# Using Pairwise Occurrence Information to Improve Knowledge Graph Completion on Large-Scale Datasets

**Esma Balkır**[1,2*], **Masha Naslidnyk**[2], **Dave Palfrey**[2] and **Arpit Mittal**[2]

[1]University of Edinburgh, Scotland, UK
[2]Amazon Research, Cambridge, UK
[1]esma.balkir@ed.ac.uk
[2]{naslidny, dpalfrey, mitarpit}@amazon.co.uk

## Abstract

Bilinear models such as DistMult and ComplEx are effective methods for knowledge graph (KG) completion. However, they require large batch sizes, which becomes a performance bottleneck when training on large scale datasets due to memory constraints. In this paper we use occurrences of entity-relation pairs in the dataset to construct a joint learning model and to increase the quality of sampled negatives during training. We show on three standard datasets that when these two techniques are combined, they give a significant improvement in performance, especially when the batch size and the number of generated negative examples are low relative to the size of the dataset. We then apply our techniques to a dataset containing 2 million entities and demonstrate that our model outperforms the baseline by 2.8% absolute on hits@1.

## 1 Introduction

A Knowledge Graph (KG) is a collection of facts which are stored as triples, e.g. *Berlin is-capital-of Germany*. Even though knowledge graphs are essential for various NLP tasks, open domain knowledge graphs have missing facts. To tackle this issue, there has recently been considerable interest in KG completion methods, where the goal is to rank correct triples above incorrect ones.

Embedding methods such as DistMult (Yang et al., 2014) and ComplEx (Trouillon et al., 2016) are simple and effective methods for this task, but are known to be sensitive to hyperparameter and loss function choices (Kadlec et al., 2017; Lacroix et al., 2018). When paired with the right loss function, these methods need large minibatches and a large number of corrupted triples per each positive triple during training to reach peak performance. [1] This causes memory issues for KGs in the wild, which are several magnitudes bigger than the common benchmarking datasets.

To address the issue of scalability, we develop a framework that could be used with any bilinear KG embedding model. We name our model **JoBi** (**Jo**int model with **Bi**ased negative sampling). Our framework uses occurrences of entity-relation pairs to overcome data sparsity, and to bias the model to score plausible triples higher. The framework trains a base model jointly with an auxiliary model that uses occurrences of pairs within a given triple in the data as labels. For example, the auxiliary model would receive the label 1 for the triple (*Berlin is-capital-of France*) if the pairs (*Berlin is-capital-of*) and (*is-capital-of France*) are present in the training data, while the base model would receive the label 0.

The intuition for using bigram occurrences is to capture some information about restrictions on the set of entities that could appear as the object or subject of a given relation; that information should implicitly correspond to some underlying type constraints. For example, even if *(Berlin is-capital-of France)* is not a correct triple, *Berlin* is the right type for the subject of *is-capital-of*.

Our framework also utilizes entity-relation pair occurrences to improve the distribution of negative examples for contrastive training, by sampling a false triple e.g. (*Berlin is-capital-of France*) with higher probability if the pairs (*Berlin is-capital-of*) and (*is-capital-of France*) both occur in the dataset. This tunes the noise distribution for the task so that it is more challenging, and hence the model needs a fraction of the negative examples compared to a uniform distribution.

We show empirically that joint training is especially beneficial when the batch size is small,

*Work done while the author was an intern

[1]The choice of loss function has a very strong effect on the optimal negative ratio, but with any loss function, larger batches tend to improve the results.



*Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3591–3596, Hong Kong, China, November 3–7, 2019. 

and biased negative sampling helps model learn higher quality embeddings with much fewer negative samples. We show that the two techniques are complementary and perform significantly better when combined. We then test JoBi on a large-scale dataset, and demonstrate that JoBi learns better embeddings in very large KGs.

## 2 Background

Formally, given a set of entities $\mathcal{E} = \{e_0, \ldots, e_n\}$ and a set of relations $\mathcal{R} = \{r_0, \ldots, r_m\}$, a Knowledge Graph (KG) is a set triples in the form $\mathcal{G} = \{(h, r, t)\} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, where if a triple $(h, r, t) \in \mathcal{G}$, then relation $r$ holds between entities $h$ and $t$. Given such a KG, the aim of KG completion is score each triple in $\mathcal{E} \times \mathcal{R} \times \mathcal{E}$, so that correct triples are assigned higher scores than the false ones. KG embedding methods achieve this by learning dense vector representations for entities and relations through optimizing a chosen scoring function. A class of KG completion models such as RESCAL (Nickel et al., 2012), DistMult (Yang et al., 2014), ComplEx (Trouillon et al., 2016), SimplE (Kazemi and Poole, 2018) and TUCKER (Balažević et al., 2019) define their scoring function to be a bilinear interaction of the embeddings of entities and relations in the triple. For this work we consider DistMult, ComplEx and SimplE as our baseline models due to their simplicity.

**DistMult.** (Yang et al., 2014) is a knowledge graph completion model that defines the scoring function for a triple as a simple bilinear interaction, where the entity has the same representation regardless of whether it appears as the head or the tail entity. For entities $h, t$, relation $r$, and the embeddings $\boldsymbol{h}, \boldsymbol{t}, \boldsymbol{r} \in \mathbb{R}^d$, the scoring function is defined as:

$$s(h, r, t) = \boldsymbol{h}^T \operatorname{diag}(\boldsymbol{r})\, \boldsymbol{t} \tag{1}$$

where $\operatorname{diag}(\boldsymbol{r})$ is a diagonal matrix with $\boldsymbol{r}$ on the diagonal.

**ComplEx.** (Trouillon et al., 2016) is a bilinear model similar to DistMult. Because of its symmetric structure, DistMult cannot model anti-symmetric relations. ComplEx overcomes this shortcoming by learning embeddings in a complex vector space, and defining the embedding of an entity in tail position as the complex conjugate of the embedding in the head position.

Let $\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \in \mathbb{C}^d$ be the embeddings for $h, r, t$. The score for ComplEx is defined as follows:

$$s(h, r, t) = \operatorname{Re}\left(\boldsymbol{h}^T \operatorname{diag}(\boldsymbol{r}) \bar{\boldsymbol{t}}\right) \tag{2}$$

Where $\overline{\boldsymbol{a}}$ denotes the complex conjugate of $\boldsymbol{a}$, and $\operatorname{Re}(\boldsymbol{a})$ denotes the real part of the complex vector $\boldsymbol{a}$.

**SimplE.** (Kazemi and Poole, 2018) is also a bilinear model similar to DistMult. Each entity has two associated embeddings $\boldsymbol{e}_1, \boldsymbol{e}_2 \in \mathbb{R}^d$, where one is the representation of $e$ as the head, and the other as the tail entity of the triple. Each relation also has two associated embeddings: $\boldsymbol{r}$ and $\boldsymbol{r}^{-1}$, where $\boldsymbol{r}^{-1}$ is the representation for the reverse of $r$. The score function is defined as:

$$\begin{aligned} s(h, r, t) = 1/2\, (\boldsymbol{h}_1)^T \ \operatorname{diag}(\boldsymbol{r}) \boldsymbol{t}_1 \\ +1/2\, (\boldsymbol{t}_2)^T \ \operatorname{diag}(\boldsymbol{r}^{-1}) \boldsymbol{h}_2 \end{aligned} \tag{3}$$

## 3 Joint framework

JoBi contains two copies of a bilinear model, where one is trained on labels of triples, and the other on occurrences of entity-relation pairs within the triples. For the pair module, we label a triple $(h, r, t)$ correct if there are triples $(h, r, t')$ and $(h', r, t)$ in the training set for some $t'$ and $h'$.

The scoring functions for the two models are $s_{\text{bi}}$ and $s_{\text{tri}}$, for the pair and the triple modules respectively. We tie the weights of the entity embeddings, but let the embeddings for the relations be optimized separately. The equations using ComplEx as the base model are as follows:

$$s_{\text{tri}}(h, r, t) = \operatorname{Re}\left(\boldsymbol{h}^T \operatorname{diag}(\boldsymbol{r}_{\text{tri}}) \bar{\boldsymbol{t}}\right) \tag{4}$$

$$s_{\text{bi}}(h, r, t) = \operatorname{Re}\left(\boldsymbol{h}^T \operatorname{diag}(\boldsymbol{r}_{\text{bi}}) \bar{\boldsymbol{t}}\right) \tag{5}$$

We define the framework for DistMult and SimplE analogously. During training, we optimize the two jointly, but use only $s_{\text{tri}}$ during test time. Hence, the addition of the auxiliary module has no effect on the number of final parameters of the trained model. Note that even during training, this doesn't increase model complexity in any significant way since the number of relations in KGs are often a fraction of the number of entities.

For each triple in the minibatch, we generate $n_{\text{neg}}$ negative examples per positive triple by randomly corrupting the head or the tail entity. For $s_{\text{tri}}$, we use the negative log-likelihood of softmax as the loss function, and for $s_{\text{bi}}$ we use binary cross

| Dataset | #entities | #relations | #train |
|---|---|---|---|
| FB15K | 14,951 | 1,345 | 483,142 |
| FB15K-237 | 14,541 | 237 | 272,115 |
| YAGO3-10 | 123,182 | 37 | 1,079,040 |
| FB1.9M | 1,892,241 | 3,247 | 19,323,513 |

Table 1: Statistics of datasets used in experiments.

entropy loss. We combine the two losses via a simple weighted addition with a tunable hyperparameter $\alpha$:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{tri}} + \alpha \mathcal{L}_{\text{bi}} \quad (6)$$

**Biased negative sampling.** We also examine the effect of using the pair cooccurrence information for making the contrastive training more challenging for the model. For this, we keep the model as is, but with probability $p$, instead of corrupting the head or the tail of the triple with an entity chosen uniformly at random, we corrupt it with an entity that is picked with uniform probability from the set of entities that occur as the head or tail entity of the relation in the given triple. To illustrate, when sampling a negative tail entity for the tuple *Berlin is-capital-of*, this method causes the model to pick *France* with higher probability than *George Orwell* if *France* but not *George Orwell* occurs as the head entity for the relation *is-capital-of* in the training data.

## 4 Experiments

We perform our experiments on standard datasets FB15K (Bordes et al., 2013), FB15K-237 (Toutanova et al., 2015), YAGO3-10 (Dettmers et al., 2018), and on a new large-scale dataset FB1.9M which we construced from FB3M (Xu and Barbosa, 2018). [2] We focus on YAGO3-10 since it is 10 times larger than the other two and better reflects how the performance of the models scale. We present the comparison of the sizes of these datasets in Table 1, and further details could be found in Appendix A.

For evaluation, we rank each triple $(h, r, t)$ in the test set against $(h', r, t)$ for all entities $h'$, and similarly against $(h, r, t')$ for all entities $t'$. We filter out the candidates that have occurred in training, validation or test set as described in Bordes et al. (2013), and we report average hits@1, 3, 10 and mean reciprocal rank (MRR).

We re-implement all our baselines and obtain very competitive results. In our preliminary experiments on baselines, we found that the choice of loss function had a large effect on performance, with negative log-likelihood (NLL) of softmax consistently outperforming both max-margin and logistic losses. Larger batch sizes lead to better performance. With NLL of sampled softmax, we found that increasing the number of generated negatives steadily increases performance[3], and state-of-the-art results could be reached by using the full softmax as used in Joulin et al. (2017) and Lacroix et al. (2018). This technique is possible for standard benchmarks but not for large KGs, and we report results in Appendix D for all datasets small enough to allow for full contrastive training. However, our main experiments use NLL of sampled softmax since our focus is on scalability. Note that results with full softmax (Appendix D) demonstrate that our implementation of baselines is very competitive.Our implementation of ComplEx performs significantly better than ConvE (Dettmers et al., 2018) on two out of the three datasets, and come close to results of Lacroix et al. (2018) who use extremely large embeddings as well as full softmax, thus cannot be scaled. Our code is publicly available. [4]

For most of our experiments, we choose to use ComplEx as the base for our model (**JoBi ComplEx**), since this configuration consistently outperformed others in preliminary experiments. To test the effect of our techniques on different bilinear models, we report results with DistMult (**JoBi DistMult**) and SimplE (**JoBi SimplE**) on FB15K-237.

**Discussion.** It could be seen in Table 2 that JoBi ComplEx outperforms both ComplEx and DistMult on all three standard datasets, on all the metrics we consider. For Hits@1, JoBi Complex outperforms baseline ComplEx by 4% on FB15K-237, 6.4% on FB15K and 5.6% on YAGO3-10.

Moreover, results in Table 2 demonstrate that JoBi improves performance on DistMult and SimplE. It should be noted that on FB15K-237, all JoBi models outperform all the baseline models, regardless of the base model used.

Lastly, results on FB1.9M (Table 3) demonstrate that JoBi improves performance on this very

[2] We do not perform experiments on WordNet derived datasets WN18 or WN18RR because bigram modelling would not provide any information – all entities are synsets and almost all can occur as an object or subject to all the possible relations.

[3] This effect is not observed when using logistic-loss or max-margin loss

[4] https://github.com/awslabs/joint_biased_embeddings

| FB15K-237 | *h@1* | *h@3* | *h@10* | *MRR* |
|---|---|---|---|---|
| SimplE | 0.160 | 0.268 | 0.430 | 0.248 |
| DistMult | 0.158 | 0.271 | 0.432 | 0.247 |
| ComplEx | 0.159 | 0.275 | 0.441 | 0.25 |
| JoBi SimplE | 0.188 | 0.301 | 0.461 | 0.277 |
| JoBi DistMult | **0.205** | 0.316 | 0.466 | 0.29 |
| JoBi ComplEx | 0.199 | **0.319** | **0.479** | **0.29** |
| **FB15K** | *h@1* | *h@3* | *h@10* | *MRR* |
| DistMult | 0.587 | 0.785 | 0.867 | 0.697 |
| ComplEx | 0.617 | 0.803 | 0.874 | 0.72 |
| JoBi ComplEx | **0.681** | **0.824** | **0.883** | **0.761** |
| **YAGO3-10** | *h@1* | *h@3* | *h@10* | *MRR* |
| DistMult | 0.252 | 0.407 | 0.568 | 0.357 |
| ComplEx | 0.277 | 0.44 | 0.589 | 0.383 |
| JoBi ComplEx | **0.333** | **0.477** | **0.617** | **0.428** |

Table 2: Performance on different datasets against baselines, where h@$k$ denotes hits at $k$. Results are reported on test sets with the best parameters found in grid search for each model.

| ComplEx | *h@1* | *h@3* | *h@10* | *MRR* |
|---|---|---|---|---|
| Baseline | 0.424 | 0.598 | 0.721 | 0.530 |
| JoBi | **0.452** | **0.615** | **0.726** | **0.550** |

Table 3: Performance on the large-scale FB1.9M dataset, measured against the best performing baseline.

| | # epochs | training time |
|---|---|---|
| ComplEx | 70 | 5 days 5 hours 8 minutes |
| JoBi ComplEx | 30 | 4 days 19 minutes |

Table 4: Runtimes of ComplEx and JoBi Complex on FB1.9M.

large dataset, where it is not possible to perform softmax over the entire set of entities, or have very large embedding sizes due to memory constraints.

Although one epoch for JoBi takes slightly longer than the baseline, JoBi converges in fewer epochs, resulting in shorter running time overall. We report running times on FB1.9M in Table 4.

**Comparison with TypeComplex** For results of TypeComplex, Jain et al. (2018) use a wider set of negative ratios in their grid search than we do. To isolate the effects of the different models from hyperparameter choices, we set the negative ratio for our model to be 400 to match the setting on their best performing models. We keep the other hyperparameters the same as the best performing models for the previous experiments.

Jain et al. (2018) use a modified version of the ranking evaluation procedure to report their results, where they only rank the tail entity against all other entities. To be able to compare our model to theirs, we also report the performance of our framework on this modified metric. The results

| FB15K-237 | *h@1* | *h@3* | *h@10* | *MRR* |
|---|---|---|---|---|
| ComplEx | 0.209 | 0.347 | 0.535 | 0.314 |
| TypeComplex | **0.296** | - | 0.575 | **0.389** |
| JoBi ComplEx | 0.276 | 0.416 | **0.587** | 0.377 |
| **FB15K** | *h@1* | *h@3* | *h@10* | *MRR* |
| ComplEx | 0.630 | 0.818 | 0.895 | 0.734 |
| TypeComplex | 0.663 | - | 0.885 | 0.754 |
| JoBi ComplEx | **0.702** | 0.847 | **0.906** | **0.782** |
| **YAGO3-10** | *h@1* | *h@3* | *h@10* | *MRR* |
| ComplEx | 0.412 | 0.587 | 0.701 | 0.516 |
| TypeComplex | **0.516** | - | 0.702 | 0.587 |
| JoBi ComplEx | 0.507 | 0.647 | **0.742** | **0.591** |

Table 5: Comparison with TypeComplex where the scores are calculated ranking only the tail entities. Results for TypeComplex are taken from Jain et al. (2018). h@$k$ denotes hits at $k$.

for these experiments can be found in Table 5.

Our model generally outperforms TypeComplex by a large margin on hits@10. It also outperforms TypeComplex on MRR by a moderate margin except on FB15K-237, the smallest dataset. On the other hand, TypeComplex outperforms our model on hits@1 in two out of the three datasets. In fact for FB15K, TypeComplex does worse on hits@10 compared to the baseline model. This suggests that TypeComplex may be compromising on hits@k where k is larger to improve the hits@1 metric, which might be undesirable depending on the application.

**Qualitative analysis.** We analyzed correct predictions made by JoBi ComplEx but not regular ComplEx. Among relations in YAGO3-10, major gains can be observed for *hasGender* (Appendix C). The improvement comes solely from tail-entity predictions, with hits@1 increasing from 0.22 to 0.86. Furthermore, we found that the errors made by ComplEx are exactly of the kind that can be mitigated by enforcing plausibility: ComplEx predicts an object that is not a gender (e.g. a sports team or a person) 65% of the time; JoBi makes such an obvious mistake only 2% of the time.

**Ablation studies.** We compare joint training without biased sampling (**Joint**) and biased sampling without joint training (**BiasedNeg**) to the full model JoBi on YAGO3-10. The results can be found in Table 6. We also conduct experiments to isolate the effect of our techniques on varying batch sizes and negative ratios. The results for this experiment are presented in Figures 1 and 2. Training details can be found in Appendix B.

| | *h@1* | *h@3* | *h@10* | *MRR* |
|---|---|---|---|---|
| Baseline | 0.277 | 0.44 | 0.589 | 0.383 |
| BiasedNeg | 0.276 | 0.427 | 0.568 | 0.375 |
| Joint | 0.287 | 0.447 | 0.601 | 0.392 |
| JoBi | **0.333** | **0.477** | **0.617** | **0.428** |

Table 6: Results of ablation study on ComplEx model.

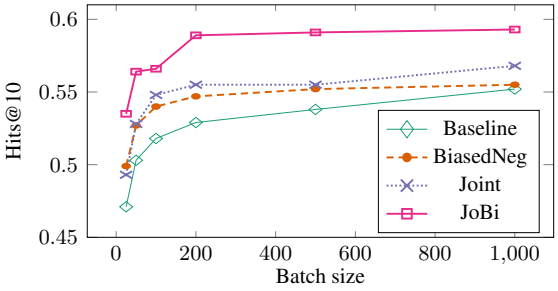


Figure 1: Performances on YAGO3-10 with different batch sizes

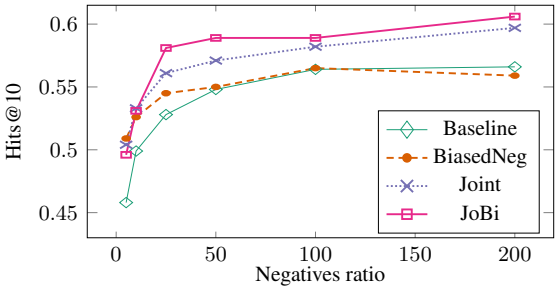


Figure 2: Performances on YAGO3-10 with different negative ratios

In Table 6 it can be seen that Joint on its own gives a slight performance boost over the baseline, and BiasedNeg performs slightly under the baseline on all measures. However, combining our two techniques in JoBi gives 5.6% points improvement on hits@1. This suggests that biased negative sampling increases the efficacy of joint training greatly, but is not very effective on its own.

Figure 1 and 2 shows that JoBi not only consistently performs the best over the entire range of parameters, but also delivers a performance improvement that is especially large when the batch size or the negative ratio is small. This setting was designed to reflect the training conditions on very large datasets. It can be seen that BiasedNeg is more robust to low values of negative ratios, and both BiasedNeg and Joint alone show less deterioration in performance as the batch size decreases. When these two methods are combined in JoBi, the training becomes more robust to different choices on both these parameters.

The reason behind BiasedNeg performing worse on its own but better with Joint could be the choice of binary cross entropy loss for the pair module. We speculate that as the negative ratio increases, the ratio of negative to positive examples for this module becomes more skewed. Biasing the negative triples in the training alleviates this problem by making the classes more balanced, and allows the joint training to be more effective.

### 4.1 Related work

Utilizing pair occurrences for embedding models have been considered before, both as explicit model choices and as negative sampling strategies. Chang et al. (2014) and Krompaß et al. (2015) use pair occurrences to constrain the set of triples to be used in the optimization procedure. For methods that rely on SGD with contrastive training, this translates to a special case of our biased sampling method where $p = 1$. Garcia-Durán et al. (2016) present *TATEC*, a model that combines bigram and trigram interactions. The trigram model uses a full matrix representation for relations, and hence has many more parameters compared to our model. Jain et al. (2018) present *JointDM* and *JointComplex*, which could be viewed as a simplification of *TATEC*. Unlike our model, both of these methods use the bigram terms both in training and evaluation, do not share any of the embeddings between two models, and do not provide supervision based on pair occurrences in the data. Other methods that have been considered for improving the negative sampling procedure includes adversarial (Cai and Wang, 2018) and self-adversarial (Sun et al., 2019) training. None of these methods focus on improving the models to scale to large KGs.

## 5 Conclusion

We have presented a joint framework for KG completion that utilizes entity-relation pair occurrences as an auxiliary task, and combined it with a technique to generate informative negative examples with higher probability. We have shown that joint training makes the model more robust to smaller batch sizes, and biased negative sampling to different values of the number of generated negative samples. Furthermore, these techniques perform well above baselines when combined, and are effective on a very large KG dataset. Applying JoBi to non-bilinear models is also possible, but left for future work.

## References


Ivana Balažević, Carl Allen, and Timothy M Hospedales. 2019. Tucker: Tensor factorization for knowledge graph completion. *arXiv preprint arXiv:1901.09590*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, pages 2787–2795. Curran Associates Inc.

Liwei Cai and William Yang Wang. 2018. Kbgan: Adversarial learning for knowledge graph embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1470–1480.

Kai-Wei Chang, Scott Wen-tau Yih, Bishan Yang, and Chris Meek. 2014. Typed tensor decomposition of knowledge bases for relation extraction. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*.

Tim Dettmers, Minervini Pasquale, Stenetorp Pontus, and Sebastian Riedel. 2018. Convolutional 2D Knowledge Graph Embeddings. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence*, pages 1811–1818.

Alberto Garcia-Durán, Antoine Bordes, Nicolas Usunier, and Yves Grandvalet. 2016. Combining two and three-way embedding models for link prediction in knowledge bases. *Journal of Artificial Intelligence Research*, 55:715–742.

Prachi Jain, Pankaj Kumar, Soumen Chakrabarti, and others. 2018. Type-Sensitive Knowledge Base Inference Without Explicit Type Supervision. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 75–80.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Maximilian Nickel, and Tomas Mikolov. 2017. Fast Linear Model for Knowledge Graph Embeddings. *arXiv preprint arXiv:1710.10881*.

Rudolf Kadlec, Ondrej Bajgar, and Jan Kleindienst. 2017. Knowledge Base Completion: Baselines Strike Back. *arXiv preprint arXiv:1705.10744*.

Seyed Mehran Kazemi and David Poole. 2018. SimplE Embedding for Link Prediction in Knowledge Graphs. *arXiv preprint arXiv:1802.04868*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Denis Krompaß, Stephan Baier, and Volker Tresp. 2015. Type-constrained representation learning in knowledge graphs. In *International Semantic Web Conference*, pages 640–655.

Timothe Lacroix, Nicolas Usunier, and Guillaume Obozinski. 2018. Canonical Tensor Decomposition for Knowledge Base Completion. In *Proceedings of the 35th International Conference on Machine Learning*.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2012. Factorizing YAGO: scalable machine learning for linked data. In *Proceedings of the 21st International Conference on World Wide Web*, pages 271–280, New York, New York, USA. ACM.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. YAGO. In *Proceedings of the 16th International Conference on World Wide Web - WWW '07*, page 697, New York, New York, USA. ACM Press.

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

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 *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1499–1509, Lisbon. ACM.

Tho Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 2071 – 2080.

Peng Xu and Denilson Barbosa. 2018. Investigations on Knowledge Base Embedding for Relation Prediction and Extraction. *arXiv preprint arXiv:1802.02114*.

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*.