Revisiting Simple Neural Networks for Learning Representations of Knowledge Graphs

Srinivas Ravishankar[*]
Department of Computer Science
R.V. College of Engineering, Bangalore
srini.shank@gmail.com

Chandrahas
Department of Computer Science and Automation
Indian Institute of Science, Bangalore
dewangan.chandrahas@gmail.com

Partha Pratim Talukdar
Department of Computational and Data Sciences
Indian Institute of Science, Bangalore
partha@talukdar.net

Abstract

We address the problem of learning vector representations for entities and relations in Knowledge Graphs (KGs) for Knowledge Base Completion (KBC). This problem has received significant attention in the past few years and multiple methods have been proposed. Most of the existing methods in the literature use a predefined characteristic scoring function for evaluating the correctness of KG triples. These scoring functions distinguish correct triples (high score) from incorrect ones (low score). However, their performance vary across different datasets. In this work, we demonstrate that a simple neural network based score function can consistently achieve near start-of-the-art performance on multiple datasets. We also quantitatively demonstrate biases in standard benchmark datasets, and highlight the need to perform evaluation spanning various datasets.

1 Introduction

Knowledge Graphs (KGs) such as NELL [7] and Freebase[1] are repositories of information stored as multi-relational graphs. They are used in many applications such as information extraction, question answering etc. oSuch KGs contain world knowledge in the form of relational triples (h, r, t), where entity h is connected to entity t using directed relation r. For example, *(DonaldTrump, PresidentOf, USA)* would indicate the fact that *Donald Trump* is the president of *USA*. Although current KGs are fairly large, containing millions of facts, they tend to be quite sparse [17]. To overcome this sparsity, Knowledge Base Completion (KBC) or Link Prediction is performed to infer missing facts from existing ones. Low dimensional vector representations of entities and relations, also called embeddings, have been extensively used for this problem [15; 9]. Most of these methods use a characteristic score function which distinguishes correct triples (high score) from incorrect triples (low score). Some of these methods and their scoring functions are summarized in Table 1.

WN18 and FB15k are two standard benchmarks datasets for evaluating link prediction over KGs. Previous research have shown that these two datasets suffer from *inverse relation bias* [3] (more in Section 4.2.1). Performance in these datasets is largely dependent on the model's ability to predict inverse relations, at the expense of other independent relations. In fact, a simple rule-based model exploiting such bias was shown to have achieved state-of-the-art performance in these two datasets

[*]Research carried out during an internship at the Indian Institute of Science, Bangalore.

6th Workshop on Automated Knowledge Base Construction (AKBC) at NIPS 2017, Long Beach, CA, USA.

TransE [2]	$-\|\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|\|$
HolE [9]	$(\mathbf{r}^T.(\mathbf{h} \star \mathbf{t}))$
DistMult [20]	$< \mathbf{h}, \mathbf{r}, \mathbf{t} >$
ComplEx [15]	$Re(< \mathbf{h}, \mathbf{r}, \bar{\mathbf{t}} >)$

Table 1: Score functions of some well known Knowledge Graph embedding models. Here **h**, **t**, **r** are vector embeddings for entities h, t and relation r, respectively. $\star$, $< \cdot, \cdot, \cdot >$ and Re represent circular correlation, sum of component-wise product, and real part of complex number, respectively.

[3]. To overcome this shortcoming, several variants, such as FB15k-237 [14], WN18RR [15], FB13 and WN11 [12] have been proposed in the literature.

ComplEx [15] and HolE [9] are two popular KG embedding techniques which achieve state-of-the-art performance on the WN18 and FB15k datasets. However, we observe that such methods do not perform as well uniformly across all the other datasets mentioned above. This may suggest that using a predefined scoring function, as in ComplEx and HolE, might not be the best option to achieve competitive results on all datasets. Ideally, we would prefer a model that achieves near state-of-the-art performance on any given dataset.

In this paper, we demonstrate that a simple neural network based score function that can adapt to different datasets and achieve near state-of-the-art performance on multiple datasets. The main contributions of this papers can be summarized as follows.

- We quantitatively demonstrate the severity of the inverse relation bias in the standard benchmark datasets.
- We empirically show that current state-of-the-art methods do not perform consistently well over different datasets.
- We demonstrate that ER-MLP[4], a simple neural network based scoring function, has the ability to adapt to different datasets achieving near state-of-the-art performance consistently. We also consider a variant, ER-MLP-2d.

Code is available at https://github.com/Srinivas-R/AKBC-2017-Paper-14.git.

2 Related Work

Several methods have been proposed for learning KG embeddings. They differ in the way entities and relations are modeled, the score function used for scoring triples, and the loss function used for training. For example, TransE [2] uses real vectors for representing both entities and relations, while RESCAL [10] uses real vectors for entities, and real matrices for relations.

Translational Models: One of the initial models for KG embeddings is TransE [2], which models relation r as translation vectors from head entity h to tail entity t for a given triple (h, r, t). A pair-wise ranking loss is then used for learning these embeddings. Following the basic idea of translation vectors in TransE, there have been many methods which improve the performance. Some of these methods are TransH [16], TransR [6], TransA [18], TransG [19] etc.

Multiplicative Models: HolE [9] and ComplEx [15] are recent methods which achieve state-of-the-art performance in link prediction in commonly used datasets FB15k and WN18. HolE models entities and relations as real vectors and can handle asymmetric relations. ComplEx uses complex vectors and can handle symmetric, asymmetric as well as anti-symmetric relations. We use these methods as representatives of the state-of-the-art in our experiments.

Neural Models: Several methods that use neural networks for scoring triples have been proposed. Notable among them are NTN [12], CONV [14], ConvE [3], and R-GCN [11]. CONV uses the internal structure of textual relations as input to a Convolutional Neural Network. NTN learns a tensor net [12] for each relation in the knowledge graph. ConvE uses convolutional neural networks (CNNs) over reshaped input vectors for scoring triples. R-GCN takes a different approach and uses Graph Convolutional Networks to obtain embeddings from the graph. DistMult (or some other linear model) is then used on these embeddings to obtain a score. We focus on simple neural models, such as ER-MLP [11], and find that such simple models are more effective in KG embedding and link prediction than more complicated models such as ConvE or R-GCN.

3 Knowledge Graph Embedding using Simple Neural Networks

Rather than using a predefined function to score triples, and then learn embeddings to fit this scoring function, we use a neural network to *jointly* learn both the scoring function and embeddings together to fit the dataset.

3.1 Neural Network as a Score Function

We use a simple feed-forward Neural Network with a single hidden layer as the approximator of the scoring function of a given triple. In particular, we use ER-MLP [11], a previously proposed neural network model for KG embedding, and ER-MLP-2d, a variant of ER-MLP we propose. Architectures of the two models are shown in Figure 1.

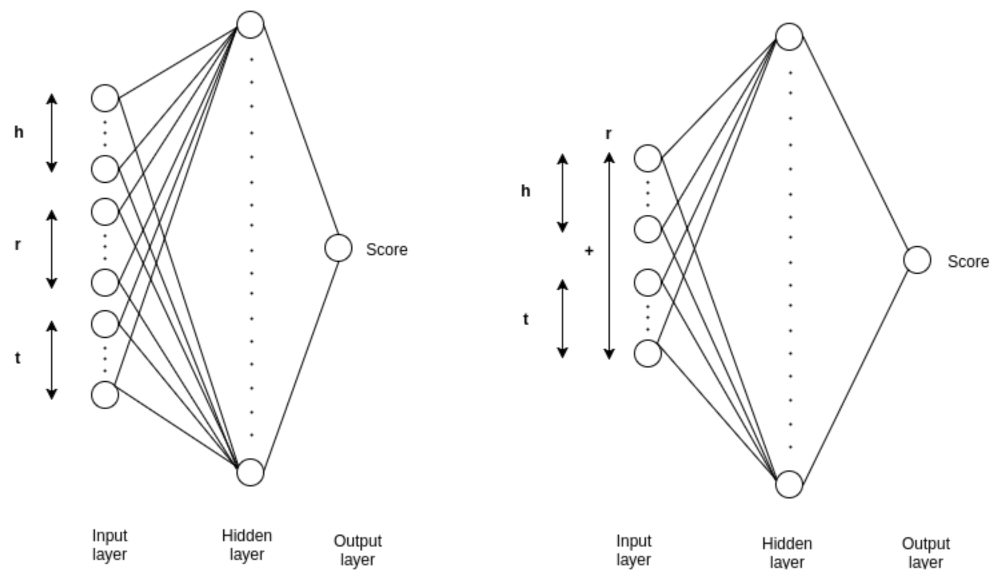

Figure 1: Architecture of (a) ER-MLP, (b) ER-MLP-2d

Let $\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$ be the d-dimensional embeddings of the entities h and t respectively[2]. Similarly, $\mathbf{r}$ is the embedding of relation r, whose dimensions are d and $2d$ in ER-MLP and ER-MLP-2d, respectively, as we shall explain below. In ER-MLP, the head, relation and tail embeddings are concatenated and fed as input to the NN, so its input layer is of size $3d$. In ER-MLP-2d, the concatenated head and tail embeddings are translated using the relation embedding of size $2d$, so input size in ER-MLP-2d is $2d$. Both models have a single fully connected hidden layer. This leads to an output node, which is taken as the score of the given triple (h, r, t).

Let $g()$ denote the activation function and $[\mathbf{a}, \mathbf{b}]$ denote concatenation of vectors $\mathbf{a}$ and $\mathbf{b}$. Let M_1 and A_1 be respectively the hidden and output layer weight matrices in ER-MLP. b_1 is a single bias value. Let the equivalent parameters in ER-MLP-2d be M_2, A_2, and b_2. The triple scoring functions for ER-MLP and ER-MLP-2d respectively are given below.

$$\begin{aligned} f_{\mathrm{ER-MLP}}(\mathbf{h}, \mathbf{r}, \mathbf{t}) &= (A_1 * g(M_1 * [\mathbf{h}, \mathbf{r}, \mathbf{t}])) + b_1 \\ f_{\mathrm{ER-MLP-2d}}(\mathbf{h}, \mathbf{r}, \mathbf{t}) &= (A_2 * g(M_2 * ([\mathbf{h}, \mathbf{t}] + \mathbf{r}))) + b_2 \end{aligned}$$

[2]We use boldface to refer to embeddings of corresponding italicized objects

Model	Number of Parameters	WN18	FB15K
HolE	$N_e d + N_r d$	4.096×10^6	1.629×10^6
ComplEx	$2N_e d + 2N_r d$	8.192×10^6	3.259×10^6
ConvE	$N_e d + N_r d$	4.096×10^6	1.629×10^6
ER-MLP	$N_e d + N_r d + 30d^2 + 10d$	4.397×10^6	1.83×10^6
ER-MLP-2d	$N_e d + 2N_r d + 20d^2 + 10d$	4.298×10^6	1.965×10^6

Table 2: Number of parameters of various methods over the WN18 and FB15K datasets. Above, d represents entity embedding size, N_e is the number of entities, and N_r is the number of relations.

	WN18	FB15K	WN18RR	FB15K-237	WN11	FB13
Number of Relations	18	1345	11	237	11	13
Percentage of Trivial Test Triples	72.12%	54.42%	0%	0%	0%	0%

Table 3: Inverse Relation Bias present in various datasets. Please see Section 4.2.1 for more details.

We consider the sigmoid function, $\sigma(f(\mathbf{h}, \mathbf{r}, \mathbf{t}))$, to be the probability of correctness of a triple. We train the model to assign probability of 1 to correct triples and 0 to incorrect triples. Let $\mathcal{T} = \{((h, r, t), y)\}$ be the set of positive and (sampled) negative triples, with label $y \in \{1, 0\}$. We optimize the cross-entropy loss given below, with f replaced by $f_{\mathrm{ER-MLP}}$ and $f_{\mathrm{ER-MLP-2d}}$ for ER-MLP and ER-MLP-2d, respectively.

$$Loss \;=\; -\sum_{((h,r,t),y)\in\mathcal{T}} [y \times \log(\sigma(f(\mathbf{h}, \mathbf{r}, \mathbf{t}))) + (1 - y) \times \log(1 - \sigma(f(\mathbf{h}, \mathbf{r}, \mathbf{t})))]$$

4 Experimental Setup

4.1 Implementation Details

For the neural network-based models, we used Dropout [13] with p = 0.5 on the hidden layer to prevent overfitting. The regularization parameter for weight decay was chosen from {0.001, 0.01, 0.1} based on cross validation. We chose the hidden layer size between {10d, 20d}. Since the size of this layer determines the expressive power of the model, datasets with simple relations (lower relation-specific indigree) require smaller number of hidden units and more difficult datasets require a higher number to achieve optimal performance.

ReLU [8] activation is used in the hidden layer to achieve fast convergence. To minimize the objective function, we used ADAM [5] with learning rate 0.001. Dimensionality of entity and relation embeddings were set equal in ER-MLP, i.e., $d_e = d_r = d$. For ER-MLP-2d, we have $2d_e = d_r = 2d$. d was cross validated on {100, 200} for all datasets. All experiments were run using Tensorflow on a single GTX 1080 GPU. To achieve maximum GPU utilization, we set the batch size larger than used previously in literature, choosing from {10000, 20000, 50000} using cross validation. For sampling negative triples, we used *bernoulli* method as described in [2].

4.2 Datasets

We ran experiments on the datasets listed in Table 3. Previous work has noted that the two benchmark datasets – FB15K and WN18 – have a high number of redundant and reversible relations [14]. A simple rule-based model, exploiting such deficiencies, was shown to have achieved state-of-the-art performance in these datasets [3]. This suggests that evaluation restricted only to these two datasets may not be an accurate indication of the model's capability. In order to address this issue, we evaluate model performance over six datasets, as summarized in Table 3.

4.2.1 Inverse Relation Bias

In a knowledge graph, a pair of relations r and r' are said to be inverse relations if a correct triple (h,r,t) implies the existence of another correct triple (t,r',h), and vice versa. A *trivial triple* refers to the existence of a triple (h, r, t) in the *test* dataset when (t, r', h) is already present in the *training*

	WN18			FB15K		
	Hits@10	MR	MRR	Hits@10	MR	MRR
HolE	94.12	810	0.934	84.35	113	0.64
ComplEx	94.64	826	0.938	**87.33**	113	**0.75**
ConvE	95.5	504	**0.942**	87.3	**64**	0.745
R-GCN	**96.4**	-	0.814	84.2	-	0.696
ER-MLP	94.2	**299**	0.895	80.14	81	0.57
ER-MLP-2d	93.66	372	0.893	80.04	81	0.567

	FB15K-237			WN18RR		
	Hits@10	MR	MRR	Hits@10	MR	MRR
HolE	47.0	501	0.298	42.4	6129	0.395
ComplEx	50.7	381	0.326	**50.7**	5261	**0.444**
ConvE	45.8	330	0.301	41.1	7323	0.342
R-GCN	41.7	-	0.248	-	-	-
ER-MLP	54.0	**219**	**0.342**	41.92	4798	0.366
ER-MLP-2d	**54.65**	234	0.338	42.1	**4233**	0.358

	FB13			WN11		
	Hits@10	MR	MRR	Hits@10	MR	MRR
HolE	54.87	1436	0.392	10.48	10182	0.059
ComplEx	51.38	6816	0.382	10.9	11134	0.071
ER-MLP	**63.13**	**705**	**0.479**	**14.01**	4660	0.071
ER-MLP-2d	62.66	821	0.476	13.26	**4290**	**0.073**

Table 4: Link prediction performance of various methods on different datasets. Available results for ConvE and R-GCN are taken from [3]. Please see Section 5 for more details.

dataset, with r and r' being inverse relations. A model that can learn inverse relations well at the expense of other types of relations will still achieve very good performance on datasets involving such biased relations. This is undesirable, since our goal is to learn effective embeddings of highly multi-relational graphs.

We quantitatively investigated the bias of various datasets towards such inverse relations by measuring the fraction of trivial triples present in them. The results are summarized in Table 3. Using the training dataset, each pair of relations were tested for inversion. They were identified as inverses if 80% or more triples that contained one relation appeared as inverse triple involving the other relation. As can be seen from the table, the two standard benchmark datasets – FB15K and WN18 – have a large number of trivial triples. This is in contrast to four other pre-existing datasets in literature – FB13, WN11, FB15K-237, and WN18RR – which do not suffer from such bias. As mentioned above, we perform experiments spanning all six datasets.

5 Experiments

We chose HolE and ComplEx for comparison as these are the state-of-the-art in the current literature. Both models were re-implemented by us for fair comparison. We were able to achieve better performance for HolE on FB15K than what was reported in the original paper. Available results of ConvE and R-GCN have been taken from [3] for comparison. We evaluated these models on the link prediction task and the results are reported in Table 4.

5.1 Analysis of results

Based on the results in in Table 4, we make the following observations.

1. Neural network based models achieve state-of-the-art performance on WN11, FB13 and FB15K-237, and perform competitively on WN18. This is encouraging since all these datasets (except WN18) have zero trivial triples (Table 3) and are therefore more challenging compared to the other datasets.

2. Surprisingly, linear models such as ComplEx and HolE perform better than neural models on WN18RR, a dataset without trivial triples. This behavior has been related with the PageRank (a measure of indegree) of central nodes in different datasets by [3]. They found that linear models perform better on simpler datasets with low relation-specific indegree, such as WordNet. This is because they are easier to optimize and are able to find better local minima. Neural models show superior performance on complex datasets with higher relation-specific indegree.
3. Despite the effectiveness of a simple neural model like ER-MLP, such methods haven't received much attention in recent literature. Even though ER-MLP was compared against HolE in [9], a rigorous comparison involving diverse datasets was missing. Results in this paper address this gap, and show that such simple models merit further consideration in the future.

6 Conclusions and Future Work

In this work, we showed that the current state-of-the-art models do not achieve uniformly good performance across different datasets, and that the current benchmark datasets can be misleading when evaluating a model's ability to represent multi-relational graphs. We recommend that models henceforth be evaluated on multiple datasets so as to ensure their adaptability to Knowledge Graphs with different characteristics. We also showed that a neural network with a single hidden layer, which learns the scoring function together with the embeddings, can achieve competitive performance across datasets in spite of its simplicity. In future, we plan to identify the characteristics of datasets that determine the performance of various models.

References

[1] 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*. AcM, 1247–1250.

[2] 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*. 2787–2795.

[3] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel. 2017. Convolutional 2D Knowledge Graph Embeddings. *ArXiv e-prints* (July 2017). arXiv:cs.LG/1707.01476

[4] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 601–610.

[5] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[6] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion.. In *AAAI*. 2181–2187.

[7] Tom M Mitchell, William W Cohen, Estevam R Hruschka Jr, Partha Pratim Talukdar, Justin Betteridge, Andrew Carlson, Bhavana Dalvi Mishra, Matthew Gardner, Bryan Kisiel, Jayant Krishnamurthy, et al. 2015. Never Ending Learning.. In *AAAI*. 2302–2310.

[8] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*. 807–814.

[9] Maximilian Nickel, Lorenzo Rosasco, Tomaso A Poggio, et al. 2016. Holographic Embeddings of Knowledge Graphs.. In *AAAI*. 1955–1961.

[10] Maximilian Nickel and Volker Tresp. 2013. Logistic tensor factorization for multi-relational data. *arXiv preprint arXiv:1306.2084* (2013).

[11] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling. 2017. Modeling Relational Data with Graph Convolutional Networks. *ArXiv e-prints* (March 2017). arXiv:stat.ML/1703.06103

[12] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*. 926–934.

[13] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research* 15, 1 (2014), 1929–1958.

[14] Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing Text for Joint Embedding of Text and Knowledge Bases.. In *EMNLP*, Vol. 15. 1499–1509.

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

[16] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes.. In *AAAI*. 1112–1119.

[17] Robert West, Evgeniy Gabrilovich, Kevin Murphy, Shaohua Sun, Rahul Gupta, and Dekang Lin. 2014. Knowledge base completion via search-based question answering. In *Proceedings of the 23rd international conference on World wide web*. ACM, 515–526.

[18] Han Xiao, Minlie Huang, Yu Hao, and Xiaoyan Zhu. 2015. Transa: An adaptive approach for knowledge graph embedding. *arXiv preprint arXiv:1509.05490* (2015).

[19] Han Xiao, Minlie Huang, Yu Hao, and Xiaoyan Zhu. 2015. TransG: A Generative Mixture Model for Knowledge Graph Embedding. *arXiv preprint arXiv:1509.05488* (2015).

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