PIE: A PARAMETER AND INFERENCE EFFICIENT SOLUTION FOR LARGE SCALE KNOWLEDGE GRAPH EMBEDDING REASONING

Linlin Chao, Xiexiong Lin, Taifeng Wang, Wei Chu
Ant Group
{chulin.cll, xiexiong.lxx, taifeng.wang, weichu.cw}@antgroup.com

May 6, 2022

ABSTRACT

Knowledge graph (KG) embedding methods which map entities and relations to unique embeddings in the KG have shown promising results on many reasoning tasks. However, the same embedding dimension for both dense entities and sparse entities will cause either over parameterization (sparse entities) or under fitting (dense entities). Normally, a large dimension is set to get better performance. Meanwhile, the inference time grows log-linearly with the number of entities for all entities are traversed and compared. Both the parameter and inference become challenges when working with huge amounts of entities. Thus, we propose PIE, a **p**arameter and **i**nference **e**fficient solution. Inspired from tensor decomposition methods, we find that decompose entity embedding matrix into low rank matrices can reduce more than half of the parameters while maintaining comparable performance. To accelerate model inference, we propose a self-supervised auxiliary task, which can be seen as fine-grained entity typing. By randomly masking and recovering entities' connected relations, the task learns the co-occurrence of entity and relations. Utilizing the fine grained typing, we can filter unrelated entities during inference and get targets with possibly sub-linear time requirement. Experiments on link prediction benchmarks demonstrate the proposed key capabilities. Moreover, we prove effectiveness of the proposed solution on the Open Graph Benchmark large scale challenge dataset WikiKG90Mv2 and achieve the state of the art performance.

1 Introduction

Knowledge graphs store structured data as the form of triples, where each triple (h, r, t) represents a relation r between a head entity h and a tail entity t. Typical projects such as WordNet Miller [1995], Freebase Bollacker et al. [2008], YAGO Suchanek et al. [2007] and DBpedia Lehmann et al. [2015], have gained widespread traction from their successful use in tasks such as question answering Bordes et al. [2014], Hamilton et al. [2018], information extractionHoffmann et al. [2011], recommender systems Zhang et al. [2016] and so on. Particularly, knowledge graph embedding (KGE) methods, which embed all entities and relations into a low dimension space, play a key role for these downstream tasks. It has been an active research area for the past couple of years. Methods such as TransE Bordes et al. [2013], ComplExTrouillon et al. [2016], RotatESun et al. [2019] provide a way to perform reasoning in knowledge graphs with simple numerical computation in continuous spaces. Although widely used in many reasoning tasks, these methods have to respond to several challenges when working with large scale KGs.

Parameter redundancy. KGE methods assign the same embedding dimension for all the entitiesWang et al. [2017]. However, entities in KGs naturally have long-tailed distributions, where different entities have different degrees (see Figure 1). The same embedding dimension for both dense entities and sparse entities will cause either over parameterization (sparse entities) or under fitting (dense entities). On small conventional benchmark datasets, such as FB15kBordes et al. [2013], YAGO3-10Dettmers et al. [2018], embedding dimensions for the state-of-the-art models can be as many as thousands Sun et al. [2019], Ebisu and Ichise [2018]. Furthermore, Hu et al. [2020], Chao et al. [2021] show on some dataset larger dimensions can achieve better performance. Given that there are many sparse entities, especially for large scale KGs, the redundancy in the parameterization of entity embeddings will be a severe problem.

Meanwhile, the parameter issue has been a challenge for huge KGs. Since the parameters of knowledge graph embedding methods grow linearly with the number of entities and embedding dimension, increasing the dimension for huge KGs can lead to the explosion of the parameters rapidly. With more parameters, the models consume more memory, storage and computing resources. This motivates us to reduce parameter redundancy and improve parameter efficiency to alleviate this challenge.

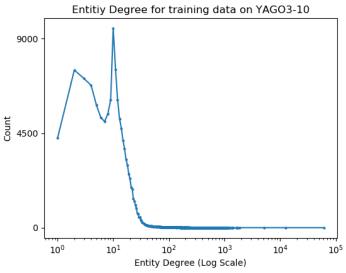

Figure 1: Entity degree distribution

Inefficiency inference. The inference efficiency, which has long been neglected, is also an inevitable challenge. Take the one-hop question answering reasoning task as an example. Given a test query $(?, PresidentOf, UnitedStates)$, all the entities in KG should be traversed and sorted to find the target entities. The time complexity is $O(n + n\log(n))$, where n is the size of entity set. Such a high complexity makes it difficult to scale up to large scale KGs, such as WikiKG90Mv2Hu et al. [2021]. One way to accelerate inference is only selecting and comparing relevant entities based on the query relation. For example, according to $PresidentOf$, we can infer the type of answers is person, or more precisely, politician. However, there are two obstacles to getting the wanted typing. The first one is KGs are incomplete, so as the entity type Zhao et al. [2020]. The other one is that the types are often several discrete classes Yogatama et al. [2015], Zhou et al. such as "person" and "Location". We argue these coarse typing might not fine grained enough to improve inference efficiency.

To solve these two challenges, we propose PIE, a **P**arameter and **I**nference **E**fficient solution. In the field of KGE, one group of works utilize latent factorization methods Nickel et al. [2011], Trouillon et al. [2016], Amin et al. [2020]. Taking triples in KG as the form of a partially observed 3D tensor, they decompose it into a product of embedding matrices with much smaller rank, resulting in fixed-dimensional vector representations for each entity and relation. Following this line of works, we propose low rank entity embedding (LRE). Rather than decomposing the observed 3D tensor directly, LRE decomposes the entity embedding matrix to low rank matrices. Extensive experiments show adding LRE to existing KGE methods can reduce more than half of the parameters while maintain comparable performance. When the computing resource is limited, the LRE based methods show significant advantages. To accelerate model inference, we propose an auxiliary self-supervised task, which can be seen as fine grained entity typing. Different with traditional entity typing, which defines entity types to discrete classes Moon et al. [2017], Zhao et al. [2020], we take the one-hop relation neighbor distribution (observed and unobserved) of entities as their types. This task learns the co-occurrence of entities and relations by randomly masking and recovering entities relation neighbors. Under the proposed task, we explore an inductive, entity independent and storage efficient model. Specifically, a relational message passing based graph neural network Wang et al. [2021a] is built to model entities' relation subgraphs to infer unobserved relations. With the learned fine grained entity typing, we can filter unrelated entities during inference process and get targets with possibly sub-linear time requirement.

Our key contributions are outlined as follows:

- We propose utilizing tensor factorization to improve parameter efficiency of existing KGE methods, which is general and efficient;
- We propose a new perspective for entity typing, which is self-supervised and fine grained. It can be used to improve inference efficiency seamlessly. Under the proposed task, we explore an effective relational message passing based model;
- On real-world extremely large benchmark, WikiKG90Mv2, the proposed solution improves both parameter and inference efficiency for existing KGE models and achieves state-of-the-art performance under limited computing resources.

2 Related Work

Knowledge graph embedding methods. Given a set of entities E and relations R in a KG, KGE is learned by assigning a score s to a triple (e_s, r, e_o):

$$s = f_r(e_s, e_o),$$

where $e_s, e_o \in E$ are the entities, and $r \in R$ is the relation between them. The scoring function f estimates the general binary tensor $T \in |E| \times |R| \times |E|$, by assigning a score of 1 to T_{ijk} if relation r_j exists between entities e_s and e_o, 0 otherwise.

Based on the scoring function, previous works can be roughly classified into tensor factorization models and distance based models. Tensor factorization models use different factorization methods to decompose tensor T. RESCAL

Nickel et al. [2011] and DistMult Yang et al. [2014] factorize its slices in the relation dimension. Others such as ComplEx Trouillon et al. [2016], SimplE Kazemi and Poole [2018] use Canonical Polyadic decomposition Hitchcock [1927] to factorize this 3D binary tensor directly. Distance based methods Bordes et al. [2013], Wang et al. [2014], Sun et al. [2019], Tang et al. [2019] use additive dissimilarity scoring functions, whereby they all map entities to vector representations. All these methods do not consider the redundancy in embedding matrices, especially for entity embedding, which may be problematic for large scale KGs.

Compression and parameter efficient models. Inspired by the distillation technique Hinton et al. [2015], Sanh et al. [2019], Sachan [2020] compress the trained entity matrices into discrete codes. Wang et al. [2021b] also utilizes multiple KGE models as teachers to distill a student model, which has lower dimension. One limitation of these works are that one or more well trained full embedding matrix is needed before distillation. NodePiece Galkin et al. [2021] selects some of the entities as anchor entities and represents all the entities based on these anchor entities' embedding. However, it needs compute the shortest paths from all the entities to anchor entities, which is time and resource consuming. For KG with 1M+ entities, the preprocessing step might be a computational bottleneck [1]. Contrary to those works, the proposed method learns embeddings more efficiently.

Entity typing. This task aims at inferring possible missing entity type instances in KG. One way to get the type information is to infer from external text information Xu and Barbosa [2018], which might not be feasible for some KGs. Others utilize embedding based models. The embedding models Moon et al. [2017], Zhao et al. [2020] combine triple knowledge and entity type instances for type prediction, which require entity type instance as supervised signal. Meanwhile, the types defined in these methods is often a small set of coarse labels. Instead, our proposed self-supervised fine grained entity typing task can overcome these limitations, providing a new perspective to this task. Besides, the proposed inductive relational message passing model, which aggregates the neighborhood relations, can also be a useful supplement to existing entity typing models.

Relation prediction. A recent line of works, such as GraILTeru et al. [2020] and PathConWang et al. [2021a], utilize entity-pair based subgraph and paths to predict the relation between two entities. We want to emphasize these works are different to our proposed fine grained entity typing task, where they belong to link prediction and ours are node property prediction. Besides, the fine grained entity typing is much more challenge. Because the target space for entity pair based relation prediction is certain. Due to the incomplete character of KGs, the label set can be very ambiguous for the proposed fine grained entity typing.

3 Methodology

3.1 Parameter Efficient: Low Rank Entity Embedding

Conventional KGE methods assign an unique vector for each entity. Suppose the dimension of entity embedding is d and the size of entity set is $|E|$. The total size of the entity embedding matrix is $d * |N|$, which is growing linearly with entity dimension and size of entity set. Normally, assigning the same large dimension vectors for both dense entities and sparse entities will lead to parameter redundancy. To this end, we propose LRE, which decompose entity embedding matrix into two low rank matrices. As shown in Figure 2, the entity embedding matrix Z is decomposed into matrices Z_d and W,

$$Z = Z_d * W, \tag{1}$$

where $Z_d \in \mathcal{R}^{|E| \times r}$, $W \in \mathcal{R}^{r \times d}$ and $r < d$. r is the rank of the proposed LRE. The matrix Z_d is entity dependent and matrix W is shared for all entities. The total number of entity parameters for LRE based KGE models can be $|E| * r + r * d$. Compared to the corresponding full entity embedding matrix based model, LRE can reduce parameters close to $|E| * (d - r)$, where the shared parameter W can be neglected for large entity set. The larger the entity set, the more efficiency of LRE based model.

Although the total parameter is reduced, LRE based model has the potential to recover the performance of the larger models. In LRE, the matrix W is shared for all entities. During training, the gradients of W are biased to denser entities as the imbalanced distribution of entity degrees. The biased learning ensures the performance of denser entities to some extent. While sparse entity can also learn from denser entity through the shared matrix W. As experiment results show with the same experiment settings, LRE can get comparable or even better performance than the 2x larger models.

LRE can be used to both tensor decomposition based models and distance based models. When combined with tensor decomposition based models, it can be seen as the further decomposition of the target 3D binary tensor T (see Section 2). When getting the entity embedding, all the subsequent steps keep the same with existing KGE methods.

[1] As the author states, it takes 55 hours to finish the preprocess step for ogb-wiki2 (2.5M nodes and 16M edges). https://github.com/migalkin/NodePiece/issues/1

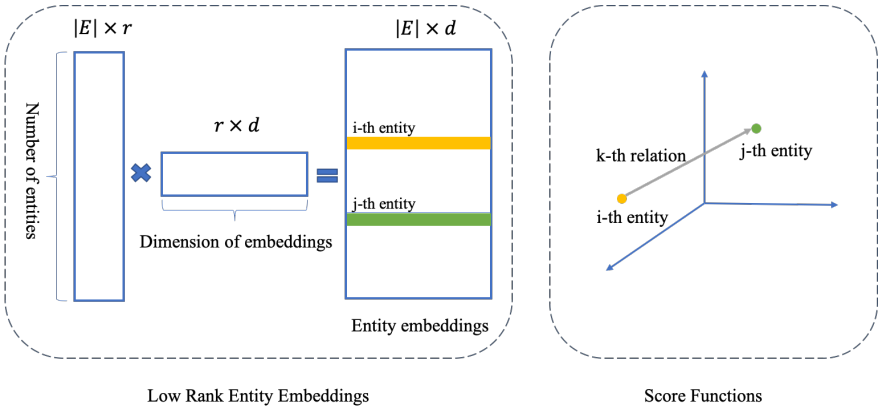

Figure 2: Low rank entity embedding for KGE methods.

Connection to NodePiece. Both LRE and NodePiece can be seen utilizing base vocabulary to represent all entities. NodePiece randomly selects some anchor entities as vocabulary while LRE uses the shared matrix W. One of the major differences is how to combine vocabulary to represent entities. NodePiece utilizes shortest path based encoding way, which is inductive but not friendly for large scale KGsGalkin et al. [2021]. LRE utilizes a low rank learnable look-up table, which is more efficient.

3.2 Inference Efficient: Fine Grained Typing Aware Inference

We propose to utilize neighborhood and entity type aware inference. Given test triple $(h, r, ?)$, we first extract neighborhoods of entity h. Based on query relation r we can also infer the type of answer entities. Then, we can select some candidates based on the inferred entity type. At last, the KGE models are utilized to rank all the candidates to get the answers. One of the key steps is to get the candidates based on entity type. However, KG is incomplete, including entity type. Besides, the existing entity typing may be too coarse to support efficient inference. Thus, a fine-grained entity typing is necessary.

Definition 1 (**Fine Grained Entity Typing**). *Given a knowledge graph with entity set E and relation set R, $\forall e \in E$, we call the distribution $p(r|e)$, $r \in R$, which measures the likelihood of relation types given a particular entity, as fine grained entity typing of entity e.*

Once we get the fine grained entity typing, we can select candidates based on the posterior distribution, which is calculated by

$$p(e|r) = \frac{p(e)p(r|e)}{p(r)} \propto p(e)p(r|e), e \in N(e_q), \tag{2}$$

where $N(e_q)$ is the neighborhood entities of query entity e_q. $p(e)$ and $p(r)$ are prior distributions over entities and relations respectively, which can be calculated based on their degrees.

Self-supervised entity typing model. To learn entity typing, we propose a self supervised task. Given an entity in KG, we randomly mask one of the one-hop relations and recover it by reasoning from the masked KG. The key idea is that the missing relations can be inferred from neighborhood relations and entities. *Note that the observed triples in KG can only provide partial information of the fine grained entity typing. The challenge is to infer the unobserved relation types for a particular entity.* Apart from structure information, node attributes also provide important clues. In order to generalize KG without node attributes, we only focus on structure information in this paper.

To scale up to large scale KGs efficiently, we build a lightweight inductive model. The model is built around the Graph Neural Network (GNN) Scarselli et al. [2008], Bronstein et al. [2017]. Two modules are required: (i) relational subgraph extraction, (ii) reasoning on the relational subgraph. Example of fine grained entity typing based on relation subgraph is shown in Figure 3.

Relational subgraph extraction. We assume that local graph neighborhood of a particular entity will contain the evidence needed to deduce the fine grained entity typing. As neighborhood triples with different relation types are more important than triples with the same relation, we try to keep all the relation types when we have to sample the neighbors. We also add the reverse triples with new relation types to the subgraph because edge directions for entity relation pair can have totally different semantics.

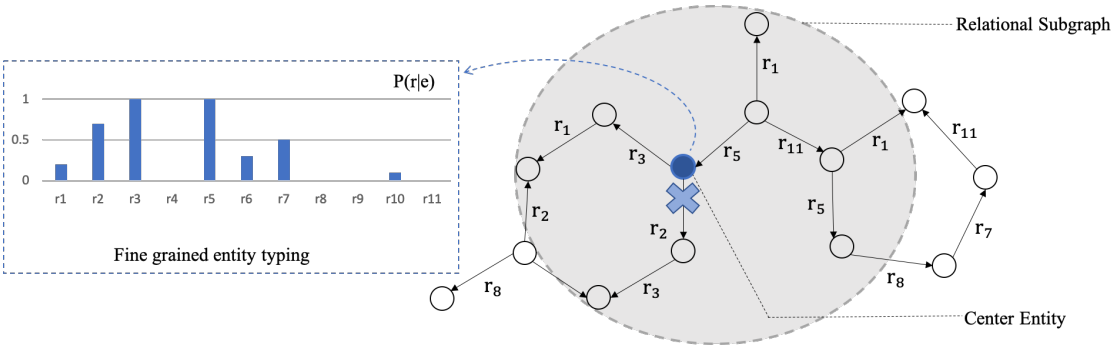

Figure 3: Example of fine grained entity typing based on relational subgraph. We randomly mask one of the relation types during model learning.

Reasoning on the relational subgraph. We adopt the alternate relational message passing framework described in Wang et al. [2021a], where node and edge representations are iteratively updated by combining their selves with aggregation of their neighbor edges' or neighbor nodes' representations. In particular, define s_e^i as the hidden state of edge e in iteration i, and m_v^i as the message stored at node v in iteration i. The representation of each edge is learned by:

$$m_v^i = \sum_{e \in \mathcal{N}(v)} s_e^i, \tag{3}$$

$$s_e^{i+1} = \sigma([m_v^i, m_u^i, s_e^i] \cdot W^i + b^i), v, u \in \mathcal{N}(e), \tag{4}$$

where $[\cdot]$ is the concatenation function, W^i, b^i, and $\sigma(\cdot)$ are the learnable transformation matrix, bias and nonlinear activation function respectively. In our implementation, σ is the $ReLU$ function. $s_e^0 = x_e$ is initial feature of edge e, which is the one hot identity vector of the relation type that e belongs to. The message passing in Equation. 3 and 4 are repeated for K times. Inspired by the JK-connection mechanism Xu et al. [2018], we use the representations from all the intermittent layers and the last layer, while the final representation for the target node t is represented by

$$m_t = [m_t^0, m_t^1, ..., m_t^{K-1}]. \tag{5}$$

Loss function. During learning process, the model randomly masks one relation and recover it by reasoning on the subgraph. Taking this task as a multi-classification problem, where the softmax function and cross entropy loss are utilized to maximize probability of the masked relation as other self-supervised models Devlin et al. [2018], is one choice. However, KG is incomplete and the unobserved relation types except the masked relation can also be the correct labels. With larger relation set or more missing triples, the incompleteness brings new challenge for this task. Besides, we hope the model outputs the distribution regard to all relation types rather than one specific type. Thus, we task this task as a multi-label learning task and utilize the following loss Sun et al. [2020], which contains the pair wised ranking between the observed relation types and the unobserved relations. The loss is represented as

$$L = \log[1 + \sum_{i \in \Omega_{obs}} \sum_{j \in \Omega_{uno}} \exp(\gamma(s^j - s^i + m))] = \log[1 + \sum_{j \in \Omega_{uno}} \exp(\gamma(s^j + m)) \sum_{i \in \Omega_{obs}} \exp(\gamma(-s^i))], \tag{6}$$

where Ω_{obs} and Ω_{uno} represent the observed type set and unobserved type set respectively and $|\Omega_{obs}| + |\Omega_{uno}| = |R|$. s represent the logits corresponding to each relation type. γ is a scale factor and m is a margin for better separation.

4 Experiment

We first comprehensively evaluate the effectiveness of the proposed two capabilities on relative small benchmarks (Section 4.1 and Section 4.2). Then we report out results on one of the largest benchmark WikiKG90Mv2 (Section 4.3).

Dataset. We choose link prediction as evaluation task. We believe our solution can also be generalize to other reasoning tasks, such as complex question answering on

Dataset	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{E}\rvert$	Train	Valid	Test
WikiKG90Mv2	1,387	91,230,610	601,062,811	15,000	15,000
ogbl-wikikg2	535	2,500,604	16,109,182	429,456	598,543
YAGO3-10	37	123,143	1,079,040	4,978	4,982
ogbl-biokg	51	93,773	4,762,678	162,886	162,870
FB15k	1,345	14,951	483,142	50,000	59,071
CoDEx-Large	69	77,951	551,193	30,622	30,622

Table 1: Number of entities, relations and observed triples in each split for the five utilized benchmarks.

incomplete KG Hamilton et al. [2018], Saxena et al. [2021]. The chosen benchmarks are listed in Table 1. Except wikiKG90Mv2, we choose ogbl-wikikg2 Hu et al. [2020], YAGO3-10 Dettmers et al. [2018], ogbl-biokg Hu et al. [2020] and CoDEx-LargeSafavi and Koutra [2020] for their entity sets are relative large. The FB15kBordes et al. [2013] is chosen for its large relation set, which is more difficult to infer fine grained entity typing.

Implementation. For Open Graph Benchmark (OGB) related datasets, ogbl-wikikg2, ogbl-biokg and WikiKG90Mv2[2], we utilize the official implementations. For the other two datasets, YAGO3-10 and FB15k, code from RotatE is utilized. Since part of the OGB official implementations are based on the repository of RotatE, all the codes share the same self-adversarial negative sampling loss function Sun et al. [2019]. Our code is public available[3].

4.1 Parameter Efficiency

Setup. We choose PairRE as baseline model. NodePiece, the entity hashing based efficient model, is also compared. The designed experiments are mainly to prove the parameter efficiency of LRE based models rather to outperform the best existing models.

Following the state-of-the-art methods, we measure the quality of the ranking of each test triple among all possible head entity and tail entity substitutions. Two evaluation metrics, including Mean Reciprocal Rank (MRR) and Hit ratio with cut-off value 10, are utilized. For experiments on ogbl-wikikg2 and ogbl-biokg, we follow the evaluation protocol of these two benchmarks Hu et al. [2020]. We also report the total parameters for all models and the required hardware for larger dataset.

-	ogbl-wikikg2						ogbl-biokg				
Model	#Dim	#LRE_Rank	Test MRR	Valid MRR	#Param(M)	GPU	#Dim	#LRE_Rank	Test MRR	Valid MRR	#Param(M)
AutoSFZhang et al. [2020]	200	-	0.5458	0.5510	500.2	45G	1000	-	0.8309	0.8317	93.8
AutoSF+NodePieceGalkin et al. [2021]	200	-	**0.5703**	**0.5806**	6.9	32G	-	-	-	-	-
PairREChao et al. [2021]	200	-	0.5289	0.5529	500.3	16G	2000	-	0.8164	0.8172	187.8
PairRE	400	-	0.5805	0.5529	1,000.7	32G	-	-	-	-	-
PairRE+LRE	700	200	0.5836	0.5861	501.0	16G	2000	800	0.8244	0.8250	76.6
PairRE+LRE	5000	200	**0.5970**	**0.6033**	506.5	24G	5000	2000	**0.8360**	**0.8367**	198.1

Table 2: Link prediction results on ogbl-wikikg2 and ogbl-biokg. Best results for each group are in bold. Each experiment is run ten times. We do not add the standard deviations for limited space.

-	YAGO3-10					CoDEx-Large				
Model	#Dim	#LRE_Rank	MRR	Hit@10	#Param(M)	#Dim	#LRE_Rank	MRR	Hit@10	#Param(M)
RotatESun et al. [2019]	500	-	**0.495**	**0.670**	123	500	-	**0.258**	**0.387**	77
RotatE+NodePieceGalkin et al. [2021]	500	-	0.247	0.488	4.1	500	-	0.190	0.313	3.6
PairREChao et al. [2021]	700	-	**0.540**	**0.693**	86.3	1000	-	**0.290**	**0.415**	78.1
PairRE+LRE	1000	30	0.284	0.472	3.8	1000	30	0.234	0.375	2.5

Table 3: Link prediction results on YAGO3-10 and CoDEx-Large.

Results. Comparisons for ogbl-wikikg2 and ogbl-biokg are shown in Table 2. All the models are in the same experiment settings and implementation. The results shows larger dimension can improve the Test MRR significantly on ogbl-wikikg2. While the LRE based models can boost the performance further with the same budget of parameters and computing resource. The same phenomenon can also be observed on ogbl-biokg. The LRE based models can improve test MRR with 6.9% and 1.96% respectively. Results for YAGO3-10 and CoDEx-Large are shown in Table 3. When compared to NodePiece based model, on ogbl-wikikg2, LRE based model consumes more parameters to outperform their performance. However, as stated before, LRE based models can avoid the long preprocess time of NodePiece based methods. On YAGO3-10 and CoDEx-Large, the LRE based model can outperform NodePiece based model with smaller budget of parameters (see Table 3). All these experiments show LRE based models is advantageous when meet with limited resource and large scale datasets.

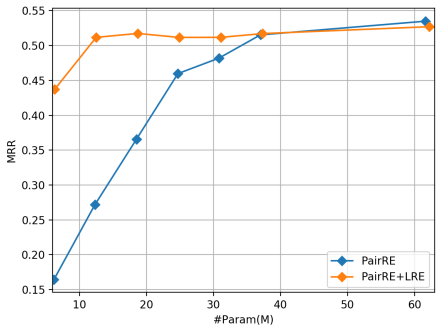

Figure 4: Parameter efficiency comparison on YAGO3-10 dataset.

[2]The official code for wikiKG90M is utilized, which is based on DGL-KEZheng et al. [2020]

[3]https://github.com/alipay/Parameter_Inference_Efficient_PIE

We further study the influence of the parameter size to the performance (Figure 4). On YAGO3-10, the performances of PairRE and PairRE+LRE increase with the increase of parameter size. The dimension of PairRE+LRE is set to 1000 and the LRE rank is increasing from 50 to 300. While the dimension of PairRE is increased from 50 to 500. The performance of PairRE+LRE saturates at 20M, while PairRE at 60M. When the size of parameter is less than 40M, PairRE+LRE shows much better performances. Particularly, LRE powered models can achieve comparable performances compared to 3x larger models (LRE+PairRE@12M VS PairRE@38M). This indicates the proposed LRE can improve the parameter efficiency for KGE methods.

4.2 Inference Efficiency

Setup. We first report the proposed entity typing model performances on YAGO3-10 and FB15k. The comparison between softmax based loss and the proposed ranking loss is added as we find loss function is a vital step in our experiments. When evaluating the entity typing model, we follow the data splits of the conventional link prediction task. For a given triple (h, r, t) in the test or valid set, we rank the ground truth relation type r against all other relation types based on the relation subgraph of entity h. We also use MRR and Hit@5 as evaluation metrics. The rankings are computed after removing all the other observed relation types that appear in training set.

-	YAGO3-10		FB15k	
Loss	MRR	Hit@5	MRR	Hit@5
Softmax	0.9765	**0.9972**	0.7005	0.8495
Ranking loss	**0.9898**	0.9971	**0.9017**	**0.9391**

Table 4: Evaluations for the entity typing model on YAGO3-10 and FB15k.

Then two baselines are compared to the fine grained typing model powered inference. We name the proposed inference method as **F**ine-grained entity **T**yping **A**ware **I**nference (FTAI). Baseline one is the conventional inference, which is scoring and ranking all the entities. Baseline two is the neighborhood based inference. The difference between FTAI and baseline two is that only entity degree is used to select candidates from neighborhood entities for baseline two. The second baseline can be seen as the ablation analysis for the proposed entity typing model. Two evaluation metrics are utilized, inference time and Hit@10.

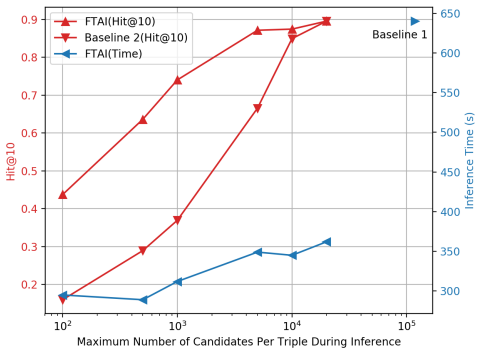

Figure 5: Influence of the candidate number per test triple of FTAI on test MRR and inference time on FB15k for the PairRE model. Baseline 1 is scoring all entities. Baseline 2 is the proposed FTAI without use entity typing result.

Results. Table 4 shows the evaluation results of the entity typing models. Clearly, the FB15k dataset is more challenge as there is much larger relation sets compared to YAGO3-10 (1,315 vs 37). The table also shows pronounced gap between softmax loss and the proposed ranking loss on FB15k. One of the reasons might be the incompleteness of FB15k is more severe. Under this situation, the ranking loss based model can largely outperform softmax based model.

Incorporate the entity typing model, the proposed FTAI shows clear advantages compared to these two baselines (See Figure 5). Compared to the conventional inference, FTAI can achieve 1.8x speedup and the same test Hit@10 performance when at most 20k candidates are selected for each test triple. Ablation analysis shows for both FTAI and baseline two, with larger candidate set, the better performance and the slower inference. The results also show the entity typing model enables a much large recall compared with baseline two, which proves the crucial function of the proposed entity typing model.

4.3 Results on WikiKG90Mv2

Setup. We follow the official evaluation of WikiKG90Mv2. That is, for each $(h, r, ?)$, the model is asked to predict the top 10 tail entities that are most likely to be positive. Since traversal all the 90 million entities is impractical during inference, the organizers of this dataset provide at most 20,000 tail candidates for each test triple. They pick tail candidates t by choosing the entities with at least one triple $(\_, r, t)$ on the KG, and sort all the tails by its degree. We take these candidates as baseline candidates. In accordance with baseline models, we also pick at most 20k candidates for FTAI.

Results. The reported results (Table 5) show competitive performances of PIE powered ComplEx and TransE models. We tune the baseline models carefully and get much better performances compared to the official results. With the same hyper-parameter settings and parameter budget, LRE can improve ComplEx and TransE 0.56% and 0.75% respectively. Utilizing the fine grained entity typing model, we can improve ComplEx and TransE 0.97% and 2.47% respectively. This is because the candidates predicted by our entity typing model have much higher recall. As Table 6 shows the

candidates output by FTAI have a much higher recall than the baseline candidates. When combine TransE and the proposed PIE, we achieve state-of-the-art performance, with 2.82% improvement compared to the entity attribute powered model.

Model	#Dim	#LRE_Rank	Test MRR	Valid MRR	#Param(B)
TransE-ConcatHu et al. [2021]◊	200	-	0.1761	0.2060	18.2
ComplExHu et al. [2021]	100	-	0.0985	0.1150	18.2
ComplEx(ours)	100	-	-	0.1795	18.2
+LRE	400	200	-	0.1851	18.2
+FTAI	100	-	-	0.1892	18.2
+LRE+FTAI	400	200	-	**0.1919**	18.2
TransEHu et al. [2021]	200	-	0.0824	0.1103	18.2
TransE(ours)	200	-	-	0.2049	18.2
+LRE	600	200	-	0.2124	18.2
+FTAI	200	-	-	0.2296	18.2
+LRE+FTAI	600	100	-	0.2247	9.1
+LRE+FTAI	600	200	**0.1883**	**0.2342**	18.2

Table 5: Link prediction results on WikiKG90Mv2.◊ means the entity attribute is utilized.

Ablation analysis result of the proposed FTAI is shown in Table 6. With the same candidate number, the recall of FTAI is higher than the baseline candidates with close to 10%, which proves the necessity and the effectiveness of the fine grained typing model. The evaluation metrics for entity typing model on this dataset is 65% for MRR and 72% for Hit@5. Compared to results on FB15k, there is still large room for further improvement. We leave this as future work.

Model	Recall@20k
BaselineHu et al. [2021]	0.5770
FTAI	**0.6695**

Table 6: Evaluations for entity typing model on WikiKG90Mv2 valid set.

5 Conclusion and Future Work

With the explosion of data, the emerging large scale KGs bring new challenges for KGE based reasoning. We explored the parameter redundancy and inefficiency inference. For parameter redundancy, we propose tensor factorization based method LRE, which decomposes the entity embedding matrix. Experiment results show LRE based model can approximate the performance of full rank entity matrix based models, which saves more than 50% parameters. For inference optimization, we proposed a fine grained entity typing model. Without extra annotation, we learn the concurrence distribution of entity and relation with a lightweight and inductive model. Utilizing the learned entity typing, the full entity set traversing is avoid during inference. Experiment results on FB15k shows we achieve a 1.8x speedup over the full entity set traversing inference. We also verify the proposed solution on large scale dataset WikiKG90Mv2 and achieve state of the art performance.

The deep models have shown tremendous successes Goodfellow et al. [2016]. Compared to models from computer vision He et al. [2016] and NLPDevlin et al. [2018], where the layer-wise hierarchy representations are widely used, KGE models are shallow. The proposed LRE can be seen as an exploration from shallow to deep, where the large look-up table is replaced with a smaller table plus a linear layer. For future work, we will try more "decomposition" for the look-up table with techniques such as residual connectionHe et al. [2016] and auxiliary losses in intermediate layersSzegedy et al. [2015]. The incompleteness of KGs also leads to challenge for the fine grained entity typing. A framework that unifies KG reasoning and fine grained entity typing should boot performances of these two tasks. We will leave these for future works.

References

George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995. doi:10.1145/219717.219748. URL `https://doi.org/10.1145/219717.219748`.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 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, 2008.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM, 2007. doi:10.1145/1242572.1242667. URL `https://doi.org/10.1145/1242572.1242667`.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. Dbpedia a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195, 2015.

Antoine Bordes, Jason Weston, and Nicolas Usunier. Open question answering with weakly supervised embedding models. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 165–180. Springer, 2014.

Will Hamilton, Payal Bajaj, Marinka Zitnik, Dan Jurafsky, and Jure Leskovec. Embedding logical queries on knowledge graphs. *Advances in neural information processing systems*, 31, 2018.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 541–550, 2011.

Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 353–362, 2016.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795, 2013. URL `https://proceedings.neurips.cc/paper/2013/file/1cecc7a77928ca8133fa24680a88d2f9-Paper.pdf`.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*, pages 2071–2080, 2016. URL `http://proceedings.mlr.press/v48/trouillon16.html`.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=HkgEQnRqYQ`.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743, 2017.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Takuma Ebisu and Ryutaro Ichise. Toruse: Knowledge graph embedding on a lie group. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

Linlin Chao, Jianshan He, Taifeng Wang, and Wei Chu. PairRE: Knowledge graph embeddings via paired relation vectors. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4360–4369, Online, August 2021. Association for Computational Linguistics. doi:10.18653/v1/2021.acl-long.336. URL `https://aclanthology.org/2021.acl-long.336`.

Weihua Hu, Matthias Fey, Hongyu Ren, Maho Nakata, Yuxiao Dong, and Jure Leskovec. Ogb-lsc: A large-scale challenge for machine learning on graphs. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

Yu Zhao, Anxiang Zhang, Ruobing Xie, Kang Liu, and Xiaojie Wang. Connecting embeddings for knowledge graph entity typing. *arXiv preprint arXiv:2007.10873*, 2020.

Dani Yogatama, Dan Gillick, and Nevena Lazic. Embedding methods for fine grained entity type classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 291–296, 2015.

Ben Zhou, Daniel Khashabi, Chen-Tse Tsai, and Dan Roth. Zero-shot open entity typing as type-compatible grounding.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *ICML*, volume 11, pages 809–816, 2011.

Saadullah Amin, Stalin Varanasi, Katherine Ann Dunfield, and Günter Neumann. Lowfer: Low-rank bilinear pooling for link prediction. In *International Conference on Machine Learning*, pages 257–268. PMLR, 2020.

Changsung Moon, Paul Jones, and Nagiza F Samatova. Learning entity type embeddings for knowledge graph completion. In *Proceedings of the 2017 ACM on conference on information and knowledge management*, pages 2215–2218, 2017.

Hongwei Wang, Hongyu Ren, and Jure Leskovec. Relational message passing for knowledge graph completion. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1697–1707, 2021a.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*, 2014.

Seyed Mehran Kazemi and David Poole. Simple embedding for link prediction in knowledge graphs. In *Advances in Neural Information Processing Systems*, pages 4284–4295, 2018.

Frank L Hitchcock. The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematics and Physics*, 6(1-4):164–189, 1927.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *Twenty-Eighth AAAI conference on artificial intelligence*, 2014.

Yun Tang, Jing Huang, Guangtao Wang, Xiaodong He, and Bowen Zhou. Orthogonal relation transforms with graph context modeling for knowledge graph embedding. *arXiv preprint arXiv:1911.04910*, 2019.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

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

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

Mikhail Galkin, Jiapeng Wu, Etienne Denis, and William L Hamilton. Nodepiece: Compositional and parameter-efficient representations of large knowledge graphs. *arXiv preprint arXiv:2106.12144*, 2021.

Peng Xu and Denilson Barbosa. Neural fine-grained entity type classification with hierarchy-aware loss. *arXiv preprint arXiv:1803.03378*, 2018.

Komal Teru, Etienne Denis, and Will Hamilton. Inductive relation prediction by subgraph reasoning. In *International Conference on Machine Learning*, pages 9448–9457. PMLR, 2020.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning*, pages 5453–5462. PMLR, 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. Circle loss: A unified perspective of pair similarity optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6398–6407, 2020.

Apoorv Saxena, Soumen Chakrabarti, and Partha Talukdar. Question answering over temporal knowledge graphs. *arXiv preprint arXiv:2106.01515*, 2021.

Tara Safavi and Danai Koutra. Codex: A comprehensive knowledge graph completion benchmark. *arXiv preprint arXiv:2009.07810*, 2020.

Da Zheng, Xiang Song, Chao Ma, Zeyuan Tan, Zihao Ye, Jin Dong, Hao Xiong, Zheng Zhang, and George Karypis. Dgl-ke: Training knowledge graph embeddings at scale. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 739–748, New York, NY, USA, 2020. Association for Computing Machinery.

Yongqi Zhang, Quanming Yao, Wenyuan Dai, and Lei Chen. Autosf: Searching scoring functions for knowledge graph embedding. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 433–444. IEEE, 2020.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.