models assume the training data is static, i.e., do not change over time, which makes them impractical in real-world applications.

2.2 Continual Learning

Continual learning, also known as life-long learning, helps alleviate catastrophic forgetting and enables incremental training for stream data. Methods for continual learning in natural language processing (NLP) field can mainly be divided into two categories: (1) consolidation-based methods (Kirkpatrick et al., 2017; Zenke et al., 2017), which slow down parameter updating to preserve old knowledge, and (2) memory-based methods (Lopez-Paz and Ranzato, 2017; Shin et al., 2017; Chaudhry et al., 2019; Wang et al., 2019), which retain examples from old data for re-play upon learning the new data. Although continual learning has been widely studied in NLP (Sun et al., 2020) and computer vision (Kirkpatrick et al., 2017), its exploration on graph embedding is relatively rare. Sankar et al. (2018) seek to train graph embedding on constantly evolving data. However, it assumes the timestamp information is known beforehand, which hinders its application to other tasks. Song and Park (2018) extend the idea of regulation-based methods to continually learn graph embeddings which straightforwardly limits parameter updating on only the embedding layer. It is therefore hard to generalize to more complex multi-relational data. Our proposed DiCGRL model is distinct from previous works in two aspects: (1) Our method does not require pre-annotated timestamps, which make it more feasible in various types of multi-relational data; (2) Inspired by procedural knowledge learning, we exploit disentanglement to conduct continual learning and achieve promising results.

3 Methodology

3.1 Task Formulation and Overall Framework

We represent multi-relational data as a multi-relational graph $G = (V, E)$, where V, E denote the node set and the edge set within a graph G, and G can be formalized as a set of relational triplets $\{(u, r, v)\} \subseteq V \times E \times V$. Given a relational triplet $(u, r, v) \in G$, we denote the embeddings of them as $\boldsymbol{u}$, $\boldsymbol{v} \in \mathbb{R}^d$ and $\boldsymbol{r} \in \mathbb{R}^l$, where d and l indicate the vector dimension.

Continual graph representation learning trains graph embedding (GE) models on constantly growing multi-relational data, where the i-th part of multi-relational data has its own training set $\mathcal{T}_i$, validation set $\mathcal{V}_i$, and query set $\mathcal{Q}_i$. The i-th training set is defined as a set of relational triplets, i.e., $\mathcal{T}_i = \{(u_1^{\mathcal{T}_i}, r_1^{\mathcal{T}_i}, v_1^{\mathcal{T}_i}), \ldots, (u_N^{\mathcal{T}_i}, r_N^{\mathcal{T}_i}, v_N^{\mathcal{T}_i})\}$, where N is the instance number of $\mathcal{T}_i$. The i-th validation and query sets are defined similarly. As new relational triplets emerges, continual graph representation learning requires GE models to achieve good results on all previous query sets. Therefore, after training on the i-th training set $\mathcal{T}_i$, GE models will be evaluated on $\tilde{\mathcal{Q}}_i = \bigcup_{j=1}^{i} \mathcal{Q}_j$ to measure whether they could well model both new and old multi-relational data. The evaluation protocol indicates that it will be more and more difficult for the model to achieve high performance as the emerging of new relational triplets.

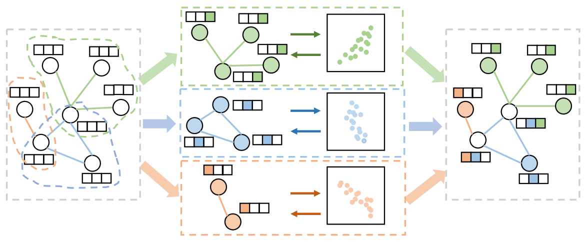


Figure 2: The disentangle module of our model. Different colors indicate different semantic aspects of nodes.

In general, our model continually learns on the streaming data. Whenever there comes a new part of multi-relational data, DiCGRL will learn the new graph embeddings and meanwhile prevent catastrophically forgetting old learned knowledge through two procedures: (1) **Disentangle module**. It decouples the relational triplets in the graph into multiple components according to their semantic aspects, and learns disentangled graph embeddings that divide node embeddings into multiple independent components where each component describes a semantic aspect of node; (2) **Updating module**. When new relational triplets arrive, DiCGRL first activates the old relational triplets from previous graphs which have relevant semantic aspects with the new ones, and only updates the corresponding components of their graph embeddings.

3.2 Disentangle Module

When the i-th training set $\mathcal{T}_i$ becomes available, DiCGRL needs to update the graph embeddings according to these new relational triplets. To this end, for each node $u \in V$, we want to learn a disentangled node embedding $\boldsymbol{u}$, which is composed of K independent components, i.e., $\boldsymbol{u} = [\boldsymbol{u}^1, \boldsymbol{u}^2, ..., \boldsymbol{u}^k, ..., \boldsymbol{u}^K]$, where $(0 \leq k \leq K)$ and $\boldsymbol{u}^k \in \mathbb{R}^d$. The component $\boldsymbol{u}^k$ is used to represent the k-th semantic aspect of node u. As shown in Figure 2, the key challenge of the disentangle module is how to decouple the relational triplets into multiple components according to their semantic aspects, and learn the disentangled graph embeddings in different components independently.

Formally, given a relational triplet $(u, r, v) \in \mathcal{T}_i$, we aim to extract the most related semantic components of u and v with respective to the relation r. Specifically, we model this process with an attention mechanism, where (u, r, v) is associated with K attention weight $(\alpha_r^1, \alpha_r^2, \ldots, \alpha_r^K)$, which respectively represent the probability being assigned to the k-th semantic component. After that, we select the top-n related components of $\boldsymbol{u}$ and $\boldsymbol{v}$ with the highest attention weight. Then we leverage exiting GE methods to extract different features in the selected top-n related components, and we denote the feature extraction operation as $\boldsymbol{f}$. Here, $\boldsymbol{f}$ could be any graph embedding operation that can incorporate the features of node u and v in the selected top-n related components. In this work, we adapt our DiCGRL model in two typical graph embeddings including:

Knowledge Graph Embeddings (KGEs) Intuitively, the most related semantic components of a relational triplet in KG are related to their relation r. Therefore, we can directly set K attention values for each explicit relation r, and the k-th attention value a_r^k $(0 \leq k \leq K)$ is a trainable parameter which indicates how related this edge is to the k-th component. The normalized attention weight α_r^k is computed as:

$$\alpha_r^k = \frac{\exp(a_r^k)}{\sum_{j=1}^{K} \exp(a_r^j)}. \tag{1}$$

As described in related work, KGE models can mainly be divided into two categories: reconstruction-based and bilinear-based models. We explore the effectiveness of both two lines of works to extract features in our framework. Specifically, we leverage two classic KGE models as $\boldsymbol{f}$ to extract latent features in our experiment including TransE (reconstruction-based):

$$\boldsymbol{f} = ||\hat{\boldsymbol{u}} + \boldsymbol{r} - \hat{\boldsymbol{v}}||_p, \tag{2}$$

and ConvKB (bilinear-based):

$$\boldsymbol{f} = \boldsymbol{W}_1 \left(\mathrm{ReLU}\big(\mathrm{Conv}([\hat{\boldsymbol{u}}; \boldsymbol{r}; \hat{\boldsymbol{v}}])\big)\right), \tag{3}$$

where $\hat{\boldsymbol{u}}, \hat{\boldsymbol{v}}$ are the concatenation of top-n component embeddings of node u and node v respectively; $|| \cdot ||_p$ denotes the p-norm operation; $[\cdot;\cdot]$ denotes the concatenate operation; $\mathrm{Conv}(\cdot)$ indicates the convolutional layer with M filters, and $\boldsymbol{W}_1 \in \mathbb{R}^{1\times\frac{Mdn}{K}}$ is a trainable matrix. In total, $\boldsymbol{f}$ is expected to give higher scores for valid triplets than invalid ones.

Network Embeddings (NEs) We first determine α_r^k according to the representations of node u and node v since NE usually does not provide explicit relations. Hence, α_r^k is calculated by performing a non-linearity transformation over the concatenation of $\boldsymbol{u}$ and $\boldsymbol{v}$:

$$\alpha_r^k = \frac{\exp\left(\mathrm{ReLU}(\boldsymbol{W}_2[\boldsymbol{u}^k; \boldsymbol{v}^k])\right)}{\sum_{j=1}^{K} \exp\left(\mathrm{ReLU}(\boldsymbol{W}_2[\boldsymbol{u}^j; \boldsymbol{v}^j])\right)}, \tag{4}$$

where $\boldsymbol{W}_2 \in \mathbb{R}^{1\times 2d}$ is a trainable matrix[1].

Graph attention networks (GATs) (Veličković et al., 2018) gather information from the node's neighborhood and assign varying levels of importance to neighborhoods, which is a widely used and powerful way to learn embeddings for information networks. Thus we leverage GATs as $\boldsymbol{f}$ to extract latent features for NE. Given a target node u and its neighbors $\{v|v \in N_u\}$, we first determine the top-n related components for each pair of nodes (u, v) according to the attention weights α_r^k. When updating the k-th component of u, a neighbor v is considered if and only if the k-th component is in the the top-n related components for the node pair (u, v). In this way, we can thoroughly disentangle the neighbors of the target node into different components to play their roles separately. GATs are used to update each component as follows:

$$\boldsymbol{u}^k = \sum_{v\in N_u} \sigma(\boldsymbol{W}_3 \boldsymbol{v}^k)\boldsymbol{W}_4^k \boldsymbol{v}^k, \tag{5}$$

where $\boldsymbol{W}_3 \in \mathbb{R}^{1\times d}$ and $\boldsymbol{W}_4 \in \mathbb{R}^{h\times d}$ are two trainable matrices, h is hidden size within GATs, and σ is the softmax function which is used to calculate the neighbor's relative attention value in the k-th component.

3.3 Updating Module

Now, the remaining problem is how to update the disentangled graph embedding when new relation triplets appear while preventing catastrophic forgetting. As shown in Figure 3, this process mainly includes two steps:

(1) **Neighbor activation**: DiCGRL needs to identify which relational triplets from $\mathcal{T}_1, ..., \mathcal{T}_{i-1}$ need to be updated. Since in the multi-relational data, nodes are not independent, and therefore a new relational triplet may have influence on the embeddings of nodes that not directly connect to it. Inspired by that, for each relational triplet (u, r, v), we activate both their direct and indirect neighbor triplets[2]. Specifically, neighbors of triplet

[1]Note that we also evaluate this definition on KG data, and the experimental results are presented in Appendix A.

[2]1-order and 2-order neighbors are both considered in our experiments.

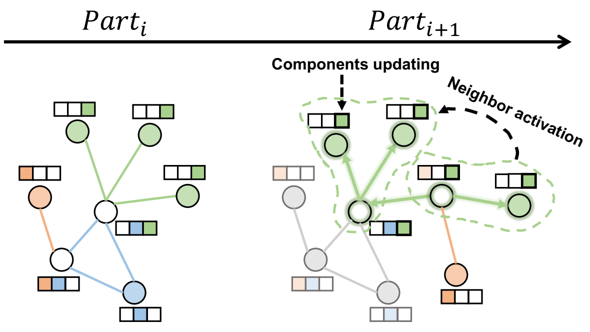


Figure 3: The updating module of DiCGRL, where different colors indicate disentangled components, and a node (white) may contain multiple components.

$(u, r, v) \in \mathcal{T}_i$ refers to all triplets which contain node u or node v on the previous multi-relational graph $(\mathcal{T}_1, ..., \mathcal{T}_{i-1})$. In practice, adding all neighbors to $\mathcal{T}_i$ is computationally expensive occasionally since some nodes have very high degrees[3], i.e., they have a huge amount of neighbors. Therefore, we leverage a selection mechanism inspired by human's ability to learn procedural knowledge introduced in Section 1 and only update a few related neighbors: for each (u, r, v), we only activate the neighbors with related semantic components (i.e., they share at least one component in their top-n semantic components).

(2) **Components updating**: It is not necessary to update all semantic components of activated neighbors. For example, if a relational triplet $(u', r', t') \in (\mathcal{T}_1, \cdots, \mathcal{T}_{i-1})$ is activated by $(u, r, t) \in \mathcal{T}_i$, we only need to update *the common components*, i.e., top-n components, since the semantics of other components does not change. We use existing GE embedding method to update their features, as explained in our disentangle module. Generally speaking, in each epoch, we iteratively train new relational triplets and relevant semantic aspects of activated neighbor relational triplets. Through this training process, our model can not only effectively prevent catastrophic forgetting, but also learn the embeddings of new data.

3.4 Training Objective

As mentioned before, for newly arrived multi-relational data $\mathcal{T}_i$, we iteratively train our model on $\mathcal{T}_i$ and its activated neighbor relational triplets. We denote loss functions of these two parts as $\mathcal{L}_{new}$ and $\mathcal{L}_{old}$ respectively. For KGE, we utilize soft-margin loss to train DiCGRL on link prediction task. The loss function $\mathcal{L}_{new}$ can be defined as:

$$\mathcal{L}_{new} = - \sum_{(u,r,v) \in \mathcal{T}_i \cup \mathcal{T}_i^*} \log\left(1 + \exp\left(y \cdot \boldsymbol{f}(u, r, v)\right)\right), \tag{6}$$

where $\mathcal{T}_i^*$ represents a set of relational invalid triplets for $\mathcal{T}_i$; $y = 1$ if $(u, r, t) \in \mathcal{T}_i$, otherwise, $y = -1$. For NE, we leverage a standard cross entropy loss according to GATs and train our model on node classification task. $\mathcal{L}_{new}$ can be formulated as follows:

$$\mathcal{L}_{new} = - \sum_{u \in N(\mathcal{T}_i)} \frac{1}{|C|} \sum_{c=1}^{|C|} y(c) \cdot \log\left(\sigma(\boldsymbol{W_5 u})\right), \tag{7}$$

where c is node's class and $y(c) = 1$ if the node label is c, otherwise $y(c) = 0$; $N(\mathcal{T}_i)$ is the node set of $\mathcal{T}_i$, $|C|$ indicates the number of class, and $\boldsymbol{W_5} \in \mathbb{R}^{|C| \times d}$ is a trainable matrix. For KGE and NE, $\mathcal{L}_{old}$ can be defined in the same way with $\mathcal{L}_{new}$ on the selected old relational triplets set.

Intuitively, *the less components a relation focuses on, the better the disentanglement is*. Therefore, we add a constraint loss terms $\mathcal{L}_{norm}$ for $\mathcal{T}_i$ [4] to encourage the sum of the attention weights of the top-n selected components to reach 1, i.e.,

$$\mathcal{L}_{norm} = \sum_{(u,r,v) \in \mathcal{T}_i} (1 - \sum_{k}^{n} \alpha_r^k), \tag{8}$$

where n indicates the number of selected components.

The overall loss function $\mathcal{L}$ of our proposed model is defined as follows:

$$\mathcal{L} = \mathcal{L}_{old} + \mathcal{L}_{new} + \beta \cdot \mathcal{L}_{norm}, \tag{9}$$

where β is a hyper-parameter.

4 Experiments

In this section, we evaluate our model on two popular tasks: *link prediction* for knowledge graph and *node classification* for information network.

4.1 Datasets

We conduct experiments on several continual learning datasets adapted from existing graph embedding benchmarks:

KGE datasets. We considered two link prediction benchmark datasets, namely FB15K-237 (Toutanova and Chen, 2015) and

[3]The highest degree of FB15k-237 dataset is 7,614.

[4]The attention weights for the activated neighbor relational triplets are computed using the last checkpoint of the model and not updated during training $\mathcal{T}_i$.

WN18RR (Dettmers et al., 2018). We randomly split each benchmark dataset into five parts to simulate the real world scenarios, with each part having the ratio of $0.8 : 0.05 : 0.05 : 0.05 : 0.05$ respectively [5]. We further divide each part into training set, validation set and query set. The statistics of FB15k-237 and WN18RR datasets are presented in Appendix C.

NE datasets. We conduct our experiments on three real-world information networks for node classification task: Cora, CiteSeer and PubMed (Sen et al., 2008). The nodes, edges and labels in these three citation datasets represent articles, citations and research areas respectively, and their nodes are provided with rich features. Like KGE datasets, we split each dataset into four parts and the partition ratio is $0.7 : 0.1 : 0.1 : 0.1$. We further split train/validation/query set for each part. The statistics of Cora, CiteSeer and PubMed are presented in Appendix C.

4.2 Experimental Settings

We use Adam (Kingma and Ba, 2015) as the optimizer and fine-tune the hyper-parameters on the validation set for each task. We perform a grid search for the hyper-parameters specified as follows: the number of components $K \in \{2, 4, 6, 8, 10\}$, the number of top components $n \in \{2, 4\}$, node embedding dimension $d \in \{100, 200\}$ (note that relation embedding dimension in KG is $l = \frac{d \times n}{K}$, and d is fixed to feature length in information network), initial learning rate $lr \in \{1e-3, 1e-4\}$, and the weight of regulation loss L_{norm} $\beta \in \{0.1, 0.3\}$. The optimal hyper-parameters on FB15k-237 dataset are: $K = 8$, $n = 4$, $d = 200$, $lr = 1e-4$, $\beta = 0.3$; and the optimal hyper-parameters on WN18RR dataset are: $K = 4$, $n = 2$, $d = 200$, $lr = 1e-4$, $\beta = 0.1$. For the NE datasets, the optimal hyper-parameters are: $K = 8$, $n = 4$, $lr = 1e-3$, $\beta = 0.1$.

For a fair comparison, we implement the baseline models (TransE, ConvKB, and GATs) by ourselves based on released codes respectively, and use the same hyper-parameters as DiCGRL. For example, the embedding dimension for TransE and ConvKB are both 200; the number of filters for ConvKB is 50, and the number of heads in GATs is 8. We implement the baseline models of continual learning based on the toolkit [6] released by Han et al. (2020). For fair comparison, we use the same memory size for both our model and baselines. For other hyper-parameters, we also follow the settings in Han et al. (2020).

Following existing works (Bordes et al., 2013), we use the "Filtered" setting protocol, i.e., not taking any corrupted triples that appear in the KG into accounts, and the evaluation metrics of link prediction task include mean reciprocal rank (MRR) and proportion of valid test triplets in top-10 ranks (H@10). For node classification task, we use accuracy as our evaluation metric. We use two settings to evaluate the overall performance of our DiCGRL after learning on all graphs: (1) **whole performance** calculates the evaluation metrics on the whole test set of all data; (2) **average performance** averages the evaluation metrics on all test sets. As average performance highlights the performance of handling catastrophic forgetting problem, thus it is the main metric to evaluate models.

4.3 Baselines

We compare our model with several baselines including two theoretical models to measure the lower and upper bounds of continual learning:

(1) **Lower Bound**, which continually fine-tunes models on the new multi-relational dataset without memorizing any historical instances;

(2) **Upper Bound**, which continually re-train models with all historical and new incoming instances. In fact, this model serves as the ideal upper bound for the performance of continual learning; and several typical continual learning models:

(3) **EWC** (Kirkpatrick et al., 2017), which adopts elastic weight consolidation to add regularization on parameter changes. It uses Fisher information to measure the parameter importance, and slows down the update of those important parameters when learning new data;

(4) **EMR** (Parisi et al., 2019), a basic memory-based method, which memorizes a few historical instances and conducts memory replay. Each time when new data comes in, EMR mixes memorized instances with new instances to fine-tune models;

(5) **GEM** (Lopez-Paz and Ranzato, 2017), which memorizes a few historical instances and adds a constraint on directions of new gradients to make sure that there is no conflict of optimization directions with gradients on old data.

[5] The characteristics of multi-relational graphs in real world scenarios are: 1) large-scale 2) the new multi-relational data is coming every day and small in scale proportional to the original size of these graphs.

[6] https://github.com/thunlp/ContinualRE

	Dataset	FB15k-237		WN18RR	
KGE	Model	W	A	W	A
TransE	Lower Bound	24.1	29.8	11.3	18.9
	EWC	29.8	32.2	14.6	21.3
	EMR	37.4	45.9	25.1	28.6
	GEM	38.3	46.1	25.3	28.9
	DiCGRL	**40.9**	**47.7**	**33.3**	**35.9**
	Upper Bound	43.5	49.7	50.8	49.5
ConvKB	Lower Bound	13.2	19.3	6.3	8.2
	EWC	18.8	22.7	8.5	10.6
	EMR	29.5	33.8	21.3	26.6
	GEM	28.0	35.0	25.3	27.4
	DiCGRL	**31.7**	**40.2**	**33.1**	**37.5**
	Upper Bound	33.9	42.4	51.0	50.2

Table 1: H@10 (%) results of models on two KG benchmarks. “W” stands for the **W**hole performance, and “A” stands for the **A**verage performance.

Dataset	Cora		CiteSeer		PubMed	
Model	W	A	W	A	W	A
Lower	61.2	60.5	60.9	61.8	82.3	81.8
EWC	63.4	61.2	62.3	62.1	82.5	82.4
EMR	72.4	73.9	66.8	68.9	83.1	83.0
GEM	75.3	76.1	70.9	70.2	**85.5**	84.2
DiCGRL	**78.1**	**79.6**	**72.1**	**71.5**	85.1	**85.0**
Upper	84.1	85.5	70.9	73.4	85.9	86.1

Table 2: Accuracy results of models on three information network benchmarks (%). “Lower” and “Upper” are abbreviations of the “Lower Bound” and “‘Upper Bound” baselines.

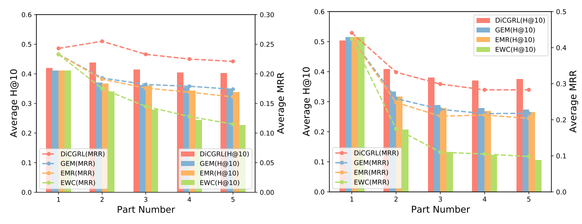
4.4 Overall Results

Table 1 and Table 2 show the overall performance on both KGE and NE benchmarks under two evaluation settings. From the tables, we can see that:

(1) Our proposed DiCGRL model significantly outperforms other baselines and achieves state-of-the-art performance almost in all settings and datasets. It verifies the effectiveness of our disentangled approach in continual learning, which decouples the node embeddings into multiple components with respect to the semantic aspects, and only updates the corresponding components of graph embedding for new relational triplets.

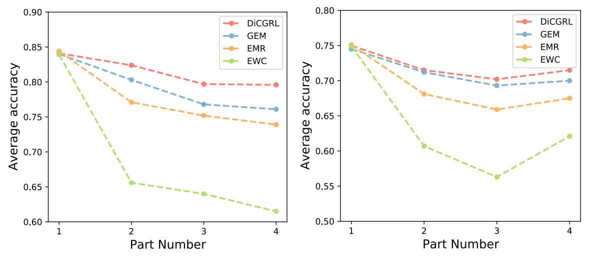
(2) There is still a gap between our model and the upper bound. It indicates although we have proposed an effective approach for continual graph representation learning, it still remains an open problem deserving further exploration.

(3) Although DiCGRL outperforms other baselines in almost all settings in three information network benchmarks, the performance gain is not as high as it is on the KG datasets. The reason is that these three citation benchmarks are provided with rich node features, which would reduce the impact of topology changes. As can be seen, even the weakest Lower Bound achieves relatively high results close to Upper Bound.



(a) FB15k-237 (b) WN18RR

Figure 4: Changes in H@10 and MRR with increasing knowledge graph data through the continual learning process, and the feature extraction method used in DiCGRL is ConvKB.



(a) Cora (b) CiteSeer

Figure 5: Changes in accuracy with increasing information network data through the continual learning process. The result of PubMed dataset is presented in Appendix B.

To further investigate how evaluation metrics change while learning new relational triplets, we show the average performance on the KG and NE datasets at each part in Figure 4 and Figure 5. From the figures, we observe that:

(1) With increasing numbers of new relational triplets, the performance of all the models in almost all the datasets decreases to some degree (CiteSeer may introduce some instability in random data splitting since this dataset is small). This indicates that catastrophically forgetting old data is inevitable, and it is indeed one of the major difficulties for continual graph representation learning.

(2) The memory-based methods outperforms the consolidation-based methods, which demonstrates the memory-based methods may be more suitable for alleviating catastrophic forgetting in

multi-relational data to some extent.

(3) Our proposed DiCGRL model achieves significantly better results compared to other baseline models. It indicates that disentangling relational triplets and updating dynamically selected components of relational triplets are more useful and reasonable than rote memorization of static examples from old multi-relational data.

4.5 Hyper-Parameter Sensitivity

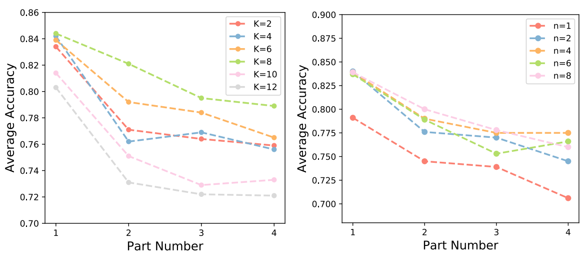
In this section, we investigate the effect of the number of components K and the top selected component number n, which are important hyper-parameters of our DiCGRL. These experiments are only performed on NE datasets, since the node embedding dimension d can be affected by K and n in KG as introduced in Section 4.2, which would make it difficult to make a fair comparison.

Component Number K: We use $n = 2, \beta = 0.1$ to run DiCGRL on the Cora dataset with five different K settings. The results are illustrated in Figure 6(a). From the figure, we find that overall the average accuracy raises when K increases from 2 to 8, which suggests the importance of disentangling components. However, when K grows larger than 8, the performance starts to decline. One possible reason is that the number of components is already larger than that of semantics aspects, making it harder to achieve a good disentanglement. Therefore, we select the component number K for each dataset on the development set, and for most dataset, we select $K = 4$ or $K = 8$.

Top Selected Component Number n: For a fair comparison, we set $K = 8, \beta = 0.1$ and vary n on the Cora dataset. As shown in Figure 6(b), except for the case of $n = 1$, the other settings have comparable performance. However, it can be seen that when $n = 4$, the average accuracy on the last task is the highest, which indicates that the model has the strongest ability to avoid catastrophic forgetting problem when $n = 4$.

4.6 Efficiency Analysis

We show the training time of different continual learning methods on the biggest benchmark FB15k-237, so as to highlight the efficiency gap in different methods. For a fair comparison, all algorithms use the same KGE method TransE. As shown in Figure 7, GEM takes much longer training time compared with other baselines, which is pretty close to Upper Bound. Although our model also requires some previous data, it takes less time than GEM



(a) Effect of K (b) Effect of n

Figure 6: Hyper-parameter sensitivity of K and n.

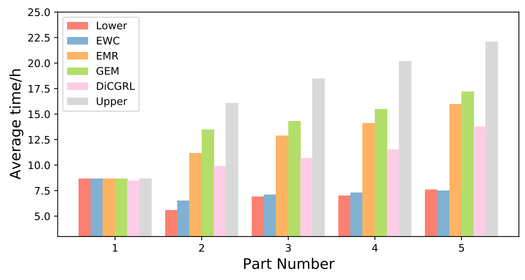
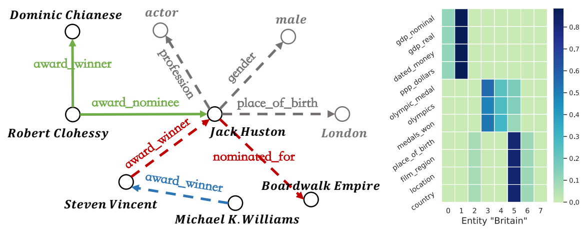


Figure 7: Training time (hour) on FB15k-237 datasets through the continual learning process.



(a) Example (b) Visualization

Figure 8: Figure(a) is an example of DiCGRL on the second part of FB15k-237 dataset, where solid line and dotted line represent newly arrived and previous relational triplets respectively. Red and blue color represents the activated 1-order and 2-order neighbors respectively. Figure(b) is the visualization of attention values on the entity *Britain*

and EMR, which verifies the efficiency of our disentangled approach in continual learning.

4.7 Case Study

In this section, we visualize an example case from FB15k-237 dataset (more readable than other datasets) to show that the activated neighbors in our updating module are in line with human commonsense. For example, as shown in Figure 8(a) newly arrived relational triplets such as (*Robert Clohessy*, `award_nominee`, *Jack Huston*) and (*Robert Clohessy*, `award_winner`, *Dominic Chianese*), both related to "award" semantic aspects. Therefore, only "award"-related neighbors of new triplets are updated, like (*Jack Huston*, `nominated_for`, *Boardwalk Empire*). since *Robert Clohessy* is also very likely to be related to the movie of *Boardwalk Empire*. Meanwhile, relational triples of `place_of_birth` and `gender`, which are not related to "award", will not be updated.

Moreover, to verify the learned representation satisfies the intuition that different relations focus on different components of entities, we plot the attention values on the components of the entity *Britain* in Figure 8(b), where the y-coordinate is sampled relations that appear in the same triplets with "*Britain*". We observe that semantically similar relations have similar attention value distributions. For example, relations "`gdp_nominal`", "`gdp_real`", "`dated_money`", "`ppp_dollars`", are all related to economics, relations "`olympic_medal`", "`olympics`", "`medal_won`" are all related to Olympics competitions. These results demonstrate that the disentangled representations learned by our DiCGRL are semantically meaningful.

5 Conclusion and Future Work

In this paper, we propose to study the problem of continual graph representation learning, aiming to handle the streaming nature of the emerging multi-relational data. To this end, we propose a disentangled-based continual graph representation learning (DiCGRL) framework, inspired by human's ability to learn procedural knowledge. Extensive experiments on several typical KGE and NE datasets show that DiCGRL achieves consistent and significant improvement compared to existing continual learning models, which verifies the effectiveness of our model on alleviating the catastrophic forgetting problem. In the future, we will explore to extend the idea of disentanglement in the continual learning of other NLP tasks.

Acknowledgments

This work is supported by NSFC under Grant No. 61532001, National Key Research and Development Program of China under Grant No. 2018AAA0101902, and MOE-ChinaMobile Program under Grant No. MCM20170503.

References

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. ACM.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

Antoine Bordes, Jason Weston, and Nicolas Usunier. 2014. Open question answering with weakly supervised embedding models. In *The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*. Springer.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. 2014. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations, CBLS, April 2014*.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. 2019. Efficient lifelong learning with a-gem. In *Proceedings of International Conference on Learning Representations*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Thirty-Second the Association for the Advance of Artificial Intelligence Conference*.

Xu Han, Yi Dai, Tianyu Gao, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2020. Continual relation learning via episodic memory activation and reconsolidation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6429–6440.

Mikael Henaff, Joan Bruna, and Yann LeCun. 2015. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S Yu. 2020. A survey on knowledge graphs: Representation, acquisition and applications. In *the Association for the Advance of Artificial Intelligence Conference*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of National Academy of Sciences of the United States of America*, pages 3521–3526.

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*, 6(2):167–195.

Xialei Liu, Marc Masana, Luis Herranz, Joost Van de Weijer, Antonio M Lopez, and Andrew D Bagdanov. 2018. Rotate your networks: Better weight consolidation and less catastrophic forgetting. In *Proceedings of International Conference on Pattern Recognition*, pages 2262–2268.

David Lopez-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. In *Proceedings of Conference and Workshop on Neural Information Processing Systems*, pages 6467–6476.

Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. Opendialkg: Explainable conversational reasoning with attention-based walks over knowledge graphs. In *Proceedings of the 57th Conference of The Association for Computational Linguistics*, pages 845–854.

Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2018. A novel embedding model for knowledge base completion based on convolutional neural network. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 327–333, New Orleans, Louisiana. Association for Computational Linguistics.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. 2019. Continual lifelong learning with neural networks: A review. *Neural Networks*, pages 54–71.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 701–710, New York, NY, USA. Association for Computing Machinery.

Hippolyt Ritter, Aleksandar Botev, and David Barber. 2018. Online structured laplace approximations for overcoming catastrophic forgetting. In *Proceedings of Conference and Workshop on Neural Information Processing Systems*, pages 3738–3748.

Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao Yang. 2018. Dynamic graph representation learning via self-attention networks. *arXiv preprint arXiv:1812.09430*.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI magazine*, 29(3):93–93.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. 2017. Continual learning with deep generative replay. In *Proceedings of Advances in neural information processing systems*, pages 2990–2999.

Robert L Solso, M Kimberly MacLin, and Otto H MacLin. 2005. *Cognitive psychology*. Pearson Education New Zealand.

Hyun-Je Song and Seong-Bae Park. 2018. Enriching translation-based knowledge graph embeddings through continual learning. *Institute of Electrical and Electronics Engineers Access*, 6:60489–60497.

Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. 2020. Lamal: Language modeling is all you need for lifelong language learning. In *International Conference on Learning Representations*.

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

Zhiqing Sun, Shikhar Vashishth, Soumya Sanyal, Partha Talukdar, and Yiming Yang. 2019b. A re-evaluation of knowledge graph completion methods. *arXiv preprint arXiv:1911.03903*.

Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015a. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, pages 1067–1077.

Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015b. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, page 1067–1077, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

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 *International Conference on Machine Learning*.

Cunchao Tu, Weicheng Zhang, Zhiyuan Liu, and Maosong Sun. 2016. Max-margin deepwalk: Discriminative learning of network representation. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 3889–3895. the Association for the Advance of Artificial Intelligence Press.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *International Conference on Learning Representations*. Accepted as poster.

Hong Wang, Wenhan Xiong, Mo Yu, Xiaoxiao Guo, Shiyu Chang, and William Yang Wang. 2019. Sentence embedding alignment for lifelong relation extraction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 796–806, Minneapolis, Minnesota. Association for Computational Linguistics.

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 *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual learning through synaptic intelligence. In *Proceedings of International Conference on Machine Learning*, pages 3987–3995.

A Results under Non-Continual Learning Settings

Results under non-continual learning settings, i.e., using the entire training sets to train the models, are presented in Table 3 and Table 4. For a fair comparison, we also reproduce the results of baselines, and using the same optimal hyper-parameters as in Section Experimental Settings in the main paper. DiCGRL (T) and DiCGRL (C) indicate using ConvKB and TransE as feature extraction method for DiCGRL respectively. DiCGRL $(\cdot)\_\alpha 1$ represents that α^k is calculated by performing a non-linearity transformation over the concatenation of $\boldsymbol{u}$ and $\boldsymbol{v}$ as doing in Network Embedding, and DiCGRL $(\cdot)\_\alpha 2$ represents that α^k is calculated by performing a non-linearity transformation over the concatenation of $\boldsymbol{u}$, $\boldsymbol{r}$ and $\boldsymbol{v}$, since relations are an integral part of KGs.

From the tables, we can see that:

(1) DiCGRL is comparable with our reproduced baselines, especially on the WN18RR with ConvKB as feature extraction method and FB15k-237 with TransE as feature extraction method, our DiCGRL performs better than vanilla GE methods. This phenomenon shows the effectiveness of our disentangled approach by decoupling the relational triplets in the graph into multiple independent components according to their semantic aspects.

Model	WN18RR		FB15k-237	
	MRR	H@10	MRR	H@10
TransE	22.6	50.1	29.4	46.5
TransE*	24.0	**51.1**	30.5	48.9
DiCGRL (T)_α1	17.3	40.8	29.0	46.8
DiCGRL (T)_α2	23.2	50.2	29.8	46.6
DiCGRL (T)	**24.2**	50.4	**30.9**	**49.6**
ConvKB	24.9	52.4	**24.3**	**42.1**
ConvKB*	40.4	53.6	22.5	41.3
DiCGRL (C)_α1	38.6	50.6	20.2	38.6
DiCGRL (C)_α2	39.2	49.4	20.7	38.2
DiCGRL (C)	**45.2**	**52.9**	23.9	41.5

Table 3: Link prediction results on whole WN18RR and FB15k-237. The best score is in **bold**. Results of TransE and ConvKB are taken from (Nguyen et al., 2018) and (Sun et al., 2019b) respectively. * indicates reproduced by us.

	Cora	Citeseer	Pubmed
GATs	87.0	74.7	85.6
DiCGRL	**88.1**	**75.1**	**85.9**

Table 4: Node classification results on three whole information networks. The best score is in **bold**.

(2) As shown in Table 3, the performance of DiCGRL $(\cdot)\_\alpha 1$ and DiCGRL $(\cdot)\_\alpha 2$ are worse than DiCGRL, even worse than original baseline in some settings. This indicates that assigning global attention values for each relation as done in DiCGRL is an optimal option for KG datasets.

B PubMed Result

The results of DiCGRL on the PubMed data is shown in Figure 9.

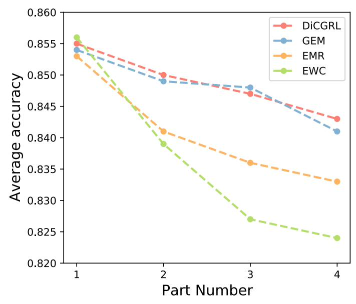


Figure 9: Changes in accuracy with increasing PubMed data through the continual learning process.

C Dataset Statistics

The statistics of FB15k-237 and WN18RR are presented in Table 5, where "Pi" denotes the i-th part, "# Accumulated Entities" and "# Accumulated Relations" represent the cumulative entities and rela-

Datasets	**FB15k-237**					**WN18RR**				
	P1	P2	P3	P4	P5	P1	P2	P3	P4	P5
# Entities	11,632	727	727	727	728	32,754	2047	2047	2047	2048
# Relations	236	236	236	236	237	11	11	11	11	11
# Train Set	178,274	20,347	23,317	26,325	23,852	54,570	7,442	8,727	8,195	7,901
# Validation Set	11,726	1,263	1,382	1,635	1,529	1,899	269	272	309	285
# Test Set	13,681	1,556	1,642	1,801	1,786	1,970	273	321	284	286
# Accumulated Entities	11,632	12,359	13,086	13,813	14,541	32,754	34,801	36,848	38,895	40,943
# Accumulated Relations	236	236	236	236	237	11	11	11	11	11

Table 5: Statistics of knowledge graph datasets.

Datasets	**Cora (# Class = 7)**				**Citeseer (# Class = 6)**				**PubMed (# Class = 3)**			
	P1	P2	P3	P4	P1	P2	P3	P4	P1	P2	P3	P4
# Nodes	1,895	271	271	271	2,328	333	333	333	13,801	1,972	1,972	1,972
# Edges	2,475	764	1,046	1,144	2,347	685	742	958	22,287	6,272	7,533	8,246
# Train Set	568	81	81	81	698	99	99	99	4,140	591	591	591
# Val Set	379	54	54	54	466	67	67	67	2,760	395	395	395
# Test Set	948	136	136	136	1,164	167	167	167	6,901	986	986	986
# Acc_Nodes	1,895	2,166	2,437	2,708	2,328	2,661	2,994	3,327	13,801	15,773	17,745	19,717
# Acc_Edges	2,475	3,239	4,285	5,429	2,347	3,032	3,774	4,732	22,287	28,559	36,092	44,338

Table 6: Statistics of three citation datasets.

tions after each new part of multi-relational data is generated. Statistics of Cora, Citeseer, and PubMed are presented in Table 6, where "# Acc_Nodes" and "# Acc_Edges" represent the cumulative nodes and edges after each new part of multi-relational data is generated.