Knowledge Base Completion: Baselines Strike Back

Rudolf Kadlec and **Ondrej Bajgar** and **Jan Kleindienst**
IBM Watson
V Parku 4, 140 00 Prague, Czech Republic
{rudolf_kadlec, obajgar, jankle}@cz.ibm.com

Abstract

Many papers have been published on the knowledge base completion task in the past few years. Most of these introduce novel architectures for relation learning that are evaluated on standard datasets such as FB15k and WN18. This paper shows that the accuracy of almost all models published on the FB15k can be outperformed by an appropriately tuned baseline — our reimplementation of the DistMult model. Our findings cast doubt on the claim that the performance improvements of recent models are due to architectural changes as opposed to hyper-parameter tuning or different training objectives. This should prompt future research to re-consider how the performance of models is evaluated and reported.

1 Introduction

Projects such as Wikidata[1] or earlier Freebase (Bollacker et al., 2008) have successfully accumulated a formidable amount of knowledge in the form of ⟨entity1 - relation - entity2⟩ triplets. Given this vast body of knowledge, it would be extremely useful to teach machines to reason over such knowledge bases. One possible way to test such reasoning is knowledge base completion (KBC).

The goal of the KBC task is to fill in the missing piece of information into an incomplete triple. For instance, given a query ⟨Donald Trump, president of, ?⟩ one should predict that the target entity is USA.

More formally, given a set of entities $\mathcal{E}$ and a set of binary relations $\mathcal{R}$ over these entities, a *knowledge base* (sometimes also referred to as a knowledge *graph*) can be specified by a set of triplets $\langle h, r, t\rangle$ where $h, t \in \mathcal{E}$ are head and tail entities respectively and $r \in \mathcal{R}$ is a relation between them. In *entity* KBC the task is to predict either the tail entity given a query $\langle h, r, ?\rangle$, or to predict the head entity given $\langle ?, r, t\rangle$.

Not only can this task be useful to test the generic ability of a system to reason over a knowledge base, but it can also find use in expanding existing incomplete knowledge bases by deducing new entries from existing ones.

An extensive amount of work has been published on this task (for a review see (Nickel et al., 2015; Nguyen, 2017), for a plain list of citations see Table 2). Among those DistMult (Yang et al., 2015) is one of the simplest.[2] Still this paper shows that even a simple model with proper hyper-parameters and training objective evaluated using the standard metric of Hits@10 can outperform 27 out of 29 models which were evaluated on two standard KBC datasets, WN18 and FB15k (Bordes et al., 2013).

This suggests that there may be a huge space for improvement in hyper-parameter tuning even for the more complex models, which may be in many ways better suited for relational learning, e.g. can capture directed relations.

2 The Model

Inspired by the success of word embeddings in natural language processing, distributional models for KBC have recently been extensively studied. Distributional models represent the entities and sometimes even the relations as N-dimensional real vectors[3], we will denote these vectors by bold font, $\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{R}^N$.

[1] https://www.wikidata.org/

[2] We could even say too simple given that it assumes symmetry of all relations which is clearly unrealistic.

[3] Some models represent relations as matrices instead.

Proceedings of the 2nd Workshop on Representation Learning for NLP, pages 69–74,
Vancouver, Canada, August 3, 2017.

The DistMult model was introduced by Yang et al. (2015). Subsequently Toutanova and Chen (2015) achieved better empirical results with the same model by changing hyper-parameters of the training procedure and by using negative-log likelihood of softmax instead of L1-based max-margin ranking loss. Trouillon et al. (2016) obtained even better empirical result on the FB15k dataset just by changing DistMult's hyper-parameters.

DistMult model computes a score for each triplet $\langle \mathbf{h}, \mathbf{r}, \mathbf{t} \rangle$ as

$$s(\mathbf{h}, \mathbf{r}, \mathbf{t}) = \mathbf{h}^T \cdot W_\mathbf{r} \cdot \mathbf{t} = \sum_{i=1}^{N} h_i r_i t_i$$

where $W_\mathbf{r}$ is a diagonal matrix with elements of vector $\mathbf{r}$ on its diagonal. Therefore the model can be alternatively rewritten as shown in the second equality.

In the end our implementation normalizes the scores by a softmax function. That is

$$P(t|h, r) = \frac{exp(s(h, r, t))}{\sum_{\bar{t} \in \mathcal{E}_{h,r}} exp(s(h, r, \bar{t}))}$$

where $\mathcal{E}_{h,r}$ is a set of candidate answer entities for the $\langle h, r, ? \rangle$ query.

3 Experiments

Datasets. In our experiments we use two standard datasets WN18 derived from WordNet (Fellbaum, 1998) and FB15k derived from the Freebase knowledge graph (Bollacker et al., 2008).

Method. For evaluation, we use the *filtered* evaluation protocol proposed by Bordes et al. (2013). During training and validation we transform each triplet $\langle h, r, t \rangle$ into two examples: tail query $\langle h, r, ? \rangle$ and head query $\langle ?, r, t \rangle$. We train the model by minimizing negative log-likelihood (NLL) of the ground truth triplet $\langle h, r, t \rangle$ against randomly sampled pool of M negative triplets $\langle h, r, t' \rangle, t' \in \mathcal{E} \setminus \{t\}$ (this applies for tail queries, head queries are handled analogically).

In the filtered protocol we rank the validation or test set triplet against all corrupted (supposedly untrue) triplets – those that do not appear in the train, valid and test dataset (excluding the test set triplet in question itself). Formally, for a query $\langle h, r, ? \rangle$ where the correct answer is t, we compute the rank of $\langle h, r, t \rangle$ in a candidate set $C_{h,r} = \{\langle h, r, t' \rangle : \forall t' \in \mathcal{E}\} \setminus (Train \cup Valid \cup Test) \cup \{\langle h, r, t \rangle\}$, where $Train$, $Valid$ and $Test$ are sets of true triplets. Head queries $\langle ?, r, t \rangle$ are handled analogically. Note that softmax normalization is suitable under the filtered protocol since exactly one correct triplet is guaranteed to be among the candidates.

In our preliminary experiments on FB15k, we varied the batch size b, embedding dimensionality N, number of negative samples in training M, L2 regularization parameter and learning rate lr. Based on these experiments we fixed lr=0.001, L2=0.0 and we decided to focus on influence of batch size, embedding dimension and number of negative samples. For final experiments we trained several models from hyper-parameter range: $N \in \{128, 256, 512, 1024\}$, $b \in \{16, 32, 64, 128, 256, 512, 1024, 2048\}$ and $M \in \{20, 50, 200, 500, 1000, 2000\}$.

We train the final models using Adam (Kingma and Ba, 2015) optimizer ($lr = 0.001, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}, decay = 0.0$). We also performed limited experiments with Adagrad, Adadelta and plain SGD. Adagrad usually required substantially more iterations than ADAM to achieve the same performance. We failed to obtain competitive performance with Adadelta and plain SGD. On FB15k and WN18 validation datasets the best hyper-parameter combinations were $N = 512$, $b = 2048$, $M = 2000$ and $N = 256$, $b = 1024$, $M = 1000$, respectively. Note that we tried substantially more hyper-parameter combinations on FB15k than on WN18. Unlike most previous works we do not normalize neither entity nor relation embeddings.

To prevent over-fitting, we stop training once Hits@10 stop improving on the validation set. On the FB15k dataset our Keras (Chollet, 2015) based implementation with TensorFlow (Abadi et al., 2015) backend needed about 4 hours to converge when run on a single GeForce GTX 1080 GPU.

Results. Besides single models, we also evaluated performance of a simple ensemble that averages predictions of multiple models. This technique consistently improves performance of machine learning models in many domains and it slightly improved results also in this case.

The results of our experiments together with previous results from the literature are shown in Table 2. DistMult with proper hyperparameters twice achieves the second best score and once the third best score in three out of four commonly reported benchmarks (mean rank (MR) and

Hits@10 on WN18 and FB15k). On FB15k only the IRN model (Shen et al., 2016) shows better Hits@10 and the ProjE (Shi and Weniger, 2017) has better MR.

Our implementation has the best reported mean reciprocal rank (MRR) on FB15k, however this metric is not reported that often. MRR is a metric of ranking quality that is less sensitive to outliers than MR.

On WN18 dataset again the IRN model together with R-GCN+ shows better Hits@10. However, in MR and MRR DistMult performs poorly. Even though DistMult's inability to model asymmetric relations still allows it to achieve competitive results in Hits@10 the other metrics clearly show its limitations. These results highlight qualitative differences between FB15k and WN18 datasets.

Interestingly on FB15k recently published models (including our baseline) that use only $\mathbf{r}$ and $\mathbf{h}$ or $\mathbf{t}$ as their input outperform models that utilize richer features such as text or knowledge base path information. This shows a possible gap for future improvement.

Table 1 shows accuracy (Hits@1) of several models that reported this metric. On WN18 our implementation performs worse than HolE and ComplEx models (that are equivalent as shown by Hayashi and Shimbo (2017)). On FB15k our implementation outperforms all other models.

3.1 Hyper-parameter influence on FB15k

In our experiments on FB15k we found that increasing the number of negative examples M had a positive effect on performance.

Another interesting observation is that batch size has a strong influence on final performance. Larger batch size always lead to better results, for instance Hits@10 improved by 14.2% absolute when the batch size was increased from 16 to 2048. See Figure 1 for details.

Compared to previous works that trained DistMult on these datasets (for results see bottom of Table 2) we use different training objective than Yang et al. (2015) and Trouillon et al. (2017) that optimized max margin objective and NLL of softplus activation function ($softplus(x) = ln(1 + e^x)$), respectively. Similarly to Toutanova and Chen (2015) we use NLL of softmax function, however we use ADAM optimizer instead of RProp (Riedmiller and Braun, 1993).

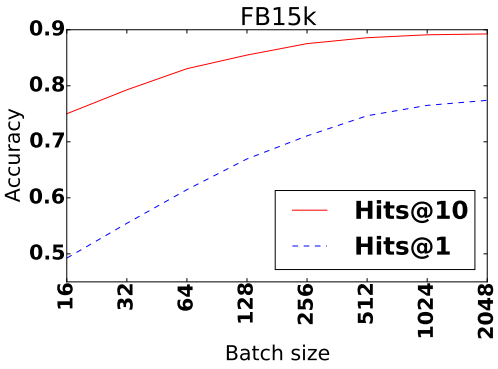

Figure 1: Influence of batch size on Hits@10 and Hits@1 metrics for a single model with $N = 512$ and $M = 2000$.

Method	Accuracy(Hits@1)	
	WN18	FB15k
HolE †	93.0	40.2
DistMult ‡	72.8	54.6
ComplEx ‡	**93.6**	59.9
R-GCN+ ♯	67.9	60.1
DistMult ensemble	78.4	**79.7**

Table 1: Accuracy (Hits@1) results sorted by performance on FB15k. Results marked by †, ‡ and ♯ are from (Nickel et al., 2016), (Trouillon et al., 2017) and (Schlichtkrull et al., 2017), respectively. Our implementation is listed in the last row.

4 Conclusion

Simple conclusions from our work are: 1) Increasing batch size dramatically improves performance of DistMult, which raises a question whether other models would also significantly benefit from similar hyper-parameter tuning or different training objectives; 2) In the future it might be better to focus more on metrics less frequently used in this domain, like Hits@1 (accuracy) and MRR since for instance on WN18 many models achieve similar, very high Hits@10, however even models that are competitive in Hits@10 underperform in Hits@1, which is the case of our DistMult implementation.

A lot of research focus has recently been centred on the filtered scenario which is why we decided to use it in this study. An advantage is that it is easy to evaluate. However the scenario trains the model to expect that there is only a single correct answer among the candidates which is unrealistic in the context of knowledge bases. Hence

Method	Filtered						Extra features
	WN18			FB15k			
	MR	H10	MRR	MR	H10	MRR	
SE (Bordes et al., 2011)	985	80.5	-	162	39.8	-	None
Unstructured (Bordes et al., 2014)	304	38.2	-	979	6.3	-	
TransE (Bordes et al., 2013)	251	89.2	-	125	47.1	-	
TransH (Wang et al., 2014)	303	86.7	-	87	64.4	-	
TransR (Lin et al., 2015b)	225	92.0	-	77	68.7	-	
CTransR (Lin et al., 2015b)	218	92.3	-	75	70.2	-	
KG2E (He et al., 2015)	331	92.8	-	59	74.0	-	
TransD (Ji et al., 2015)	212	92.2	-	91	77.3	-	
lppTransD (Yoon et al., 2016)	270	94.3	-	78	78.7	-	
TranSparse (Ji et al., 2016)	211	93.2	-	82	79.5	-	
TATEC (Garcia-Duran et al., 2016)	-	-	-	58	76.7	-	
NTN (Socher et al., 2013)	-	66.1	0.53	-	41.4	0.25	
HolE (Nickel et al., 2016)	-	94.9	**0.938**	-	73.9	0.524	
STransE (Nguyen et al., 2016)	**206**	93.4	0.657	69	79.7	0.543	
ComplEx (Trouillon et al., 2017)	-	94.7	**0.941**	-	84.0	0.692	
ProjE wlistwise (Shi and Weniger, 2017)	-	-	-	**34**	88.4	-	
IRN (Shen et al., 2016)	249	**95.3**	-	**38**	**92.7**	-	
RTransE (García-Durán et al., 2015)	-	-	-	50	76.2	-	Path
PTransE (Lin et al., 2015a)	-	-	-	58	84.6	-	
GAKE (Jun Feng and Zhu, 2015)	-	-	-	119	64.8	-	
Gaifman (Niepert, 2016)	352	93.9	-	75	84.2	-	
Hiri (Liu et al., 2016)	-	90.8	0.691	-	70.3	0.603	
R-GCN+ (Schlichtkrull et al., 2017)	-	**96.4**	0.819	-	84.2	0.696	
NLFeat (Toutanova and Chen, 2015)	-	94.3	**0.940**	-	87.0	**0.822**	Text
TEKE_H (Wang and Li, 2016)	**114**	92.9	-	108	73.0	-	
SSP (Xiao et al., 2017)	**156**	93.2	-	82	79.0	-	
DistMult (orig) (Yang et al., 2015)	-	94.2	0.83	-	57.7	0.35	None
DistMult (Toutanova and Chen, 2015)	-	-	-	-	79.7	0.555	
DistMult (Trouillon et al., 2017)	-	93.6	0.822	-	82.4	0.654	
Single DistMult (this work)	655	94.6	0.797	42.2	**89.3**	**0.798**	
Ensemble DistMult (this work)	457	**95.0**	0.790	**35.9**	**90.4**	**0.837**	

Table 2: Entity prediction results. MR, H10 and MRR denote evaluation metrics of mean rank, Hits@10 (in %) and mean reciprocal rank, respectively. The three best results for each metric are in bold. Additionally the best result is underlined. The first group (above the first double line) lists models that were trained only on the knowledge base and they do not use any additional input besides the source entity and the relation. The second group shows models that use path information, e.g. they consider paths between source and target entities as additional features. The models from the third group were trained with additional textual data. In the last group we list various implementations of the DistMult model including our implementation on the last two lines. Since DistMult does not use any additional features these results should be compared to the models from the first group. "NLFeat" abbreviates Node+LinkFeat model from (Toutanova and Chen, 2015). The results for NTN (Socher et al., 2013) listed in this table are taken from Yang et al. (2015). This table was adapted from (Nguyen, 2017).

future research could focus more on the *raw* scenario which however requires using other information retrieval metrics such as *mean average precision* (MAP), previously used in KBC for instance by Das et al. (2017).

We see this preliminary work as a small contribution to the ongoing discussion in the machine learning community about the current strong focus on state-of-the-art empirical results when it might be sometimes questionable whether they were achieved due to a better model/algorithm or just by more extensive hyper-parameter search. For broader discussion see (Church, 2017).

In light of these results we think that the field would benefit from a large-scale empirical comparative study of different KBC algorithms, similar to a recent study of word embedding models (Levy et al., 2015).

References

Martin Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Man, Rajat Monga, Sherry Moore, Derek Murray, Jon Shlens, Benoit Steiner, Ilya Sutskever, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Oriol Vinyals, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow : Large-Scale Machine Learning on Heterogeneous Distributed Systems .

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, New York, NY, USA, SIGMOD '08, pages 1247–1250. https://doi.org/10.1145/1376616.1376746.

Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2014. A semantic matching energy function for learning with multi-relational data. *Machine Learning* 94(2):233–259.

Antoine Bordes, Nicolas Usunier, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-Relational Data. *NIPS* 26:2787–2795. https://doi.org/10.1007/s13398-014-0173-7.2.

Antoine Bordes, Jason Weston, Ronan Collobert, and Yoshua Bengio. 2011. Learning structured embeddings of knowledge bases. In *Conference on artificial intelligence*. EPFL-CONF-192344.

Francois Chollet. 2015. Keras https://github.com/fchollet/keras/.

Kenneth Ward Church. 2017. Emerging trends: I did it, I did it, I did it, but... *Natural Language Engineering* 23(03):473–480. https://doi.org/10.1017/S1351324917000067.

Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew Mccallum. 2017. Chains of Reasoning over Entities, Relations, and Text using Recurrent Neural Networks. *EACL* .

Christiane Fellbaum. 1998. *WordNet*. Wiley Online Library.

Alberto García-Durán, Antoine Bordes, and Nicolas Usunier. 2015. Composing Relationships with Translations. In *Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*. Lisbonne, Portugal, pages 286–290. https://doi.org/10.18653/v1/D15-1034.

Alberto Garcia-Duran, Antoine Bordes, Nicolas Usunier, and Yves Grandvalet. 2016. Combining Two And Three-Way Embeddings Models for Link Prediction in Knowledge Bases. *Journal of Artificial Intelligence Research* 55:715—-742. https://doi.org/10.1613/jair.5013.

Katsuhiko Hayashi and Masashi Shimbo. 2017. On the Equivalence of Holographic and Complex Embeddings for Link Prediction pages 1–8. http://arxiv.org/abs/1702.05563.

Shizhu He, Kang Liu, Guoliang Ji, and Jun Zhao. 2015. Learning to Represent Knowledge Graphs with Gaussian Embedding. *CIKM '15 Proceedings of the 24th ACM International on Conference on Information and Knowledge Management* pages 623–632. https://doi.org/10.1145/2806416.2806502.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge Graph Embedding via Dynamic Mapping Matrix. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* pages 687–696. http://www.aclweb.org/anthology/P15-1067.

Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. 2016. Knowledge Graph Completion with Adaptive Sparse Transfer Matrix. *Proceedings of the 30th Conference on Artificial Intelligence (AAAI 2016)* pages 985–991.

Minlie Huang Yang Yang Jun Feng and Xiaoyan Zhu. 2015. GAKE: Graph Aware Knowledge Emebdding. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING'16)*. pages 641–651.

Diederik P. Kingma and Jimmy Lei Ba. 2015. Adam: a Method for Stochastic Optimization. *International Conference on Learning Representations* pages 1–13.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving Distributional Similarity with Lessons Learned from Word Embeddings. *Transactions of the Association for Computational Linguistics* 3:211–225. https://doi.org/10.1186/1472-6947-15-S2-S2.

Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2015a. Modeling relation paths for representation learning of knowledge bases. *CoRR* abs/1506.00379. http://arxiv.org/abs/1506.00379.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015b. Learning Entity and Relation Embeddings for Knowledge Graph Completion. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence Learning* pages 2181–2187.

Qiao Liu, Liuyi Jiang, Minghao Han, Yao Liu, and Zhiguang Qin. 2016. Hierarchical random walk inference in knowledge graphs. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, pages 445–454.

Dat Quoc Nguyen. 2017. An overview of embedding models of entities and relationships for knowledge base completion https://arxiv.org/pdf/1703.08098.pdf.

Dat Quoc Nguyen, Kairit Sirts, Lizhen Qu, and Mark Johnson. 2016. STransE: a novel embedding model of entities and relationships in knowledge bases. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* pages 460–466. https://doi.org/10.18653/v1/N16-1054.

Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2015. A Review of Relational Machine Learning for Knowledge Graph. *Proceedings of the IEEE* (28):1–23. https://doi.org/10.1109/JPROC.2015.2483592.

Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. 2016. Holographic Embeddings of Knowledge Graphs. *AAAI* pages 1955–1961. http://arxiv.org/abs/1510.04935.

Mathias Niepert. 2016. Discriminative gaifman models. In *Advances in Neural Information Processing Systems*. pages 3405–3413.

Martin Riedmiller and Heinrich Braun. 1993. A direct adaptive method for faster backpropagation learning: The rprop algorithm. In *Neural Networks, 1993., IEEE International Conference on*. IEEE, pages 586–591.

Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2017. Modeling Relational Data with Graph Convolutional Networks http://arxiv.org/abs/1703.06103.

Yelong Shen, Po-Sen Huang, Ming-Wei Chang, and Jianfeng Gao. 2016. Implicit reasonet: Modeling large-scale structured relationships with shared memory. *arXiv preprint arXiv:1611.04642* .

Baoxu Shi and Tim Weniger. 2017. ProjE : Embedding Projection for Knowledge Graph Completion. *AAAI* .

Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. 2013. Reasoning With Neural Tensor Networks for Knowledge Base Completion. *Proceedings of the Advances in Neural Information Processing Systems 26 (NIPS 2013)* .

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality* pages 57–66.

Théo Trouillon, Christopher R. Dance, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2017. Knowledge Graph Completion via Complex Tensor Factorization http://arxiv.org/abs/1702.06879.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. *Proceedings of ICML* 48:2071–2080. http://arxiv.org/pdf/1606.06357v1.pdf.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes. *AAAI Conference on Artificial Intelligence* pages 1112–1119.

Zhigang Wang and Juanzi Li. 2016. Text-enhanced representation learning for knowledge graph. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. AAAI Press, pages 1293–1299.

Han Xiao, Minlie Huang, and Xiaoyan Zhu. 2017. Ssp: Semantic space projection for knowledge graph embedding with text descriptions. In *Pro- ceedings of the 31st AAAI Conference on Artificial In- telligence*.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. *ICLR* page 12. http://arxiv.org/abs/1412.6575.

Hee-geun Yoon, Hyun-je Song, Seong-bae Park, and Se-young Park. 2016. A Translation-Based Knowledge Graph Embedding Preserving Logical Property of Relations. *Naacl* pages 1–9.