Let the Margin SlidE$^{\pm}$ for Knowledge Graph Embeddings via a Correntropy Objective Function

*

Mojtaba Nayyeri
SDA Research, University of Bonn
Bonn, Germany
nayyeri@cs.uni-bonn.de

Xiaotian Zhou
SDA Research, University of Bonn
Bonn, Germany
s6xizhou@uni-bonn.de

Sahar Vahdati
University of Oxford
Oxford, UK
sahar.vahdati@cs.ox.ac.uk

Reza Izanloo
Ferdowsi University of Mashhad
Mashhad, Iran
rezaizanloo$_8$8@$mail.um.ac.ir$

Hamed Shariat Yazdi
SDA Research, University of Bonn
Bonn, Germany
Shariat@cs.uni-bonn.de

Jens Lehmann
SDA Research, University of Bonn
Fraunhofer IAIS, Bonn, Germany
jens.lehmann@iais.fraunhofer.de

***Abstract*—Embedding models based on translation and rotation have gained significant attention in link prediction tasks for knowledge graphs. Most of the earlier works have modified the score function of Knowledge Graph Embedding models in order to improve the performance of link prediction tasks. However, as proven theoretically and experimentally, the performance of such Embedding models strongly depends on the loss function. One of the prominent approaches in defining loss functions is to set a margin between positive and negative samples during the learning process. This task is particularly important because it directly affects the learning and ranking of triples and ultimately defines the final output. Approaches for setting a margin have the following challenges: a) the length of the margin has to be fixed manually, b) without a fixed point for center of the margin, the scores of positive triples are not necessarily enforced to be sufficiently small to fulfill the translation/rotation from head to tail by using the relation vector. In this paper, we propose a family of loss functions dubbed SlidE$^{\pm}$ to address the aforementioned challenges. The formulation of the proposed loss functions enables an automated technique to adjust the length of the margin adaptive to a defined center. In our experiments on a set of standard benchmark datasets including Freebase and WordNet, the effectiveness of our approach is confirmed for training Knowledge Graph Embedding models, specifically TransE and RotatE as a case study, on link prediction tasks.**

***Index Terms*—Graph Embedding, Loss Function, Margin Ranking Loss, Statistical Relational Learning**

I. Introduction

Knowledge graphs are one of the most important technologies for the next wave of artificial intelligence and knowledge management solutions across industrial applications [1], [3], [24]. This is evident by a broad range of use cases of KGs ranging from question answering [5], [12], [13], recommendation systems [31], semantic modeling [25] to data analysis [19], and knowledge management systems [11], [27]. To support such intelligent applications, various large-scale knowledge graphs have been made available. Some of the most used knowledge graphs are WordNet [21], Freebase [4], NELL [10], Yago [23] and DBpedia [17] and Wikidata [29]. These datasets include knowledge in multi-relational directed graphs composed of nodes $\mathcal{E}$ (usually called entities) and edges $\mathcal{R}$ (usually called links or relations). More precisely, a $\mathcal{KG}$ includes a set of triples in the form of (head, relation, tail) denoted as (h, r, t) where h, t refer to the subject (also called head) and object (also called tail) respectively and r refers to a relation e.g., (Paris, isCapitalOf, France). This representation of information empowers navigation across information and provides an effective utilization of encoded knowledge. Since it is difficult to capture all the existing knowledge from the real world, knowledge graphs are usually incomplete. This limits the inference of knowledge and influences performance of the systems utilizing such KGs. An elegant solution to solve the incompleteness of KGs are "Knowledge Graph Embeddings (KGE)". Those embeddings assign a latent feature vector to each node and relation in a KG, which can then be used in downstream machine learning tasks such as link prediction. Among the proposed KGE methods, translation-based models are considered as a key family of methods for graph completion tasks. Recently a new generation of KGEs have been proposed using rotation-based techniques. Our approach is designed to cover both types of translation and rotation-based models. Translation-based models encode entities as vectors and relationships between entities as translation vectors. TransE [7] is one of the primary models that seeks for a latent feature vector representation of a given triple (h, r, t) in which the vector representing t is same as the sum of the vectors representing h and r. Initially, the corresponding vectors (**h**, **r**, **t**) of each individual triple (h, r, t) are randomly distributed over the vector space. An embedding model employs a scoring function and a loss function in order to (approximately) satisfy $\mathbf{h} + \mathbf{r} \simeq \mathbf{t}$ for positive triples (h, r, t), and $\mathbf{h}' + \mathbf{r} \neq \mathbf{t}'$ for negative samples of (h', r, t'). The correctness of a (h, r, t) triple is calculated via a scoring

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

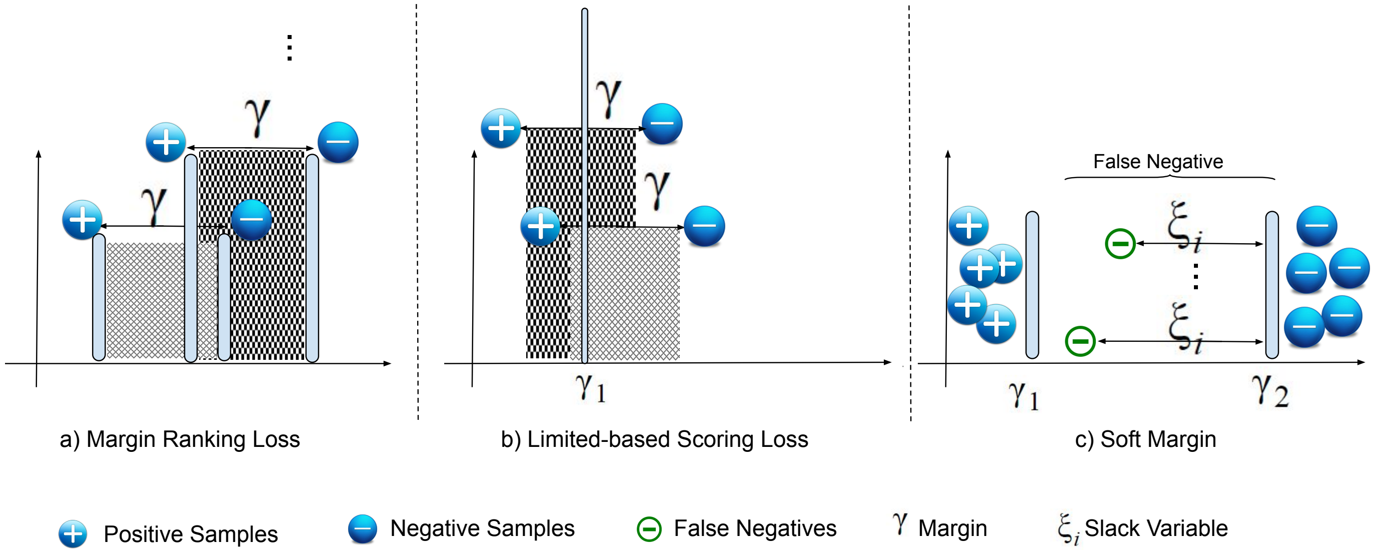


Fig. 1: **Loss Functions of KGEs.** Each sub-figure represents the strategy of sliding margin for positive and negative samples in a loss function of a KGE model. In *a*, we show the strategy of MRL in distinguishing positive and negative samples with a fixed margin of (γ) between two triples (lower hatched part and upper hatched part). For Limited-based Scoring loss in *b*, it is shown how the loss function bounds the positive triples by (γ_1). In *c*, we show the strategy of Soft Margin loss for separating positive and negative samples with two fixed γ_1 and γ_2. The slack variable ξ_i allows uncertain negative samples to slide inside the margin.

function in the embedding space such as $f_r(h,t) = \|\mathbf{h}+\mathbf{r}-\mathbf{t}\|$. Since the vectors for positive and negative (corrupted) triples are randomly distributed, the results of the scoring function also evaluates their correctness randomly as well. Therefore, a loss function (e.g. the Margin Ranking Loss) is needed to optimize the embedding vectors of entities and relations.

Margin Ranking Losses (MRLs) are widely accepted and used in embedding models and their effectiveness is proven [7], [28]. The margin-based ranking loss function forces the score of positive triples to be lower (towards 0) and assigns a higher score to negative triples by a margin of at least γ. Therefore, positive triples are separated from negative samples. However, using MRL includes the existence of cases where the score of a correct triple (h,r,t) is not sufficiently small to hold $\mathbf{h}+\mathbf{r} \simeq \mathbf{t}$. A combination of limit-based scoring loss functions for a set of translation-based embedding models [32] have been proposed in order to avoid such cases. By adding a limit of $f_r(h,r) \leq \gamma_1$, the score of correct triples is bounded within a determined range. However, the setting of γ (length of margin) and γ_1 (upper-bound of positive triples) in alignment with the score of the positive and negative triples is done with a "trial and error" method in a very big search space. Due to the lack of a unique answer for γ and γ_1 and the large search space, this task can be multiplied for any possible variation in ranking. In this work, we propose SlidE$^{\pm}$ as a family of loss function addressing both translation and rotation-based embedding models[1]. Two subsequent loss functions have been derived based on an expansion (+) and contraction (-) techniques. In order to show the effectiveness of our proposed loss function, through a systematic analysis, we selected TransE model as a baseline and the RotatE model as the recently proposed KGE model. Our method reduces the search of two hyperparameters (γ_1, γ) to one variable (γ). γ is the center of the margin that should be searched within a set of finite numbers. The margin is adjusted automatically during the learning phase by formulating a slack variable in the optimization problem.

The remaining part of this paper proceeds as follow. Section II represents the related work and previous proposals developed for loss functions of Translation-based embeddings. Section III provides a detailed description of the adaptive model. An evaluation of the newly developed loss function is shown in Section IV. In Section V, we lay out the insights and provide conjunction of this research work.

II. Related Work

The loss function has a significant impact on the performance of translation-based embedding models [2], [32]. Defining a margin to separate positive and negative triples is one of the promising solutions in keeping a high performance for loss functions. Therefore, approaches focusing on a proper adjustment for such a margin in the loss function became

[1] The codes are made available in: https://github.com/mojtabanayyeri/KGE-SlidE-Loss

important in translation-based KGEs. Here, we introduce three of the main proposed margin-based ranking loss functions. An illustration of each loss function is shown in Figure 1. A 2-dimensional illustration is used in order to visualize how the underlying loss function optimizes the distribution of negative and positive triples. The X axis represents the scores of triples and the Y axis is added for visual reasons without which all the triples should have been shown on the X axis.

A. Margin Ranking Loss

Margin Ranking Loss (MRL) is one of the primary approaches that was proposed to set a margin of γ between positive and negative samples. It is defined as follows:

$$\mathcal{L} = \sum_{(h,r,t)\in S^{+}} \sum_{(h',r,t')\in S^{-}} [f_r(h,t) + \gamma - f_r(h',t')]_{+} \tag{1}$$

where $[x]_{+} = \max(0,x)$ (S^{+} for positive samples and S^{-} for negative samples). S^{-} includes training samples with two patterns of triples: 1) a corrupted head replaced by a random entity for a fixed tail , 2) for a fixed head, a corrupted tail is replaced by a random entity. The score of any such corrupted triple $f_r(h',t')$ in negative samples is forced to be higher than the positive triples $f_r(h,t)$ with a margin of γ. The loss function assigns scores to the triples in a way that $f_r(h',t') - f_r(h,t) \geq \gamma$ holds. However, this loss function does not guarantee that the scores assigned to the positive samples are low enough to present the correct translation (i.e. $\mathbf{h}+\mathbf{r} \simeq \mathbf{t}$). For example, for an initial $\gamma = 1$, the model forcing to hold this condition possibly assigns scores for the following positive and negative samples:

$$\begin{cases} (f_r(h',t') = 1) - (f_r(h,r) = 0) \geq (\gamma = 1) \\ (f_r(h',t') = 11) - (f_r(h,r) = 10) \geq (\gamma = 1) \\ (f_r(h',t') = 101) - (f_r(h,r) = 100) \geq (\gamma = 1) \\ (f_r(h',t') = 1001) - (f_r(h,r) = 1000) \geq (\gamma = 1) \end{cases} \tag{2}$$

Although the calculated loss is the same number for each of these examples, the score scale of the latter sample is higher than the first one. This makes the positive training triples with high scores hardly meeting the conditions of $\mathbf{h} + \mathbf{r} \simeq \mathbf{t}$, illustrated as Margin Ranking Loss in Figure 1. Thus, with such a loss function, it is possible that the model produces ineffective results.

B. Limited-based Scoring Loss

In order to fulfill the gap of MRL in assigning high scores to positive samples, a limited-based scoring function has been proposed [32]. This method limits the score of positive samples by adding an upper-bound (γ_1). It is represented as limited-based scoring loss illustrated in Figure 1. In this way, the scores of positive samples are forced to stay before the upper bound which significantly improves the performance of translation-based KGE models [2], [22], [32]. In [32], the MRL is revised by adding a term ($[f_r(h,t) - \gamma_1]_{+}$) to limit maximum value of positive score:

$$\mathcal{L}_{RS} = \sum\sum [f_r(h,t) + \gamma - f_r(h',t')]_{+} + \lambda[f_r(h,t) - \gamma_1]_{+} \tag{3}$$

The possible combination of variables for γ and γ_1 is wide with a complexity of n^2 where n is possible number of values in the search space. Considering that, the setting of (γ, γ_1) is yet a manual task in experiments. The model and the results suffer from the difficulty of finding an optimum setting by trying all the possible combinations of (γ, γ_1).

C. Soft Margin

A modified version of the two previous loss functions is introduced in our previous work [22]. This approach fixes the upper-bound of positive samples (γ_1) and uses a sliding mechanism to move false negative samples towards positive samples, shown as Soft Margin in Figure 1. θ refers to embedding parameters of all entities and relations in KG as (**h**, **r**, **t**). A slack variable is used per each triple (i.e. ξ_i, where i refers to the i-th triple) to enable false negative samples to move inside the margin.

$$\begin{aligned} \min_{\xi^{r}_{h,t},\theta} \sum_{(h,r,t)\in S^{+}} \lambda\, {\xi^{r}_{h,t}}^{2} + \lambda_{+}[f_r(h,t) - \gamma_1]_{+} + \\ \lambda_{-}\, [\gamma_2 - f_r(h',t') - \xi^{r}_{h,t}]_{+} \end{aligned} \tag{4}$$

In order to properly adjust margin, two variables (γ_1, γ_2) should be obtained. Experiments show that the performance of KGE models improves significantly by using different values for (γ_1, γ_2). Assuming γ_1 in the range of 10 possible variables $\{0, 0.5, 1, \ldots, 4.5\}$ and γ_2 in another range of 10 possibilities such as $\{0.5, 1, 1.5, \ldots, 5\}$ result in 10^2 variations for combination of (γ_1, γ_2). The setting of (γ_1, γ_2) is yet a manual task in experiments, the model and the results suffer from the difficulty of finding an optimum setting by trying all possible combinations. The results are promising with a focus on handling uncertainty in negative sampling (false negative samples). However, a correct setting of γ_2 in alignment with γ_1 still remains challenging in the performance and effectiveness of the model.

III. The Family of SlidE$^{\pm}$ Loss Functions

With the family of SlidE$^{\pm}$ Loss functions, we aim at reducing the search space to set the margin between positive and negative triples. We use a variable (γ) denoting the center of the margin. As a result, instead of searching for two parameters (γ_1, γ_2) in limited-based scoring loss, we search for one parameter (γ) illustrated in Figure 2. We propose two separate loss functions to obtain the margin automatically. One of the loss functions is using *expansion* approach (denoted by $\mathcal{L}_{SlidE^{+}}$) and the other uses *contraction* (denoted by $\mathcal{L}_{SlidE^{-}}$). The **expansion** method gradually increases the margin from zero to a bigger value. In the other method, for **contraction**, the margin shrinks from bigger values to smaller ones. These two methods are independent and are for solitary usage. The performance of each method depends on the application area and the general status of the KG and the underlying model. The authors leave the decision of using contraction or expansion methods on users based on the best performance of each loss in the defined embedding problem.

A slack variable ($\xi \geq 0$) is employed to gradually expand (or contract) the margin i.e. $\gamma_1 = \gamma - \xi, \gamma_2 = \gamma + \xi$. Therefore, the following inequalities should hold for positive and negative scores:

$$\begin{cases} f_r(h,t) \leq \gamma - \xi, \\ f_r(h^{'},t^{'}) \geq \gamma + \xi. \end{cases} \tag{5}$$

Instead of using one slack variable per triples (as it was in Soft margin), we propose to use one slack variable to adapt the margin by expansion or contraction. In order to enforce the model to satisfy Equation 5, the following penalty terms are derived to be included in the proposed optimization problem. Therefore, the loss functions of positive ($Loss^{+}$) and negative ($Loss^{-}$) samples are derived as follows:

$$\begin{cases} Loss^{+} = [f_r(h,t) - \gamma + \xi]_+ = Relu(f_r(h,t) - \gamma + \xi) \\ Loss^{-} = [-f_r(h^{'},t^{'}) + \gamma + \xi]_+ = Relu(-f_r(h^{'},t^{'}) \\ \qquad\qquad +\gamma + \xi). \end{cases} \tag{6}$$

The optimization problem is defined as following from which the two losses of expansion and contraction are derived:

$$L = \lambda_+ \, Loss^{+} + \lambda_- \, Loss^{-} \tag{7}$$

where λ_+ and λ_- are hyperparameters, making a trade-off between positive and negative losses. In our experiments, we set these parameters to 1 for simplicity.

The role of ξ is to derive the margin. It is initialized in the beginning of the algorithm, $\xi = m$ for expansion ($\xi = \mathcal{M}$ for contraction). The initial value of the margin is introduced in Equation 8:

$$\gamma_2 - \gamma_1 = \gamma + \xi - \gamma + \xi = 2\xi = 0\,(2\mathcal{M}). \tag{8}$$

In the following sections, we introduce the expansion and contraction approaches in details.

A. *Slide*$^{+}$*: The Loss with Expansion*

In the expansion approach, the margin is initialized with a very small value (e.g. zero). Then during the optimization process, margin is expanded automatically by sliding the edges of the margin. We employ *correntropy objective function* to enable the margin to be expanded. This step is done by increasing the value of ξ. The correntropy objective function is defined as $\mathcal{C}(\xi) = E(K(\xi))$ [20] where $E(.)$ is the expectation in probability theory, $K(.)$ is a kernel function and $\xi \in R^d$ is a $d-$dimensional random variable. Typically, Gaussian kernels are used in the correntropy function. A Gaussian kernel is defined as $K(\xi) = e^{-\sigma \|\xi\|^2}$.

Assuming $\xi \in R$, which is a number rather than a vector, the following part is added to the loss L (Equation 7):

$$L_\xi = e^{-\sigma\xi^2}. \tag{9}$$

In the original Equation 5 on which the expansion will be formulated, ξ should be a positive value. In order to ensure this, we use ξ^2 instead of ξ in the final formulation of the loss function:

$$\begin{aligned} & min_{\theta,\xi} \;\; e^{-\sigma\xi^2} \\ & \text{subject to} \qquad f_r(h,t) \leq \gamma - \xi^2, \\ & \qquad\qquad\qquad f_r(h^{'},t^{'}) \geq \gamma + \xi^2 \,. \end{aligned} \tag{10}$$

Using penalty method and considering Equation 7, Equation 9, instead of solving Equation 10, the following loss function is minimized:

$$\begin{aligned} \mathcal{L}_{SlidE^+} = \lambda e^{-\sigma\xi^2} + \lambda_+ \, [f_r(h,t) - \gamma + \xi]_+ + \\ \lambda_- \, [-f_r(h^{'},t^{'}) + \gamma + \xi]_+. \end{aligned} \tag{11}$$

By initializing ξ to 0, the amount of loss in Equation 9 becomes 1 (it is maximized). The minimization of the main loss (Equation 11) is realized when ξ is enforced to be increased. In theory this happens when $e^{-\xi^2} \to 0$ which holds when $\xi^2 \to \infty$. In practice, we solved the optimization using stochastic gradient descent where ξ is enforced to reach a big value ($\mathcal{M}$). Therefore, as indicated in 8, the margin is expanded from 0 to $2\mathcal{M}$.

B. *Slide*$^{-}$*: The Loss with Contraction*

In the contraction approach, the loss is formulated in such a way that the margin starts with a big value and gradually shrinks. In order to formulate the loss function with contracted margin, the following formula is employed to be added to L (Equation 7):

$$L_\xi = \xi^2. \tag{12}$$

Therefore, considering Equation 12 and Equation 5, the following optimization is proposed:

$$\begin{aligned} & \min_{\theta,\xi} \; \xi^2 \\ & \text{subject to} \qquad f_r(h,t) \leq \gamma - \xi^2, \\ & \qquad\qquad\qquad f_r(h^{'},t^{'}) \geq \gamma + \xi^2 \,. \end{aligned} \tag{13}$$

As explained previously, the variable θ denotes embedding parameters.

Adding a penalty parameter multiplied by a measure of violation of constrains is a solution to solve such constrained problems [9]. Using penalty method (by adding $[f_r(h,t) - \gamma + \xi]_+ + \lambda_-$ and $[-f_r(h^{'},t^{'}) + \gamma + \xi]_+$) and considering Equation 7 and Equation 12, instead of solving Equation 13, the following loss function is minimized:

$$\begin{aligned} \mathcal{L}_{SlidE^-} = \lambda\xi^2 + \lambda_+ \, [f_r(h,t) - \gamma + \xi]_+ + \lambda_- \\ [-f_r(h^{'},t^{'}) + \gamma + \xi]_+ \end{aligned} \tag{14}$$

We emphasize on substitute usage of the *expansion* and *contraction* methods per use case. Although in our results,

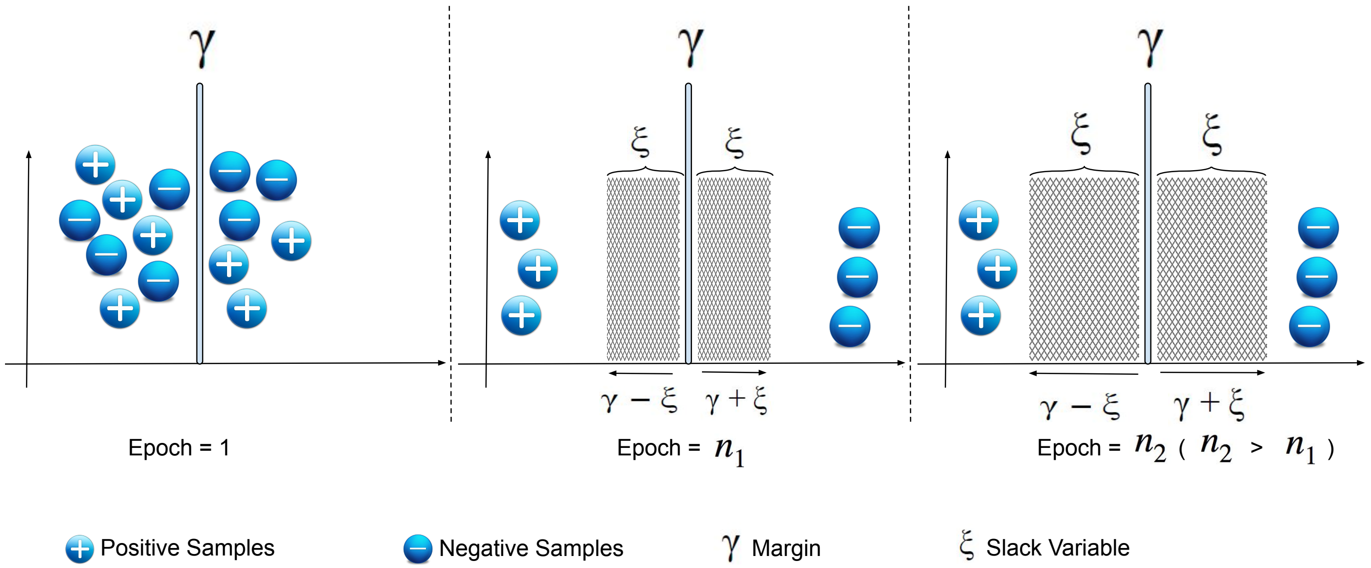


Fig. 2: **Illustration of loss function based on SlidE$^{\pm}$**. Our model uses a fixed center of margin and slides the edges on the side of positive and negative triples (the X axis represents rankings and the Y axis is for better illustration of the triple distribution). This is illustrated in three incremental epochs (left to right).

TABLE I: **Dataset Statistics.** Split of datasets.

Dataset	#training	#validation	#test
FB15k	483,142	50,000	59,071
WN18	141,442	5,000	5,000
FB15k-237	272,115	17,535	20,466
WN18RR	6,084	3,034	3,134

the contraction approach is outperformed by expansion in all aspects, our expectation is that the performance of each method can differ in various applications also based on the structure of the KG. We introduced the contraction for comprehensive proposal of our approach and report the results for transparency.

IV. Experiments

An evaluation of our proposed family of loss functions is addressed in this section. We mainly focused on training the TransE [7] and RotatE [26] models with the state-of-the-art loss functions and provided comparisons with SlidE$^{\pm}$. The main evaluation metrics for link prediction tasks are Mean Rank (MR) and Hits@K. To compute MR, two sets are generated ($S_L = \{(h, r, ?)\}, S_R = \{(?, r, t)\}$) for each test triples (h, r, t) where all entities in the KGs are replaced by $?$. Scores of all triples in S_L, S_R are computed and sorted. The rank of the original triple (i.e. (h, r, t)) is computed in both sets S_L, S_R which are respectively denoted by r_L, r_R. In any considered triple, r_L is the notation for the left ranks and r_R for the right ranks. The rank of the example triple of (h, r, t) is computed as $r = \frac{r_L + r_R}{2}$. In this way, MR is obtained by taking overall average rank of testing triples. Finally, the computation of Hits@10 is performed by counting the number of testing triples which are ranked less than 10 (i.e. $r_i \leq 10$).

A. Experimental Setup and Benchmarks

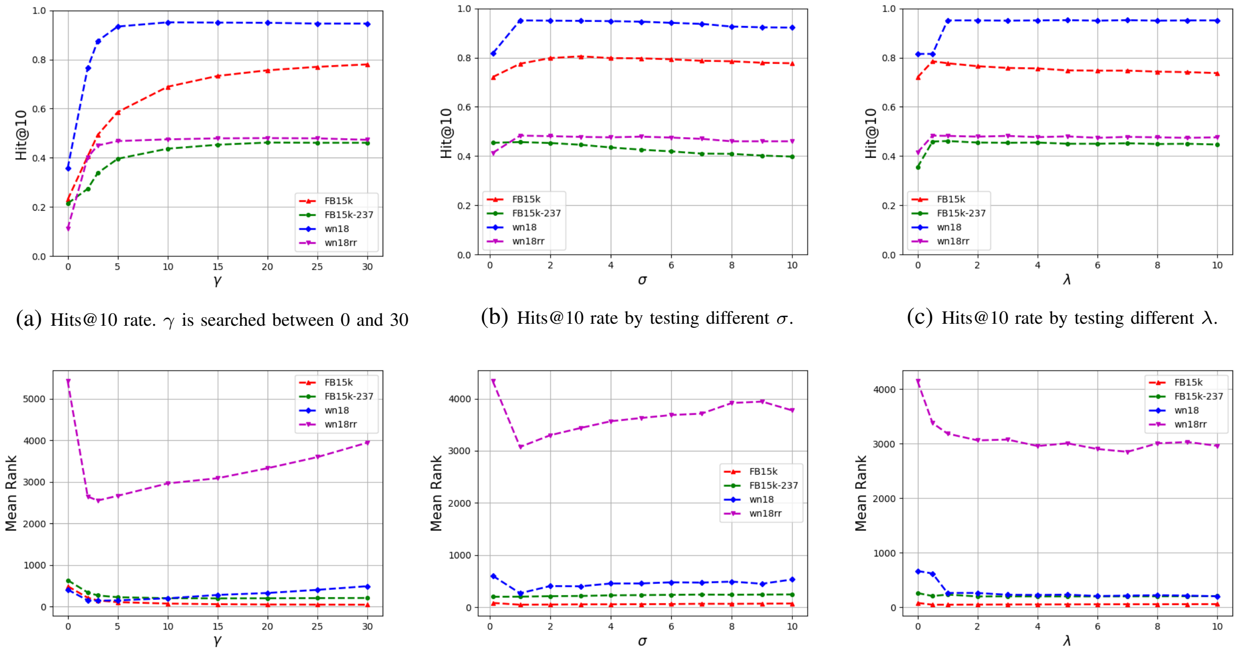
The TransE and RotatE models as well as our proposed loss functions can be trained with different settings on hyperparameters. For TransE and RotatE, embedding dimension (d) and a number of generated negative samples (n) per each positive are selected as the two hyperparameters. SlidE$^{\pm}$ has γ, λ_+, λ_- and σ as hyperparameters. TransE trained by margin ranking loss, Limited-Score Loss, soft margin loss, RotatE loss and SlidE$^{\pm}$ are denoted by TransE, TransE-RS, TransE-SM, TransE-RL, TransE-SlidE^{+} and TransE-SlidE^{-} respectively. We additionally train the recent state-of-the-art model (i.e., RotatE) with our loss to show the effectiveness of the proposed loss functions. Same naming is used for RotatE. We also compare the results with the ComplEx [28] model. The implementations of TransE and RotatE with SlidE$^{\pm}$ have been done in Pytorch using Adagrad as optimizers. The model stops training when the accuracy of Hits@10 reaches a pick value and starts to grade down.

Four benchmark datasets have been considered for the evaluations. Table I lists the number of triples in training, test, and validation sets in each KG used in our experiments.

Less Sensitive Margins Batch sizes of 512 and 1024 are tested for each dataset. In order to investigate the core effectiveness of the proposed loss function and have a fair comparison, embedding dimension is set to 100 (Table II). Moreover, only one negative sample is generated per each positive sample. To reduce the number of parameters for searching, we set $\lambda_+, \lambda_-, \lambda$ and σ to 1, and only γ is tuned in the set $\{0, 1, 2, 3, 4, 5, 10, 15, 20, 25, 30\}$. We used the

TABLE II: **Link prediction results.** Comparison of models implemented with loss function of MRL, Limited-base loss, and SlidE$^{\pm}$ (expansion and contraction) considering Mean Rank, Hits@10 on WN18 and FB15k.

Dataset	WN18				FB15k			
	Mean Rank		Hits@10(%)		Mean Rank		Hits@10(%)	
	raw	filter	raw	filter	raw	filter	raw	filter
Unstructured [6]	315	304	36	38	1074	979	5	6
RESCAL [23]	1180	1163	37	53	828	683	28	44
SE [8]	1011	985	69	81	273	162	29	40
SME (linear) [6]	545	533	65	74	274	154	31	41
SME (bilinear) [6]	526	509	55	61	284	158	31	41
LMF [14]	469	456	71	82	283	164	26	33
TransE [7]	263	251	75	89	243	125	35	47
TransH (unif) [30]	318	303	75	87	211	84	43	59
TransH (bern) [30]	401	388	73	82	212	87	46	64
TransR (unif) [18]	232	219	78	92	226	78	44	66
TransR (bern) [18]	238	225	80	92	198	77	48	69
TransD (unif) [15]	242	229	79	93	211	67	49	74
TransD (bern) [15]	224	212	80	92	194	91	53	77
TransE-RS(unif) [32]	362	348	80	94	161	62	53	72
TransE-RS(bern) [32]	385	371	80	94	161	63	53	72
TransH-RS(unif) [32]	401	389	81	95	163	64	53	73
TransH-RS(bern) [32]	371	357	80	95	178	77	54	75
RotatE(dim=50) [26]	324	312	83	95	170	57	54	75
ComplEx(dim=50) [26]	596	584	79	88	168	68	48	64
RotatE-SlidE^{+}(dim=50)	**186**	**178**	**86**	**96**	**141**	**53**	**55**	78
TransE-SlidE^{+}	226	217	84	95	177	55	**55**	**81**
TransE-SlidE^{-}	380	488	80	90	169	68	50	71



(a) Hits@10 rate. γ is searched between 0 and 30

(b) Hits@10 rate by testing different σ.

(c) Hits@10 rate by testing different λ.

(d) Mean Rank. γ is searched between 0 and 30

(e) Mean Rank by testing different σ.

(f) Mean Rank by testing different λ.

Fig. 3: **Evaluation of TransE-SlidE^{+} by testing different values of one of the hyper parameters γ, σ, λ.**

TABLE III: **Link prediction results.** Comparison of models implemented with loss function of soft margin loss and SlidE$^{\pm}$ (expansion and contraction) considering Mean Rank, Hits@10 on WN18RR and FB15k-237.

Dataset	WN18RR				FB15k-237			
	Mean Rank		Hits@10(%)		Mean Rank		Hits@10(%)	
	raw	filter	raw	filter	raw	filter	raw	filter
ComplEx(dim=50)	5974	5963	43	45	367	243	30	39
TransE-SM	3941	3927	46	49	**365**	210	**32**	**46**
RotatE(dim=50)	4551	4538	49	53	371	235	**32**	43
RotatE-SlidE^{+} (dim=50)	3855	3766	**50**	**54**	396	243	**32**	43
TransE-SlidE^{+}	**3392**	**3377**	46	49	366	**203**	**32**	**46**
TransEAMLCont	4298	4285	39	41	469	264	26	38

TABLE IV: **Optimal Setting.** Representation of different setting considering hyperparameters for TransE-SlidE^{+} and TransE-SlidE^{-} (the rest of the models have been trained with their best settings in their own original resources).

Dataset	Expansion Approach				Contraction Approach			
	FB15k	FB15k-237	WN18	WN18RR	FB15k	FB15k-237	WN18	WN18RR
ξ initialization	0.1	0.1	0.1	0.1	10	10	10	10
γ	30	20	15	15	30	30	25	15
Learning rate	0.1	0.005	0.1	0.5	0.1	0.005	0.1	0.5

same hyper-parameter search for the RotatE loss. The slack variable ξ for expansion and contraction are initialized by 0.1 and 10. The optimal hyperparameters of TransE-SlidE^{+} and TransE-SlidE^{-} obtained for each dataset are reported in the Table IV. The parameters are obtained using validation set. The experimental datasets of evaluation includes FB15k, FB15k-237, WN18 and WN18RR.

Runtime Comparison Evaluation of our model with competitors depends on the number of computational operations. Our loss has a simple addition and subtraction. Due to better performance in accuracy, we consider the expansion loss for runtime evaluation. Our runtime is fairly close to other losses e.g., on WN18, training RotatE with its original loss function takes 26 minutes and 6 seconds whereas the training time of RotatE with SlidE^{+} is 26 minutes and 46 seconds. On this dataset, RotatE-SlidE^{+} improves the results of RotatE by 42% on filtered mean rank, 3% on raw Hits@10 and 1 on filtered Hits@10.

Boosting Technique Most of the KGE models present a proper formulations (in loss and score functions) with an objective of outperforming the results of state-of-the-art models. However, there are also some additional model-independent techniques which can influence and improve the results of KGE models regardless of the corresponding formulations. We denote these side-techniques as *boosting techniques*. Here we list some of these side-techniques used in performance boosting of KGE on the example of RotatE and Canonical Tensor Decomposition (CP) [16]: 1) increasing embedding dimension (EmbDim) (up to 1000 EmbDim in a complex area in RotatE), 2) increasing number of negative samples (NegSam) (1000 negatives per each positive in RotatE), 3) using different NegSam techniques (bern/ or adversarial NegSam in RotatE and CP), 4) pre-processing and enrichment of the datasets (e.g. adding reverse-triples to dataset in RotatE and CP).

In order to properly evaluate the efficiency of our proposed loss functions and justify the advantage of the proposed core formulation, we initially avoided considering such boosting techniques in the evaluation of our model in previous part (Table II Table III). That demonstrates the novelty and contribution of our work and proves that our performance gain is due to a more proper formulation. Moreover, such boosting-techniques require a powerful computational infrastructure, adversely limit their applicability. However, in order to further clarify the effect of boosting technique in the performance, we apply some of them in our model to have a comprehensive evaluation. We have increased the embedding dimension as well as the number of negative samples up to 300 and used adversarial negative sampling. The Table V shows the reported results using boosting techniques (with .BT suffix).

B. *Results and Discussion*

The results represented in Table II show comparisons of TransE-SlidE^{+} and TransE-SlidE^{-} with TransE-RS, TransH-RS, TransE, TransH, RotatE and ComplEx as well as RotatE-SlidE^{+}. Additionally, we compare our model to LMF, SME, SE, RESCAL and UNSTRUCTURED. As stated in [32], we are also using dimension $d = 100$ to provide an identical setting for our evaluation. Let us note that, some of the models are defined in complex vector space such as RotatE and ComplEx. Therefore, by setting the dimension to $d = 50$,

TABLE V: **Boosting techniques (BT).** The experiments are shown for the two datasets of WN18RR and FB15K-237 using boosting techniques.

Datasets	WN18RR		FB15K-237	
Metrics	MR	Hits@10	MR	Hits@10
RotatE.BT	3341	57	180	52
RotatE-SlidE$^+$.BT	**3093**	**58**	180	52
TransERT.BT	3640	53	176	52
TransE-SlidE$^+$.BT	**3395**	**54**	176	**53**

100 parameters will be used for each entity (complex vector space multiplies it by 2).

Following the same principles, only one negative sample is generated per each positive triple. In order to create such negative samples, we use probabilistic corruption techniques over positive samples. Uniform negative sampling (*unif* in Table II) sets the probability of corruption for head $(?, r, t)$ and tail $(h, r, ?)$ equally. The Bernoulli (*bern*) negative sampling [30] considers different probabilities for head $(?, r, t)$ and tail $(h, r, ?)$ corruptions to reduce number of false negative samples. Results reported in Table II for other models are taken from their original publication of research works. According to the results, TransE which is trained by MRL gets 89% and 47 on WN18 and FB15k respectively. TransE-RS which is trained by the limited-based score loss improves the results on both of the datasets. It gets 94% on WN18 and 72% FB15k. The results confirm that adding the term $[f_r(h,t) - \gamma_1]_+$ to the MRL significantly improves the performance of TransE model. TransE-SlidE$^+$ obtains accuracy of 95% on WN18 and 81% on FB15k which outperforms RotatE in terms of all metrics except Hits@10 in WN18. Therefore, the proposed loss function with expansion approach improves the accuracy of TransE. TransE trained by contraction approach gets better results comparing to the margin ranking loss.

According to Table III, TransE-SlidE$^+$ slightly outperformed TransE-SM on WN18RR and FB15k-237 considering Mean Rank and Hits@10. TransE-SM is very sensitive to γ_1 and γ_2, thus, the results changes dramatically with slight changes in γ_1 and γ_2. Therefore, the search space is very huge for TransE-SM whereas TransE-SlidE$^+$ only needs to search for γ. Moreover, for TransE-SlidE$^+$ we fixed the hyper-parameters including σ (except γ) to a value. We additionally show the proposed loss function is less sensitive to the hyperparameter σ (Figure 3b, Figure 3e and Table IV). On FB15k and FB15k-237, Hits@10 of RotatE gets only 75% and 43% respectively. However, TransE-SlidE$^+$ gets 81% and 46% on the same KGs. We additionally, investigate the effectiveness of SlidE$^+$ by training RotatE with its original loss and our proposed loss using expansion technique (RotatE-SlidE$^+$). The results show that RotatE-SlidE$^+$ outperforms the RotatE model. Illustrated in Figure 3, the performance of our model is affected by the hyperparameters (γ, σ and λ). Initially, the value of σ and λ are fixed to 1.0, then different values for γ are tested. From Figure 3a Figure 3d, we can see when γ increases from 0 to 5, the Hits@10 rate increases by 20%–50%. The performance of TransE-SlidE$^+$ stays unchanged when γ is between 15 to 20 for most of the datasets (WN18, WN18RR and FB15k-237). For FB15k, the best performance (in terms of Hits@10) is obtained around γ=30. In Figure 3b, Figure 3e, Figure 3c, Figure 3f, we set the optimal configuration of our model with fixed value of γ separately, and observe the performance where $\sigma \in \{0.1, 1, 2, ..., 10\}$ and $\lambda \in \{0, 0.5, 1, 2, ..., 10\}$. According to Figure 3b, Figure 3e, in most cases, the best performance for our model is obtained when σ is between 0 and 1. Mostly, with $\sigma = 1$ our model obtains satisfactory performance. These results approve the advantage of our with less sensitivity to hyper-parameter σ. It can also be confirmed by Table IV where in the most cases, the best performance is obtained with $\sigma = 1$, same applies to λ (see Figure 3c, Figure 3f). As a conclusion, once a proper value of γ is adjusted, the best setting for σ and λ can be obtained easily.

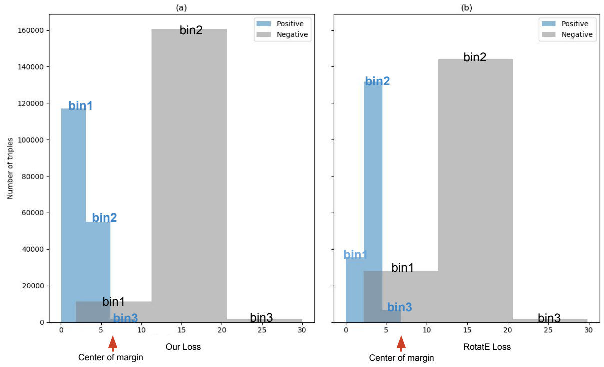


Fig. 4: **Histogram of the scores.** a) the RotatE model is trained using the SlidE$^+$, b) the RotatE model is trained using the RotatE loss. The models are trained on WN18RR. BT stands for experiments using boosting techniques.

Figure 4 illustrates the scores of positive and negative samples obtained by training the RotatE model using SlidE$^\pm$ and the RotatE loss. For each of the positive and negative samples, the scores are distributed into three bins. The center of the margin is set to six (in the X axis). Using the RotatE loss, the third bin of the positive samples and the first bin of the negative samples (which are closer to the margin) contain more samples than the corresponding bins of SlidE$^+$. These results approve that SlidE$^+$ relatively performs better in setting the margin between negative and positive samples.

V. Conclusion

We propose the SlidE$^\pm$ family of loss functions to improve the performance of embedding models in capturing knowledge from large knowledge graphs. SlidE$^\pm$ is designed and developed to tackle the problem of automatically obtaining a margin during the training process. In contrast to other approaches which are using manual settings for the upper and lower bound of positive and negative samples (to set the margin) within a large search space, SlidE$^{\pm}$ adapts the center of the margin. TransE and RotatE models are trained by SlidE$^{\pm}$ and evaluated considering Mean Rank and Hits@10 of the other loss functions. The results represent a significant improvement in accuracy with our proposed loss function. We additionally observed that RotatE-SlidE^{+} (RotatE model trained by SlidE loss function using expansion technique) outperforms the RotatE model trained by its original loss function. TransE-SlidE^{+} (TransE model trained by SlidE loss function using expansion technique) performs 95% on filter of WN18 whereas TransE trained by Margin Ranking Loss is reported to be 89% in Hits@10 and Limited-based Scoring Loss result is stated to have 94% of accuracy. On FB15k, the difference is high as TransE-SlidE^{+} reaches 81% while TransE on MRL is 47% and 72% is the reported accuracy for Limited-based Scoring. Furthermore, we used boosting techniques in order to transparently compare our results with setting of other models.

Acknowledgement: This work is supported by the EPSRC grant EP/M025268/1, the WWTF grant VRG18-013, the EC Horizon 2020 grant LAMBDA (GA no. 809965), the CLEOPATRA project (GA no. 812997), and the German national funded BmBF project MLwin.

References

[1] S. Adams. Surfing the hype cycle to infinity and beyond. *Research-Technology Management*, 62(3):45–51, 2019.

[2] Anonymous. Relation pattern encoded knowledge graph embedding by translating in complex space. anonymous preprint under review, 2018.

[3] S. A. Bini. Artificial intelligence, machine learning, deep learning, and cognitive computing: what do these terms mean and how will they impact health care? *The Journal of arthroplasty*, 33(8):2358–2361, 2018.

[4] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. AcM, 2008.

[5] A. Bordes, S. Chopra, and J. Weston. Question answering with subgraph embeddings. *arXiv preprint arXiv:1406.3676*, 2014.

[6] A. Bordes, X. Glorot, J. Weston, and Y. Bengio. Joint learning of words and meaning representations for open-text semantic parsing. In *Artificial Intelligence and Statistics*, pages 127–135, 2012.

[7] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795, 2013.

[8] A. Bordes, J. Weston, R. Collobert, and Y. Bengio. Learning structured embeddings of knowledge bases. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

[9] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[10] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka, and T. M. Mitchell. Toward an architecture for never-ending language learning. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[11] H. Chen, M. Sun, C. Tu, Y. Lin, and Z. Liu. Neural sentiment classification with user and product attention. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 1650–1659, 2016.

[12] S. He, K. Liu, Y. Zhang, L. Xu, and J. Zhao. Question answering over linked data using first-order logic. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1092–1103, 2014.

[13] K. Höffner, S. Walter, E. Marx, R. Usbeck, J. Lehmann, and A.-C. N. Ngomo. Survey on challenges of question answering in the semantic web. *Semantic Web Journal*, 2016.

[14] R. Jenatton, N. L. Roux, A. Bordes, and G. R. Obozinski. A latent factor model for highly multi-relational data. In *Advances in Neural Information Processing Systems*, pages 3167–3175, 2012.

[15] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao. Knowledge graph embedding via dynamic mapping matrix. In *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)*, volume 1, pages 687–696, 2015.

[16] T. Lacroix, N. Usunier, and G. Obozinski. Canonical tensor decomposition for knowledge base completion. *arXiv preprint arXiv:1806.07297*, 2018.

[17] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer. DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 6(2):167–195, 2015. Outstanding Paper Award (Best 2014 SWJ Paper).

[18] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

[19] Y. Lin, S. Shen, Z. Liu, H. Luan, and M. Sun. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 2124–2133, 2016.

[20] W. Liu, P. P. Pokharel, and J. C. Príncipe. Correntropy: Properties and applications in non-gaussian signal processing. *IEEE Transactions on Signal Processing*, 55(11):5286–5298, 2007.

[21] G. A. Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[22] M. Nayyeri, S. Vahdati, J. Lehmann, and H. S. Yazdi. Soft marginal transe for scholarly knowledge graph completion. *arXiv preprint arXiv:1904.12211*, 2019.

[23] M. Nickel, V. Tresp, and H.-P. Kriegel. Factorizing yago: scalable machine learning for linked data. In *Proceedings of the 21st international conference on World Wide Web*, pages 271–280. ACM, 2012.

[24] K. Panetta. trends emerge in the gartner hype cycle for emerging technologies, 2018. *Retrieved November*, 4:2018, 5.

[25] W. Shen, J. Wang, P. Luo, and M. Wang. Linking named entities in tweets with knowledge base via user interest modeling. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 68–76. ACM, 2013.

[26] Z. Sun, Z.-H. Deng, J.-Y. Nie, and J. Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*, 2019.

[27] S. Szumlanski and F. Gomez. Automatically acquiring a semantic network of related concepts. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 19–28. ACM, 2010.

[28] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*, pages 2071–2080, 2016.

[29] D. Vrandečić and M. Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

[30] Z. Wang, J. Zhang, J. Feng, and Z. Chen. Knowledge graph embedding by translating on hyperplanes. In *Twenty-Eighth AAAI conference on artificial intelligence*, 2014.

[31] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 353–362. ACM, 2016.

[32] X. Zhou, Q. Zhu, P. Liu, and L. Guo. Learning knowledge embeddings by combining limit-based scoring loss. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1009–1018. ACM, 2017.