Loss Functions in Knowledge Graph Embedding Models

Sameh K. Mohamed[1], Vít Nováček[1], Pierre-Yves Vandenbussche[2], and Emir Muñoz[1]

[1] Data Science Institute at National University of Ireland, Galway, Ireland
[2] Fujitsu Ireland Ltd, Galway, Ireland
sameh.kamal@insight-centre.org

Abstract. Knowledge graph embedding (KGE) models have become popular for their efficient and scalable discoveries in knowledge graphs. The models learn low-rank vector representations from the knowledge graph entities and relations. Despite the rapid development of KGE models, state-of-the-art approaches have mostly focused on new ways to represent embeddings interaction functions (*i.e.*, scoring functions). However, we argue that the choice of a training loss function can have a substantial impact on a model's efficiency, which has been rather neglected by the state of the art so far. In this paper, we provide a thorough analysis of different loss functions that can help with the procedure of embedding learning, providing a reduction of the evaluation metric based error. We experiment with the most common loss functions for KGE models and also suggest a new loss for representing training error in KGE models. Our results show that a loss based on training error can enhance the performance of current models on multiple datasets.

1 Introduction

The recent advent of knowledge graph embedding (KGE) models has allowed for scalable and efficient manipulation of large knowledge graphs (KGs), improving the results of a wide range of tasks such as link prediction [3,21], entity resolution [15,2] and entity classification [16]. KGE models operate by learning embeddings in a low-dimensional continuous space from the relational information contained in the KG while preserving its inherent structure. Specifically, their objective is to rank knowledge facts—relational triples (s, p, o) connecting subject and object entities s and o by a relation type p—based on their relevance. Various interactions between their entity and relation embeddings are used for computing the knowledge fact ranking. These interactions are typically reflected in a model-specific scoring function. For instance, TransE [3] uses a scoring function defined as the distance between the o embedding and the translation of the embedding associated to s by the relation type p embedding. DistMult [22], ComplEx [19] and HolE [14] use multiplicative composition of the entity embeddings and the relation type embeddings. This leads to a better reflection of the

relational semantics and to state-of-the-art performance results (see [20] for a review).

Although there is a growing body of literature proposing different KG models (mostly focusing on the design of new scoring functions), the study of loss functions—a core part of the learning process—has not received much attention to date. This has already been shown to influence the behaviour of the KGE models. For instance, [9] observed that despite the different motivations behind HolE and CompleEx models, they have equivalent scoring functions. Yet their performance still differs. In [18], the authors conclude that this difference is caused by the fact that HolE uses a max-margin loss while ComplEx uses a log-likelihood loss, showing that loss functions are important for thorough understanding, and even improvement of the performance of different KGE models. However, a comprehensive study is still missing.

In this paper, we focus on comparing different loss functions when applied to several representative KGE models. By performing a systematic analysis of the performance (in terms of Mean Reciprocal Rank, MRR) of different models using different loss functions, we hope to contribute towards improving the understanding of how loss functions influence the behaviour of KGE models across different benchmark datasets.

The summary of our contributions is as follows:

(a) We provide a comprehensive analysis of training loss functions as used in several state-of-the-art KGE models (Section 2);
(b) We preform an empirical evaluation of different KGE models with different loss functions, and show the effect of different losses on the KGE models predictive accuracy (Section 3);
(c) We propose a new formulation for a KGE loss that can provide enhancements to the performance of KGE models. Section 3 demonstrates experimentally that the proposed loss function can enhance performance of state-of-the-art KGE models over multiple datasets.

2 Loss Functions in KGE Models

Generally, KGE models are cast as learning to rank problems. They employ multiple training loss functions that comply with the ranking loss approaches. In the state-of-the-art KGE models, loss functions were designed according to various pointwise and pairwise approaches that we review next.

2.1 KGE Pointwise Losses

First, we discuss existing pointwise loss functions for KGE models, namely, square error (SE), hinge, and logistic losses. Let $\mathbf{x} \in X$ be one fact of the KG, f a scoring function, and l a labelling function.

Pointwise Square Error Loss (SE). SE is the ranking loss function used in RESCAL [15]. It models training losses with the objective of minimising the

squared difference between model scores and labels (expected output):

$$\underset{\text{SE}_{Pt}}{\mathcal{L}} = \frac{1}{2}\sum_{\mathbf{x}\in X}(f(\mathbf{x}) - l(\mathbf{x}))^2.$$

The optimal score for true and false facts is 1 and 0, respectively. A nice to have characteristic of SE loss is that it does not require configurable training parameters, shrinking the search space of hyper parameters compared to other losses (*e.g.*, the margin parameter of the hinge loss).

Pointwise Hinge Loss. Hinge loss can be interpreted as a pointwise loss, where the objective is to generally minimise the scores of negative facts and maximise the scores of positive facts to a specific configurable value. This approach is used in HolE [14], and it is defined as:

$$\underset{\text{hinge}_{Pt}}{\mathcal{L}} = \sum_{\mathbf{x}\in X}[\lambda - l(\mathbf{x})\cdot f(\mathbf{x})]_+,$$

where $l(\mathbf{x}) = 1$ if $\mathbf{x}$ is true and -1 otherwise, and $[x]_+$ denotes $max(x, 0)$. This effectively generates two different loss slopes for positive and negative scores. Thus, the objective resembles a pointwise loss that minimises negative scores to reach $-\lambda$, and maximises positives scores to reach λ.

Pointwise Logistic Loss. The ComplEx [19] model uses a logistic loss, which is a smoother version of pointwise hinge loss without the margin parameter. Logistic loss uses a logistic function to minimise the negative triples score and maximise the positive triples score. This is similar to hinge loss, but uses a smoother linear loss slope defined as:

$$\underset{\text{logistic}_{Pt}}{\mathcal{L}} = \sum_{\mathbf{x}\in X}\log(1 + \exp(-l(\mathbf{x})\cdot f(\mathbf{x}))),$$

where $l(\mathbf{x})$ is the true label of fact $\mathbf{x}$ where it is equal to 1 for positive facts and is equal to -1 otherwise.

Taking the best of the previous loss functions, we propose a new pointwise loss, called the Pointwise Square Loss (PSL), which combines the square growth of SE and the configurable margin of hinge loss.

Pointwise Square Loss (PSL). In the SE loss, the objective is to set the scores of negative and positive instances to 0 and 1, respectively. As a result, the scores of negative instances that are less than 0, and the scores of positive instances that are greater than 1 are penalised despite their actual compliance with the main training objective: $\forall_{\mathbf{x}\in X^+}\forall_{\mathbf{x}'\in X^-} f(\mathbf{x}) > f(\mathbf{x}')$, where X^+ and X^- are the sets of positive and negative facts, respectively. Therefore, we propose a new loss, PSL, that allows scores of positive instances to grow and scores of negative instances to decrease without boundaries. We also use a configurable value λ instead of 0 and 1 to allow for more search configurations as in hinge and logistic losses. PSL is defined as:

$$\underset{\text{PSL}_{Pt}}{\mathcal{L}} = \frac{1}{2}\sum_{\mathbf{x}\in X}([\lambda - l(\mathbf{x})\cdot f(\mathbf{x})]_+)^2,$$

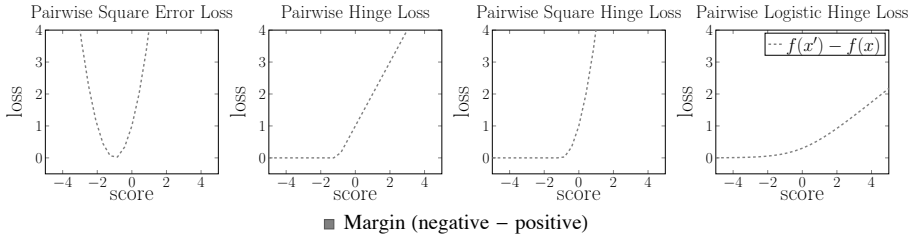

Fig. 1. Plots of growth of pairwise margin-based losses: compared to its margin with default $\lambda = 1$ and $\alpha = 0.5$.

where $l(\mathbf{x}) = 1$ if $\mathbf{x}$ is true and -1 otherwise. We can see that PSL can be made equivalent to squared hinge loss by defining it as $\mathcal{L}_{\text{PSL}_{Pt}}(f; X, l) = \mathcal{L}_{\text{hinge}_{Pt}}(f; X, l)^2$.

2.2 KGE Pairwise Losses

Here, we discuss established pairwise loss functions in KGE models, and present two new proposed loss functions, namely *tanh* and *softsign* losses. Fig. 1 shows the set of pairwise loss functions to be discussed in this subsection.

Pairwise Hinge Loss. Hinge loss is a linear learning-to-rank loss that it is used for maximum-margin classification and can be implemented in both pointwise or pairwise settings. TransE [3] and DistMult [22] models use the pairwise margin based hinge loss. It is defined as:

$$\underset{\text{hinge}_{Pr}}{\mathcal{L}} = \sum_{\mathbf{x} \in X^+} \sum_{\mathbf{x}' \in X^-} [\lambda + f(\mathbf{x}') - f(\mathbf{x})]_+,$$

where X^+ is the set of true facts, X^- is the set of false facts, and λ is a configurable margin. In this case, the objective is to maximise the difference between the scores of negative and positive instances by a good margin. This approach optimises towards having embeddings that satisfy $\forall_{\mathbf{x} \in X^+} \forall_{\mathbf{x}' \in X^-} f(\mathbf{x}) > f(\mathbf{x}')$ as in Fig. 1.

Pairwise Logistic Loss. Logistic loss can also be interpreted as pairwise margin based loss following the same approach as in hinge loss. The loss is defined as:

$$\underset{\text{logistic}_{Pr}}{\mathcal{L}} = \sum_{\mathbf{x} \in X^+} \sum_{\mathbf{x}' \in X^-} \log(1 + \exp(f(\mathbf{x}') - f(\mathbf{x}))),$$

where the objective is to minimise marginal difference between negative and positive scores with a smoother linear slope than hinge loss as shown in Fig. 1.

2.3 KGE Multi-Class Losses

Recent KGE approaches have addressed the ranking problem as a multi-class classification. Next, we discuss how this is done.

Binary Cross Entropy Loss. ConvE model [5] proposed a new binary cross entropy multi-class loss to model its training error. In this setting, the whole vocabulary of entities is used to train each positive fact such that for a triple

(s, p, o), all facts (s, p, o') with $o' \in E$ and $o' \neq o$ are considered false. Despite the extra computational cost of this approach, it allowed ConvE to generalise over a larger sample of negative instances and outperform other approaches [5].

Negative-Log Softmax Loss. In a recent work, Lacroix et. al. [10] introduced a softmax regression loss to model training error of the ComplEx model as a multi-class problem. In this approach, the objective for each triple $\mathbf{x} = (s, p, o)$ is to minimise the following loss:

$$\begin{aligned} \mathcal{L}_{spo}^{\text{softmax}} &= \mathcal{L}_{spo}^{o'} + \mathcal{L}_{spo}^{s'} \text{ , s.t.} \\ \mathcal{L}_{spo}^{o'} &= -f_{spo} + \log(\sum\nolimits_{o'} \exp(f_{spo'})) \\ \mathcal{L}_{spo}^{s'} &= -f_{spo} + \log(\sum\nolimits_{s'} \exp(f_{s'po})) \end{aligned} \tag{1}$$

where $s' \in E$, $s' \neq s$, $o' \in E$ and $o' \neq o$. This resembles a log-loss of the softmax value of the positive triple compared to all possible object and subject corruptions, where the objective is to maximise positive facts scores and minimise all other scores. This approach achieved significant improvement to the prediction accuracy of ComplEx model over all benchmark datasets when used with the 3-nuclear norm regularisation of embeddings [10].

2.4 Negative Sampling for KGE Losses

In learning to rank approaches, models use a ranking loss, *e.g.*, pointwise or pairwise loss to rank a set of true and negative instances [4], where negative instances are generated by corrupting true training facts with a ratio of negative to positive instances [3]. This corruption happens by changing either the subject or object of the true triple instance. In this configuration, the ratio of negative to positive instances is traditionally learnt using a grid search, where models compromise between the accuracy achieved by increasing the ratio and the runtime required for training.

On the other hand, multi-class based models train to rank positive triples against their all possible corruptions as a multi-class problem where the range of classes is the set of all entities. For example, training on a triple (s, p, o) is achieved by learning the right classes "s" and "o" for the pairs $(?, p, o)$ and $(s, p, ?)$, where the set of possible classes is E of size N_e. Despite the enhancements of predictions accuracy achieved by such approaches [5,10], such negative sampling procedure is exhaustive and require high space complexity due to the usage of the entire entity vocabulary for each triple.

3 Experiments

In this section, we describe the experiments conducted on three state-of-the-art KGE models, namely, TransE [3], DistMult [22] and ComplEx [19] (equivalent to HolE [9]), using the previously discussed loss functions. TransE is a distance-based scoring function, while DistMult and ComplEx are multiplicative scoring functions.

Table 1. Characteristics of the datasets.

Dataset	# Entities	# Relations	|Train|	|Valid|	|Test|
WN18	41k	18	141k	5k	5k
WN18RR	41k	11	87k	3k	3k
NELL50k	50k	497	159k	5k	5k
NELL239	48k	239	74k	3k	3k
FB15k-237	15k	237	272k	18k	20k

We present the benchmarking datasets, experiments setup, and implementation details including software and hardware configurations.

Benchmarking Datasets. In our experiments we use six knowledge graph benchmark datasets:

- WN18 & WN18RR: subsets of Wordnet dataset [11] that contain lexical information of the English language [3,5].
- NELL50k & NELL239: subsets of NELL dataset [6,7] that we have created to test our model, which contains general knowledge about people, places, teams, universities, etc.
- FB15k-237: a subset of the Freebase dataset [1] that contains information about general human knowledge [17].

Table 1 contains the characteristics of our benchmark datasets[3].

Evaluation Protocol. The three KGE models are evaluated using a unified protocol that assesses their performance in the task of link prediction. Let X be the set of facts (triples), Θ_E be the embeddings of entities E, and Θ_R be the embeddings of relations R. The KGE evaluation protocol works in three steps:

(1) *Corruption*: For each $\mathbf{x} = (s, p, o) \in X$, $\mathbf{x}$ is corrupted $2|E| - 1$ times by replacing its subject and object entities with all the other entities in E. The corrupted triples can be defined as:

$$\mathbf{x}_{\text{corr}} = \bigcup_{s' \in E} (s', p, o) \cup \bigcup_{o' \in E} (s, p, o')$$

where $s' \neq s$ and $o' \neq o$. These corruptions are considered effectively negative examples for the supervised training and testing process under the Local Closed World Assumption [13].

(2) *Scoring*: Both original triples and corrupted instances are evaluated using a model-dependent scoring function. This process involves looking up embeddings of entities and relations, and computing scores depending on these embeddings.

(3) *Evaluation*: Each triple and its corresponding corruption triples are evaluated using the reciprocal ranking metric as a separate query, where the original triples represent true objects and their corruptions false ones. It is possible that corruptions of triples may contain positive instances that exist

[3] All the benchmark datasets and experimental results are available for download in the following url: `https://figshare.com/s/8c2f1e1f98aff44b5b71`

Table 2. Link prediction results for KGE models with different loss functions on standard benchmark datasets. (*) represents the models' default loss function. In the ranking losses, best results are computed per model: bold results represent the model's best result and underlined results represent the best result in a loss approach. In multi-class losses, best results are computed across all models.

	Model	Approach	Loss	WN18		WN18RR		NELL50k		NELL239		Fb15k-237	
				MRR	H@10	MRR	H@10	MRR	H@10	MRR	H@10	MRR	H@10
Ranking Loss	TransE	Pairwise	Hinge *	0.52	**0.95**	0.20	0.47	**0.76**	**0.91**	**0.28**	**0.43**	**0.27**	**0.43**
			Logistic	**0.53**	0.92	**0.21**	**0.48**	0.71	0.86	0.27	**0.43**	0.26	**0.43**
		Pointwise	Hinge	0.15	0.38	0.12	0.34	0.28	0.40	0.19	0.32	0.12	0.25
			Logistic	0.14	0.36	0.11	0.31	0.26	0.38	0.17	0.31	0.01	0.23
			SE	0.00	0.00	0.00	0.00	0.01	0.01	0.01	0.02	0.01	0.01
			PSL	0.20	0.49	0.12	0.33	0.38	0.54	0.19	0.33	0.11	0.24
	DistMult	Pairwise	Hinge *	0.77	0.89	0.40	0.45	0.45	0.60	0.20	0.32	0.10	0.16
			Logistic	0.79	0.93	0.39	0.45	0.68	0.83	0.26	0.40	0.19	0.36
		Pointwise	Hinge	**0.85**	0.95	**0.43**	0.49	0.81	0.92	0.25	0.41	0.21	0.39
			Logistic	0.77	0.93	**0.43**	**0.50**	0.70	0.84	0.28	0.43	0.20	0.39
			SE	0.81	**0.95**	**0.43**	**0.50**	0.81	**0.94**	**0.31**	**0.48**	**0.22**	**0.40**
			PSL	**0.85**	**0.95**	0.40	0.46	**0.85**	**0.94**	0.24	0.40	0.20	0.38
	ComplEx	Pairwise	Hinge	0.94	0.95	0.39	0.45	0.86	0.94	0.24	0.38	0.20	0.35
			Logistic	0.91	0.95	0.41	0.47	0.72	0.86	0.27	0.43	0.19	0.35
		Pointwise	Hinge	0.91	0.95	0.41	0.47	0.86	0.95	0.21	0.36	0.20	0.39
			Logistic *	0.94	0.95	0.36	0.39	0.85	0.94	0.14	0.24	0.13	0.28
			SE	**0.95**	**0.96**	**0.47**	**0.53**	0.82	0.94	**0.35**	**0.52**	0.22	0.41
			PSL	0.94	0.95	0.41	0.45	**0.90**	**0.96**	0.24	0.40	**0.24**	**0.43**
Multi-class loss	CP	-	BCE	-	-	-	-	-	-	-	-	-	-
		-	Softmax	0.12	0.18	0.08	0.12	-	-	-	-	0.22	0.42
	DistMult	-	BCE	0.82	0.94	0.43	0.49	-	-	-	-	0.24	0.42
		-	Softmax	0.81	0.95	0.43	0.50	0.91	0.96	0.39	0.55	0.34	**0.53**
	ComplEx	-	BCE	**0.94**	**0.95**	**0.44**	0.51	-	-	-	-	0.25	0.43
		-	Softmax	0.92	0.95	**0.44**	**0.52**	**0.94**	**0.97**	**0.40**	**0.58**	**0.35**	**0.53**

among training or validation triples. In our experiments, we alleviate this problem by filtering out positive instances in the triple corruptions. Therefore, MRR and Hits@k are computed using the knowledge graph original triples and non-positive corruptions [3].

Implementation. We use TensorFlow framework (GPU) along with Python 3.5 to implement the KGE models. Experiments were executed on a Linux machine with processor Intel(R) Core(TM) i70.4790K CPU @ 4.00GHz, 32 GB RAM, and an nVidia Titan X GPU.

Experimental Setup. In the experiments, we use state-of-the-art KGE models TransE, DistMult, and ComplEx to analyse the impact of various loss functions. We run these models over the previously mentioned benchmark datasets. A grid search was performed to obtain the best hyperparameters for each model[4]. In all our experiments, the set of investigated parameters are: embeddings size $K \in \{50, 100, 150, 200\}$ and margin $\lambda \in \{1, 2, 3, 4, 5\}$. We use a fixed learning rate of 0.1 and generate two corruptions per triple during training. All embeddings

[4] Detailed results and best hyperparameters can be found at: `https://figshare.com/s/8c2f1e1f98aff44b5b71`

vectors of our models are initialised using the uniform Xavier random initialiser [8]. We use 10 mini-batches per epoch, with a maximum of 1,000 epochs for training. We implemented early stopping for the training with a target MRR metric that is checked every 50 epochs (*i.e.*, training stops if the filtered MRR decreases).

4 Results and Discussion

Table 2 shows evaluation results for KGE models using different loss functions on standard benchmark datasets.

4.1 Ranking Losses

The results clearly show that changing the models' default loss functions can improve the reported performance of the KGE models. Moreover, the loss function we have proposed, PSL, enhances models' performance on multiple datasets. For example, the DistMult model uses pairwise hinge loss by default, but its version with the PSL function achieve 1.2% and 7.1% better MRR scores on WN18 and NELL50k datasets, and its version with SE loss provides best result on the other datasets. On the other hand, ComplEx originally uses pointwise logistic loss, but its version with SE loss results in better MRR score on WN18, WN18RR and NELL239 datasets. ComplEx using PSL version achieves the best results in terms of MRR on the NELL50k and FB15k-237 datasets.

In addition to confirming our main assumption, the results provide for an interesting observation. The versions of the TransE model with pairwise loss functions consistently achieve better results in terms of mean rank, MRR, and Hits@k when compared to the versions with pointwise losses. Conversely, the DistMult and ComplEx models achieve the best MRR and Hits@k scores when pointwise losses are used. This behaviour is likely caused by the fact that the models use different scoring approaches: TransE scores triples using distances in the embedding space, but DistMult and ComplEx use a multiplicative approach. This observation may be used for designing optimal combinations of scoring and cost functions in future KGE models. However, for a truly comprehensive recommendation, more thorough analysis of other distance-based and multiplicative scoring functions is required.

In terms of the type of cost function, the results show that the models with our proposed pointwise square loss (PSL) function outperform their versions with other pointwise losses (the MRR score is better in 7 out of 15 experiment configurations of TransE, DistMult, and ComplEx models on all datasets).

An important technical observation is that the number of configurable parameters of a loss function has a significant impact on the time required for training the corresponding model. The training time grows exponentially with respect to the number of hyperparameters used in training. Pointwise SE and both pointwise and pairwise logistic losses do not have configurable parameters, therefore they require minimal training time when compared to other losses with additional configurable parameters. Even the margin based loss functions that require only one parameter, λ, have significantly slower training time than SE

and logistic losses. In our experiments, models with configurable margin losses required 5 times more training time than model using losses with no configurable parameters as we searched for best margin λ in a set of five elements.

In ranking loss functions, the differences in evaluation accuracy of models using different loss functions can be sometimes relatively small. In real world large scale knowledge graph applications, the choice of training loss function for a KGE model will therefore always involve a compromise between both evaluation accuracy and training time efficiency.

4.2 Multi-Class Losses

Results of multi-class loss shows that models' versions with negative-log softmax loss outperform their versions with BCE loss over all datasets. Also, it shows that multi-class loss can provide significant improvement in terms of MRR over ranking losses as on NELL239 and FB15k-237 datasets.

Despite the enhancements of predictions accuracy achieved by multi-class loss approaches [5,10], they can have scalability issues in real-world knowledge graphs with a large number of entities as they use the full entities vocabulary as negative instances [12].

5 Conclusions and Future Work

In summary, our results clearly confirm all our key assumptions. First of all, the choice of a loss function does have a considerable impact on the performance of KGE models. Secondly, loss functions can be consciously selected in a way that can optimise particular evaluation metrics. This marks a big improvement over state-of-the-art approaches where the cost functions have been selected in a rather non-systematic way. Last but not least, we have brought up several interesting observations that can inform more rational and efficient design of future KGE models.

For future work, we intend to experiment with models that use a sampled multi-class approach, *i.e.*, they sample negatives as a portion of the vocabulary rather that the whole vocabulary. We also aim to study the different properties of KGs and their effects on the performance of KGE models.

References

1. Kurt D. Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD Conference*, pages 1247–1250. ACM, 2008.
2. Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. A semantic matching energy function for learning with multi-relational data - application to word-sense disambiguation. *Machine Learning*, 94(2):233–259, 2014.
3. Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NIPS*, pages 2787–2795, 2013.

4. Wei Chen, Tie-Yan Liu, Yanyan Lan, Zhiming Ma, and Hang Li. Ranking measures and loss functions in learning to rank. In *NIPS*, pages 315–323. Curran Associates, Inc., 2009.
5. Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *AAAI*. AAAI Press, 2018.
6. Tom M. Mitchell et. al. Never-ending learning. *Commun. ACM*, 61(5):103–115, 2018.
7. Matt Gardner and Tom M. Mitchell. Efficient and expressive knowledge base completion using subgraph feature extraction. In *EMNLP*, pages 1488–1498. The Association for Computational Linguistics, 2015.
8. Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, volume 9 of *JMLR Proceedings*, pages 249–256. JMLR.org, 2010.
9. Katsuhiko Hayashi and Masashi Shimbo. On the equivalence of holographic and complex embeddings for link prediction. In *ACL (2)*, pages 554–559. Association for Computational Linguistics, 2017.
10. Timothée Lacroix, Nicolas Usunier, and Guillaume Obozinski. Canonical tensor decomposition for knowledge base completion. In *ICML*, volume 80 of *JMLR Workshop and Conference Proceedings*, pages 2869–2878. JMLR.org, 2018.
11. George A. Miller. WordNet: A lexical database for english. *Commun. ACM*, 38(11):39–41, 1995.
12. Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In *NIPS*, pages 2265–2273, 2013.
13. Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2016.
14. Maximilian Nickel, Lorenzo Rosasco, and Tomaso A. Poggio. Holographic embeddings of knowledge graphs. In *AAAI*, pages 1955–1961. AAAI Press, 2016.
15. Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *ICML*, pages 809–816. Omnipress, 2011.
16. Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. Factorizing YAGO: scalable machine learning for linked data. In *WWW*, pages 271–280. ACM, 2012.
17. Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. Representing text for joint embedding of text and knowledge bases. In *EMNLP*, pages 1499–1509. ACL, 2015.
18. Théo Trouillon and Maximilian Nickel. Complex and holographic embeddings of knowledge graphs: A comparison. *CoRR*, abs/1707.01475, 2017.
19. Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *ICML*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org, 2016.
20. Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Trans. Knowl. Data Eng.*, 29(12):2724–2743, 2017.
21. Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, pages 1112–1119. AAAI Press, 2014.
22. Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*, 2015.