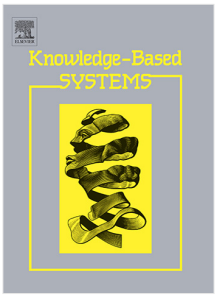


Knowledge graph embedding with self adaptive double-limited loss

Xiaoying Zou, Xizhao Wang, Si Cen, Guoquan Dai, Chao Liu

PII: S0950-7051(22)00657-8
DOI: https://doi.org/10.1016/j.knosys.2022.109310
Reference: KNOSYS 109310

To appear in: *Knowledge-Based Systems*

Received date : 19 January 2022
Revised date : 19 June 2022
Accepted date : 20 June 2022

Please cite this article as: X. Zou, X. Wang, S. Cen et al., Knowledge graph embedding with self adaptive double-limited loss, *Knowledge-Based Systems* (2022), doi: https://doi.org/10.1016/j.knosys.2022.109310.

Highlights

Knowledge Graph Embedding with Self Adaptive Double-Limited Loss

Xiaoying Zou,Xizhao Wang,Si Cen,Guoquan Dai,Chao Liu

- A new loss function, named ADL, for Knowledge graph embedding is proposed.
- Incorporating ADL into the currently popular embedding models, five extensions, i.e., TransE-ADL, TransH-ADL, TransD-ADL, TorusE-ADL, and ComplEx-ADL, are developed.
- The developed models have significantly improved the performance of link prediction in comparison with the baseline model.

Knowledge Graph Embedding with Self Adaptive Double-Limited Loss*

Xiaoying Zou[a], Xizhao Wang[a,b,*], Si Cen[a], Guoquan Dai[a] and Chao Liu[a]

[a]College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, 518060, China
[b]Guangdong Key Laboratory of Intelligent Information Processing, Shenzhen University, Shenzhen, 518060, China

ARTICLE INFO

Keywords:
Knowledge graph
Embedding
Loss function
Negative sampling
Hard pair

ABSTRACT

Many well-performing embedding models for knowledge graphs employ a negative sampling framework to complete the representation learning in which the loss function is a critical component in distinguishing between positive and negative triplets. One of the most recently proposed loss functions is the double-limited scoring loss, which sets fixed upper and lower bounds respectively for positive and negative triplets. We find that, for all positive and negative triplets, fixed upper and lower bounds are not appropriate since triplets that are difficult to be distinguished usually have changing bounds. In this paper, we propose a self adaptive double-limited loss (ADL) that dynamically adjusts the upper limit of positive triplet scores and the lower limit of negative triplet scores through evaluating the score proportion between positive and negative triplets. Furthermore, based upon ADL, we build several knowledge graph embedding models, including TransE-ADL, TransH-ADL, TransD-ADL, TorusE-ADL, and ComplEx-ADL, in which the gradient descent technique is used to train their parameters. Dynamically adjusted bounds lead to a reasonable partition of positive and negative triplets within embedding space, improving prediction accuracy significantly. The experimental results of link prediction confirm this improvement compared to the state-of-the-art baselines.

1. Introduction

Knowledge graphs(KGs), which can efficiently store and represent natural knowledge and facts, are among the essential knowledge representation mode in artificial intelligence and knowledge management for cross-industrial applications. In recent years, the emergence of large-scale knowledge bases, such as NELL[1], Freebase[2], Wordnet[3], DBpedia[4], Yago[5], have promoted the research of KGs for AI applications, such as question answering[6][7], recommendation systems[8][9] and knowledge management systems[10] [11].

KGs are essentially the semantic network of entities and edges whose the basic unit is a triplet in the form of (head entity, relation, tail entity), denoted (h, r, t). Unlike traditional symbolic representation, knowledge graph embedding (KGE) models entities and relations into vectorized representations that can be mathematically calculated for their semantic relationships, conducive to KG completion tasks, such as link prediction. In general, many KGE models, especially translation-based models [12] [13] [14] [15], employ a negative sampling loss framework to train entity vectors and relation vectors under the supervision of triplets.

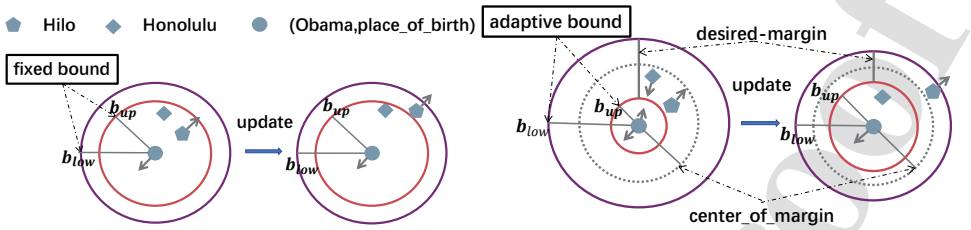
In the negative sampling loss framework, the scoring function $f_r(h, t)$ is used to measure the plausibility of the triplet (h, r, t), and the loss function is used to minimize the score of the correct triplet (h, r, t) and maximize the score of corresponding corrupted triplet (h', r, t') which are constructed by a negative sampling strategy. For example, TransE[13] is designed with a scoring function to measure the error of geometric translation, expecting a low score for positive triplets and a high score for the corresponding negative triplets. At the same time, the margin-based ranking loss is used to optimize the scores of positive and negative triplets to achieve this expectation.

Recently, many researchers have realized that designing reasonable contrast strategies for loss functions can effectively improve the performance of models. The margin-based ranking loss (MRL)[13] ensure a lower score for positive triplets than negative triplets by a margin, but does not guarantee that the score of positive triplets is sufficiently low. To avoid excessive scores for positive triplets, the limited-based loss (RSL)[16] adds an upper limit to the positive scores. The double limit scoring loss (SSL)[17] sets an upper bound to avoid high scores for positive triplets and a lower bound to further avoid low scores for negative triplets.

Most KGE methods model entity and relation vectors by comparing triplet pairs. Each pair is consisting of a positive triplet and its corresponding negative triplet. In studying these KGE methods, we find that, triplet pairs, which satisfy $f_r(h, t) > f_r(h', t')$and therefore are referred as to hard pairs, need to be particularly noticed by the model and then to be optimized specially. Although SSL can effectively avoid the flaws existing in MRL or RSL, it neither focuses on hard pairs and nor increases the strength of their differentiation, resulting in unclear boundary between their positive and negative entities. As shown in the left part of Fig.1(a), the

*This work was supported in part by the Postgraduate Innovation Development Fund Project of Shenzhen University (Grants 0000470814), in part by the National Natural Science Foundation of China (Grants 61976141, 61732011 and 62106148) and the Project funded by China Postdoctoral Science Foundation under Grant no. 2021M702259.

*Corresponding author at: College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, 518060, China
xiaoyingzou12@gmail.com (X. Zou); xizhaowang@ieee.org (X. Wang); szucensi@163.com (S. Cen); egbertdai@foxmail.com (G. Dai); zzulchao@163.com (C. Liu)
http://www.hebmlc.org/en/ (X. Wang)
ORCID(s): 0000-0001-6036-4728 (X. Wang)



(a) Optimize with fixed upper and lower bounds. (b) Optimize with adaptive upper and lower bounds

Figure 1: The illustration of optimization strategies for fixed boundary loss (SSL) and adaptive boundary loss (ADL).The positive triplet is $(Obama, place\_of\_birth, Honolulu)$ and the corresponding negative triplet is $(Obama, place\_of\_birth, Hilo)$.

positive triplet (Obama, place_of_birth, Honolulu) and the negative triplet (Obama, place_of_birth, Hilo) is a hard pair. However, since the score of (Obama, place_of_birth, Honolulu) is less than the upper bound and then the model cannot simultaneously optimize positive entities (Honolulu) and negative entities (Hilo), it finally results in unclear bounds between positive and negative entities, as shown on the right part of Fig.1(a). Specifically, this strategy is not conducive to identifying positive and negative entities when both entities have many same attributes. Therefore, we conjecture that the above-mentioned unclear boundary problem is caused by the fact that the fixed upper and lower bounds cannot be adapted to all positive-negative pairs and the optimization strategy is not reasonable enough.

Based on the idea that there should be a larger margin between indistinguishable positive and negative triplets, we use self-adaptive margins to separate each positive-negative pair to tackle this issue. Inspired by SSL, our self-adaptive margins are achieved by adjusting appropriate upper and lower bounds for each pair, which indicates our upper and lower bounds are self-adaptive. Specifically, our model automatically combines the distinguishing-difficulty which is calculated by the geographic difference ($f_r(h,t)$ versus $f_r(h',t')$) with the center of margin to generate the upper and lower bounds (b_{up} and b_{low}) for each positive-negative pair.We argue that the smaller the score of a negative triplet than the positive one, the harder to distinguish them. And so, a smaller b_{up} and a larger b_{low} will be helpful to separate positive-negative triplets. For example, as shown on the left side of Fig.1(b), at this point, positive triplet (Obama, place_of_birth, Honolulu) and negative triplet (Obama, place_of_birth, Hilo) is a hard pair, and the model automatically generates small b_{up} and large b_{low}, thus obtaining a larger desired margin that facilitates their correct classification. After one update, as shown on the right side of Fig.1(b), at this point, (Obama, place_of_birth, Honolulu) and (Obama, place_of_birth, Hilo) are a non-hard pair, and the model automatically increases the upper bounds and decreases the lower bounds to reduce the desired margin to prevent overfitting.

In this paper, we design a self-adaptive double limit loss to achieve that the upper and lower bounds are self-adaptive. Then, we apply the proposed self-adaptive double limit loss to TransE, TransH, TransD, TorusE and ComplEx, and propose several extended models, namely TransE-ADL, TransH-ADL, TransD-ADL, TorusE-ADL and ComplEx-ADL. In our experiments, we use WordNet and Freebase as standard benchmark datasets and perform link prediction to evaluate the extended models. In summary, ADL can be regarded as a comprehensive framework for translation models, and our contributions are three folds:

- A self adaptive double-limited loss framework for KGE models is presented, which adaptively adjusts the upper bound and lower bound;
- Incorporating ADL into the currently popular embedding models, we propose five extended models of TransE-ADL, TransH-ADL, TransD-ADL, TorusE-ADL, and ComplEx-ADL;
- Experiments on the Wordnet and Freebase datasets show that our proposal models have significantly improved link prediction tasks compared to the baseline model.

The rest of this paper is organized as follows. The second section gives some existing related works about KGE models. In the third section, we present and discuss our proposed loss function, ADL, and then introduce our proposed models, TransE-ADL, TransH-ADL, TransD-ADL, TorusE-ADL, and ComplEx-ADL. In the fourth section, we detail the experimental studies on our proposed models and discuss how the parameters affect the performance of our model. In the last section, we give a conclusion for our paper.

2. Related work

In recent years, much works have focused on learning excellent KGE by designing various scoring and loss functions. This section introduces the popular KGE models based on different scoring functions and then presents several works on the improvement of the loss function.

2.1. Knowledge graph embedding models

The scoring function which can discriminate triplets' credibility is an essential component in the KGE models. According to the definitions of different scoring functions, the KGE models can be briefly classified into translation-based and semantic matching models. The relevant embedding models are briefly described below.

Translation-based models interpret relations as geometric operations in the latent space and can be divided into three types[18]: Pure translational models[13], Translational models with additional embeddings[19] [20] and Roto-translational models[21][22][23]. TransE[13] is typically a pure translation model, which represents the relation as a translation from head entity to tail entity. However, the scoring approach of TransE is problematical for complex relational triplets, like one-to-many, many-to-one, and many-to-many relations. Translational models with additional embeddings learn more than one embedding for each KG element to solve the problem in TransE. TransH[14] introduces relation-specific hyperplanes with normal vectors to deal with the complex relational triplets, which projects the entity vector onto the hyperplanes before scoring the triplets. TransR/CTransR [24] considers that each entity can have many aspects and different relations focus on various aspects of the entity, thus mapping the entity from entity space to relations space before scoring. Based on TransR, TransD[25] assumes that the projection matrix is jointly determined by entities and relations and then decomposes the projection matrix into two vectors to reduce parameter numbers. In Roto-translational models, translation operations are replaced with rotation-like transformations to solve the regularization problem existing in TransE or model various relations patterns. For example, TorusE[22] defines the distance functions on a torus which is a compact Abelian Lie group; and RotatE[23] defines each relation as a rotation from a source entity to a target entity in complex vector space to model the four relations, i.e., symmetry, anti-symmetry, inversion, and composition.

Semantic matching-based models exploit similarity-based scoring functions, which measure the plausibility of facts by matching latent semantics of entities and relations embodied in their vector space representations [26]. RESCAL[27] represents each entity as a vector and each relation as a matrix and then uses a bilinear function to capture pairwise interactions between all components of head and tail entities. Distmult[28], as a variant of RESCAL, reduces the complexity of the model by setting the relations matrix as a diagonal matrix. ComplEx[29] uses complex-valued embeddings to handle the binary relations of symmetric and antisymmetric relations. ANALOGY[30] extends RESCAL to constrain the relations matrix to a normal matrix, thereby modeling the analogy properties of entities and relations. TuckER[31] scores the triplets by multiplying the core tensor obtained by Tucker decomposition with the head vector, the relation vector, and the tail vector. NTN[32] replaces the linear transformation layer in traditional neural networks with a bilinear tensor, linking head and tail vectors in different dimensions. R-GCN[33] considers that much of the missing information may be present in the graph encoded through the neighborhood structure and thus aggregates up the neighborhood information for each entity. ConvE[34] extracts the feature vectors of entities and relations by 2D convolution and then multiplies them with the vectors of the tail entity to score the triplets.

2.2. Loss Functions

The scoring of positive and negative triplets is constrained by the loss function. In the KGE models, defining margins to divide positive and negative triplets is one of the promising solutions in keeping a high performance for loss functions.

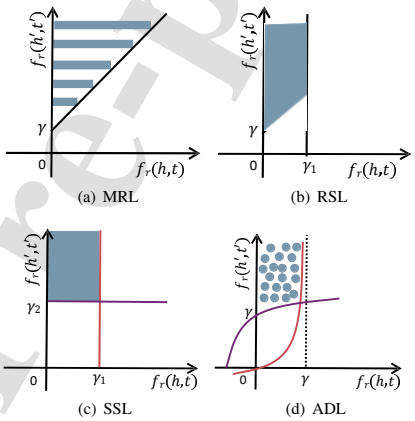


Figure 2: The division of positive and negative scores.

MRL [13] is the most commonly used loss function in translation-based models, and many works have confirmed its validity [13] [25] [24]. MRL favors lower values of the score for positive triplets than for negative triplets through setting a margin of γ between positive and negative samples. It is defined as follows:

$$L_{MR} = \sum_{(h,r,t)\in S} \sum_{(h',r,t')\in S'} [f_r(h,t) + \gamma - f_r(h',t')]_+ \quad (1)$$

where $[x]_+ = max(0, x)$. S is the set of positive triplets, while S' is the set of negative triplets, and $\gamma(> 0)$ is a margin. Minimizing L_{MR} allows positive and negative triplets to satisfy the inequality $f_r(h',t') - f_r(h,t) \geq \gamma$, the shaded part of Fig2(a). However, MRL has no limit for the maximum scores of positive triplets, which leads to excessive scores of positive triplets and overlapping between positive and negative scores.

In order to guarantee lower scores for positive triplets, RSL[16] is presented with an upper bound for golden triplets. Based on the RSL, the two translation-based models, TransE-RS and TransH-RS, significantly improve the performance by constraining the scores of positive triplets

to an upper limit. RSL is defined as follows:

$$L_{RS} = \sum_{\substack{(h,r,t)\in S \\ (h',r,t')\in S'}} [f_r(h,t)+\gamma-f_r(h',t')]_+ + \lambda[f_r(h,t)-\gamma_1]_+ \tag{2}$$

where $\lambda > 0$. $\gamma(> 0)$ is a margin, and γ_1 is the upper bound for the score of positive triplets. Fig2(b) shows the division between positive and negative scores, with the shaded areas of the RSL expected scores.

For negative triplets to score high independently, SSL[17] force the scores of positive triplets to stay before the upper bound γ_1 and the scores of negative triplets to stay after the lower bound γ_2, providing a more flexible and efficient optimization for KGE. SSL is defined as follow:

$$L_{SS} = \sum_{\substack{(h,r,t)\in S \\ (h',r,t')\in S'}} [f_r(h,t)-\gamma_1]_+ + \lambda[\gamma_2-f_r(h',t')]_+ \tag{3}$$

where $\gamma_1 > 0$, $\gamma_2 > 0$, $\lambda \geq 0$. λ is a parameter that balances the two limit-based scoring loss terms, and $\gamma_2 - \gamma_1 > 0$ represents the margin of positive and the corresponding negative triplets. The shaded part of Fig2(c) is the region of SSL expected scores.

TransESM[35] fixes the upper bound of positive triplets and uses a sliding mechanism to move false-negative triplets towards positive triplets. TransEAML[36] replaces the upper and lower bounds with the center of margin and then uses a slack variable to adjust the margin between positive and negative triplets automatically. ComplEx[29] defines the negative log-likelihood loss to optimize the parameters in the model. RotatE[23] uses a loss function that keeps the positive and negative triplets as far away from the fixed margin as possible. ConvE[34] applies the logistic sigmoid function to the score and binary cross-entropy losses for optimization.

3. Methodology

In this section, we first classify pairs containing positive and negative triplets and then analyze the characteristics of those pairs. Then, we propose a novel loss framework for KGE and five ADL-based models, TransE-ADL, TransH-ADL, TransD-ADL, TorusE-ADL, and ComplEx-ADL. Finally, we introduce the optimization and training process of our models.

3.1. Preliminaries

The SSL loss framework[17] enables more flexible and effective optimisation for positive and negative triplets, and has been successfully applied to TransE[13], TransH[14], TransD[25], ProjE[37] and ComplEx[29]. SSL takes two independent bounds to distinguish positive and negative triplets, ensuring that positive triplets are scored lower than negative triplets and that restrictions on negative triplets do not directly affect positive ones. The formal expression for SSL is shown in Eq.(3), where γ_1 is a fixed upper bound to restrict the score of positive triplets, $b_{up} = \gamma_1$, and γ_2 is a fixed lower bound to restrict the score of negative triplets, $b_{low} = \gamma_2$.

To better understand the positive and negative triplets in the model, we classify positive-negative pairs into three simple categories, namely, simple pairs, semi-hard pairs, and hard pairs. In simple pairs, positive and negative triplets are correctly distinguished and satisfy the constraints of the model, which are determined by $\{f_r(h,t) \leq b_{up}, f_r(h',t') \geq b_{low}\}$. In semi-hard pairs, positive and negative triplets can be correctly distinguished, but one of them still does not satisfy the constraints of the model. The triplets of semi-hard pairs satisfy the conditions $\{f_r(h,t) < f_r(h',t'), f_r(h,r) > b_{up}\}$ or $\{f_r(h,t) < f_r(h',t'), f_r(h',t') < b_{low}\}$. In hard pairs, positive triplets and negative triplets are not correctly distinguished, and triplets satisfy $\{f_r(h,t) \geq f_r(h',t')\}$. Based on this classification, it is clear that the model only needs to focus on the triplets in semi-hard and hard pairs during the optimization process, while the simple pairs are the target of optimisation.

Optimizing the parameters in hard pairs compared with semi-hard pairs is more effective way to improve the model performance. Then, we further divide hard pairs into three parts, namely (a) $\{f_r(h',t') \leq f_r(h,t) < b_{up}\}$, (b) $\{b_{low} < f_r(h',t') \leq f_r(h,t)\}$ and (c) $\{f_r(h',t') \leq f_r(h,t), f_r(h,t) > b_{up}, f_r(h',t') < b_{low}\}$. In SSL, it is not possible to optimise the parameters in a triplet when the score of a positive triplet is smaller than the minimum score or when the score of a negative triplet is larger than the maximum score. In other words, SSL cannot optimise the positive triplets in (a) and the negative triplets in (b). Obviously, such an optimisation strategy is not reasonable, and in particular, it is not conducive to the discrimination between positive and negative triplets with a high degree of similarity.

3.2. Self Adaptive Double-limited Loss

To better separate positive and negative triplets in hard pairs, the margin between two triplets should be adaptive. Here we suppose that, for hard pairs, the margin is larger, and for semi-hard or easy pairs, the margin can be smaller. To achieve such adaptive margin in a principled manner, we design a novel loss function ADL, where the margin between positive and negative triplets is adjusted by introducing the degree of triplet partition difficulty.

Before introducing the novel loss function, we first describe how to measure the degree of distinguishability between positive and negative triplets. In order to assign lower scores for positive triplets and higher scores for negative triplets respectively, we design a discriminator $\mathcal{M}$ for triplet pairs to estimate difficulty of distinguishing between positive and negative triplets in a pair. For each positive-negative pair (positive triplet i and negative triplet j), $\mathcal{M}$ uses their embeddings as inputs and generates their discrimination values m^r_{ij} as follows:

$$m^r_{ij} = \mathcal{M}_r(i,j) = \frac{f_r(h_i,t_i)}{k + f_r(h_j,t_j)} \tag{4}$$

where k(>0) is a super-parameter with very small value and $f_r(h,t)$ is the scoring function to measure the possibility of triplets being positive. When $0 < m_{ij}^r < 1$, positive and negative triplets in pairs are correctly distinguishable, and their distinguishability decreases as the value of m_{ij}^r increases. When $1 \leq m_{ij}^r$, positive and negative triplets in pairs are indistinguishable, and their indistinguishability increases as the value of m_{ij}^r increases.

By introducing the discriminator m_{ij}^r into the ADL, we obtain an upper bound b_{up}^{ij} for the score of positive triplet and lower bound b_{low}^{ij} for the score of negative triplet as follow:

$$\begin{aligned} b_{up}^{ij} &:= b_{up}(i,j) = \gamma - \lambda m_{ij}^r \\ b_{low}^{ij} &:= b_{low}(i,j) = \gamma + \lambda m_{ij}^r \end{aligned} \tag{5}$$

where γ is the center of margin and λ is a scale factor that determines the adjustment range of the upper and lower bounds. In result, the margin between positive and negative triplets is $2\lambda m_{ij}^r$ which is dynamically adaptive with i and j.

Finally, the loss function of ADL is defined as

$$L_{ADL} = \sum_{\substack{(h_i,r,t_i)\in S \\ (h_j,r,t_j)\in S'}} [f_r(h_i,t_i) - b_{up}^{ij}]_+ + [b_{low}^{ij} - f_r(h_j,t_j)]_+ \tag{6}$$

We find that the value of m_{ij}^r for hard pairs is usually greater than or equal to 1, while the value of m_{ij}^r for semi-hard pairs is close to 1 and the scoring ratio obtained for simple pairs is even smaller. Thus, the discriminator m_{ij}^r can be used as a driver for upper and lower bound adjustments. Specifically, we take $b_{up}^{ij} (\geq 0)$ as the upper bound for the scoring of positive triplets, and b_{low}^{ij} as the lower bound for the scoring of negative triplets. Then the positive scores are force to satisfy the inequality $f_r(h_i,t_i) \leq b_{up}^{ij}$, i.e. $f_r(h_i,t_i) \leq \frac{\gamma(k+f_r(h_j,t_j))}{\lambda(k+f_r(h_j,t_j))+1}$, and the margin between $f_r(h_i,t_i)$ and $f_r(h_j,t_j)$ is at least $2\lambda m_{ij}^r$. The scoring curve for ADL is shown in Fig.2(d).

3.3. Models

In this subsection, we present several KGE models, including TransE-ADL, TransH-ADL and TransD-ADL, TorusE-ADL, and ComplEx-ADL based on the proposed self adaptive double-limited loss.

3.3.1. TransE-ADL

TransE[13] takes relations as translation operations from head entities to tail entities and encodes them into the same feature space. For each triplet (h,r,t), TransE uses a scoring function $f_r(h,t)$ to evaluate the plausibility of the triplet, defining a scoring function as

$$f_r(h,t) = \| h + r - t \|_2^2 \tag{7}$$

with restrictions $\| e \|_2 = 1$ and $\| r \|_2 = 1$. Based on TransE, we propose a novel model, TransE-ADL, which has the same scoring function as TransE, expecting a lower score for positive triples and a higher score for negative triples. Unlike TransE, TransE-ADL uses the proposed loss function, L_{ADL}, to optimize and train the parameters in the model.

3.3.2. TransH-ADL

TransE is suitable for 1-to-1 relations but has issues for 1-to-N, N-to-1, and N-to-N relations, so TransH[14] is proposed to solve these issues by introducing a specific relational hyperplane. Given a triplet (h,r,t), TransH models the projection vector of the head entity and tail entity on the relational hyperplane via the normal vectors w_r, i.e. $h_\perp = hw^T hw$, $t_\perp = tw^T tw$. Then, the scoring function of TransH is defined as

$$f_r(h,t) = \| h_\perp + r - t_\perp \|_2^2 \tag{8}$$

where $\| e \|_2 \leq 1$, $| w_r^\top d_r | / \| d_r \|_2 \leq \epsilon$, $\| w_r \|_2 = 1$. Based on TransH, we propose an extended model, TransH-ADL, which employs the scoring function of TransH and uses our proposed L_{ADL} to optimize the parameters in the model.

3.3.3. TransD-ADL

Considering multiple types of entities and relations, TransD[25] model them into different feature spaces. For each named symbol object (entities and relations), a feature vector and a projection vector need to be constructed by TransD. For example, given a triplet (h,r,t), its vectors are $\mathbf{h}$, $\mathbf{h_p}$, $\mathbf{r}$, $\mathbf{r_p}$, $\mathbf{t}$ and $\mathbf{t_p}$. The mapping matrices of h and t are $\mathbf{M_{rh}}$, $\mathbf{M_{rt}}$, and specifically $M_{rh} = r_p h_p^\top + I^{n\times m}$ and $M_{rt} = r_p t_p^\top + I^{n\times m}$. Then, the scoring function of TransD is

$$f_r(h,t) = \| M_{rh} h + r - M_{rt} t \|_2^2 \tag{9}$$

Where $\| h \|_2 \leq 1$, $\| t \|_2 \leq 1$, $\| M_{rh} h \|_2 \leq 1$, $\| M_{rt} t \|_2 \leq 1$. Similarly, we propose a new extended model, TransD-ADL, which uses the same scoring function as TransD, i.e. eq.(9), and employs our proposed L_{ADL} as the loss function.

3.3.4. TorusE-ADL

TransE uses regularization, resulting in forcing entities to be embedded on a sphere in the embedding vector space, which cannot cover a number of cases with $h+r=t$, and then leads to a negative impact on accuracy of link prediction. To avoid regularization effect, TorusE[22] chooses a torus, one of the compact Lie groups, as the embedding space, and allows the model to learn embeddings that follow the TransE principle more accurately.

For a triple (h,r,t), TorusE follows the same principle of $h+r=t$, but the embedding space is changed from a vector space to a torus. TorusE defines the distance function in three ways:

- $d_{L1}([h]+[r],[t]) = min_{(x',y')\in[h+r]\times[t]} ||x'-y'||_1$.
- $d_{L2}^2([h]+[r],[t]) = min_{(x',y')\in[h+r]\times[t]} ||x'-y'||_2$.
- $d_{eL_2}([h]+[r],[t]) = ||g([h+r]) - g([t])||_2$

with three scoring functions:

- $f_{L_1}(h,r,t) = 2d_{L1}([h]+[r],[t])$.
- $f_{L_2}(h,r,t) = 4d^2_{L2}([h]+[r],[t])$.
- $f_{eL_2}(h,r,t) = d_{eL_2}([h]+[r],[t])/4$

respectively.

Then, the translation principle is rewritten as $[h]+[r] = [t]$. Noting that TorusE uses MRL as the loss function, our proposed TorusE-ADL uses the same scoring function as TorusE but different loss function which is just our proposed ADL.

3.3.5. ComplEx-ADL

ComplEx[29] is a bilinear model that uses complex-valued embeddings to handle many types of binary relations, including symmetric and antisymmetric relations. Given a set of observable triplets $(h,r,t) \in N_t$, we can obtain a partially observable adjacency matrix $\{Y_{(h,r,t)\in N_t} \in \{-1,1\}$, which is used to distinguish between positive and negative triplets. The goal of ComplEx is to find the probabilities of entries $Y_{(h,r,t)\notin N_t}$ being negative or positive for the unobservable triplets.

The log-odd of probability with that a triplet (h,r,t) holds well can be expressed as $P(Y_{(h,r,t)} = 1) = \sigma(\phi(h,r,t;\Theta))$ where ϕ is a scoring function and Θ denotes the parameters of ComplEx. Specifically, the entity (head or tail) complex vector is $e_h \in \mathbb{C}^K$ or $e_t \in \mathbb{C}^K$ and the relation complex vector is denoted as $w_r \in \mathbb{C}^K$. The scoring function of ComplEx is specifically computed as

$$\begin{aligned}\phi(h,r,t;\Theta) &= Re(< w_r, e_h, \bar{e_t} >)\\ &= Re(\Sigma_{k=1}^K w_{rk} e_{hk} \bar{e_{tk}})\\ &=< Re(w_r), Re(e_h), Re(e_t) >\\ &+ < Re(w_r), Im(e_h), Im(e_t) >\\ &+ < Im(w_r), Re(e_h), Im(e_t) >\\ &- < Im(w_r), Im(e_h), Re(e_t) >\end{aligned} \tag{10}$$

where $Re(v)$ and $Im(v)$ are corresponding to the real and imaginary parts of the vector v. The w_r of the antisymmetric relation is purely imaginary, while the w_r of the symmetric relation is purely real.

Our ComplEx-ADL scoring function is referenced from ComplEx-SS[17], which is defined as

$$f_r(h,r,t) = -log(\sigma(\phi(h,r,t;\Theta)) \tag{11}$$

The loss fuction of ComplEx-ADL is the same as ADL.

3.4. Optimization and Training

In the optimization phase, the self adaptive double-limited loss can be minimized and the model parameters updated through gradient descent[**?**]. For each positive triplet, at least one negative triplet is constructed by a negative sampling strategy[14] to create one or more positive-negative pairs. Given a mini-batch of triplets $\{(h_i,r_i,t_i)\} \in N_B$ from training set, first sample a negative triplet $\{(h'_i,r_i,t'_i)\}$ for each triplet $\{(h_i,r_i,t_i)\}$ to generate positive-negative pair, i.e. $\{(h_i,r_i,t_i),(h'_i,r_i,t'_i)\}$, then use stochastic gradient descent to optimize the L_{ADL} loss, as follows:

$$\begin{aligned}\nabla L_{ADL} &= \sum_{i=1}^{N_B} \nabla L_{ADL}(i)\\ &= \sum_{i=1}^{N_B} \nabla L_{pos}(i) + \nabla L_{neg}(i)\\ &= \sum_{i=1}^{N_B} \nabla[f_{r_i}(h_i,t_i) - b^{ij}_{up}]_+ + \nabla[b^{ij}_{low} - f_{r_i}(h'_i,t'_i)]_+\end{aligned} \tag{12}$$

To simplify the expression of the formula, we denote the i-th positive-negative pair by the symbol i and make $f_{rpos} = f_r(h,t)$, $f_{rneg} = f_r(h',t')$. Specifically, if the score of positive triplet is greater than its upper bound i.e.$L_{pos}(i) > 0$, then its gradient can be calculated by $\nabla L_{pos}(i) = \nabla f_{rpos}(i) + \nabla \frac{\lambda f_{rpos}(i)}{k+f_{rneg}(i)}$, else the gradient is 0 i.e.$\nabla L_{pos}(i) = 0$. Similarly, if the score of negative triplet is less than its lower bound i.e. $L_{neg}(i) > 0$, then its gradient can be calculated by $\nabla L_{neg}(i) = \nabla \frac{\lambda f_{rpos}(i)}{k+f_{rneg}(i)} - \nabla f_{rneg}(i)$, else the gradient is 0 i.e.$\nabla L_{neg}(i) = 0$. Although both $L_{pos}(i)$ and $L_{neg}(i)$ can produce gradients on (h_i,r_i,t_i) and $(h_i^{'},r_i,t_i^{'})$, their attention is different. For $L_{pos}(i) > 0$, the gradient of the positive triplets is $(1 + \frac{\lambda}{k+f_{rneg}})\nabla f_{rpos}$ which must be much larger than the gradient of the negative triplets $\lambda f_{rpos} \nabla \frac{1}{k+f_{rneg}}$. For $L_{neg}(i) > 0$, the gradient of negative triplets is $\lambda f_{rpos} \nabla \frac{1}{k+f_{rneg}} - \nabla f_{rneg}$ which must larger than the gradient of the positive triplets $\frac{\lambda}{k+f_{rneg}} \nabla f_{rpos}$. For a pair of positive and negative triplets, the model can optimize the positive one while appropriately adjusting the negative one, so that ADL can divide the positive and negative triplets more reasonably.

In the training stage, we initiate the entities and relations vectors with the random procedure [38] for TransE-ADL, TransH-ADL, TorusE-ADL, and ComplEx-ADL. For TransD-ADL, we initiate the entity and relation vectors with the results of TransE, and initiate all the transfer matrices with the identity matrices [25]. At each major iteration of the algorithm, the vectors of entities and relations are first normalized, then sampling a batch of positive triplets from the training set and generating a negative triplet for each positive triplet. Finally, there is the training and optimization of the parameters of the model, as detailed in Algorithm 1.

In addition, comparing our presented models and the related models regarding the complexity of parameters and times, we find that our models do not increase parameters and training time during the training phase, as shown in Table1. Respectively, m,n represent the dimensions of the entity and the relation vector in the embedding space; and the number of entities, relations and triplets are N_e, N_r and N_t, respectively.

Algorithm 1 KG embedding models with ADL

Input: Training set $S = \{(h, r, t)|h, t \in E, r \in R\}$, entities and rel. sets E and R, center of margin γ, parameter λ, negative triplets set S', batch size b.

Output: Entity and relation embedding **E** and **R**.

$E \leftarrow$ initialization(N_e, m)
$R \leftarrow$ initialization(N_r, n)
repeat
  $P = sample\_batch(S, b)$, $S' = \emptyset$
  for each (h, r, t)in positive sample set S **do**
    (h', r, t')=generate_negative((h, r, t))
    $S' = S' \cup (h', r, t')$
  end for
  $P = P \cup S'$
  for each$\{(h, r, t), (h', r, t')\}$ in P **do**
    Update embeddings w.r.t
    $\nabla[f_r(h,t) - \gamma + \frac{\lambda f_r(h,t)}{k+f_r(h',t')}]_+ + \nabla[\gamma + \frac{\lambda f_r(h,t)}{k+f_r(h',t')} - f_r(h', t')]_+$
  end for
until End
return **E**,**R**

Table 1
The complexity of models

Model	#Parameter	#Time Complexity
TransE	$O(N_e m + N_r n)$	$O(N_t)$
TransE-RS	$O(N_e m + N_r n)$	$O(N_t)$
TransE-SS	$O(N_e m + N_r n)$	$O(N_t)$
TransE-ADL	$O(N_e m + N_r n)$	$O(N_t)$
TransH	$O(N_e m + 2N_r n)$	$O(2mN_t)$
TransH-RS	$O(N_e m + 2N_r n)$	$O(2mN_t)$
TransH-SS	$O(N_e m + 2N_r n)$	$O(2mN_t)$
TransH-ADL	$O(N_e m + 2N_r n)$	$O(2mN_t)$
TransD	$O(2N_e m + N_r n)$	$O(2nN_t)$
TransD-SS	$O(2N_e m + N_r n)$	$O(2nN_t)$
TransD-ADL	$O(2N_e m + N_r n)$	$O(2nN_t)$
TorusE	$O(N_e m + N_r n)$	$O(N_t)$
TorusE-ADL	$O(N_e m + N_r n)$	$O(N_t)$
ComplEx	$O(N_e m + N_r n)$	$O(mN_t)$
ComplEx-SS	$O(N_e m + N_r n)$	$O(mN_t)$
ComplEx-ADL	$O(N_e m + N_r n)$	$O(mN_t)$

4. Experiment Analysis

This section introduces two popular knowledge bases, namely Wordnet and Freebase, and then conducts link prediction experiments on four datasets, namely WN18 and FB15K, WN18RR, and FB15K237, respectively. Furthermore, we compare our proposed models with the SSL-based models in both the training and testing phases. Finally, we analyze the effect of two hyperparameters on our models and analyze the defects of our proposed ADL loss.

4.1. Datasets

WordNet [3] and Freebase [2] are two popular KGs, where WordNet is a large database of English words, and Freebase is a large collaborative knowledge base consisting of data from community members. WordNet is a linguistic ontology repository as well as a semantic lexicon with a wide range of applications in natural language processing research. WN18 and WN18RR are both subsets of WordNet, with WN18RR removing the reverse edge from WN18. By storing data as a graph, Freebase can quickly traverse any connection between topics and easily add new schemas without changing the data structure. Similarly, FB15K and FB15K237 are both subsets of Freebase, where FB15K237 removes the reverse edge from FB15K. These four datasets are described in detail in table 2.

Table 2
Experimental Datasets

Dataset	#Rel	#Ent	#Train	#Valid	#Test
WN18	18	40,943	141,442	5,000	5,000
WN18RR	11	40,943	86,835	3,034	3,134
FB15K	1,345	14,951	483,142	50,000	59,071
FB15K-237	237	14,541	272,115	17,535	20,466

4.2. Link prediction

Link prediction[39] is to complete the golden triplets of missing head entity or tail entity. For a triple $(?, r, t)$ (or $(h, r, ?)$), the set of candidate triplets is constructed by replacing each entity in KG to the missing part of the triplet in turn. The scores of each candidate triplet in the set are then calculated and ranked from low to high. We expect the correct candidate triplets to receive the highest ranking. To evaluate the performance of the model, we use three evaluation metrics: Mean Rank(MR), Mean Reciprocal Rank(MRR), and the proportion of top-k rank (Hits@k) of correct candidate triplets. In the link prediction task, a satisfactory embedding model expects a lower MR and a higher MRR and Hit@k. In the strategy of replacing head or tail entities, 'unif' and 'bern' indicate the use of the equal probability and different probabilities following [14], respectively. The settings 'raw' and 'filt' indicate raw (possibly flawed) triplets and filtered triplets, respectively.

4.2.1. Results on WN18 and FB15K

To evaluate the effectiveness of our model, we conduct experiments of link prediction on both the WN18 and FB15K datasets. For TransE-ADL, TransH-ADL and TransD-ADL, we record two metrics (MR and Hits@10) that can express the experimental performance, while for TorusE-ADL, we record four metrics, MRR, Hits@1, Hits@3 and Hits@10. For experiments with our models, we select the learning rate α among {0.0001, 0.0003, 0.0005, 0.001, 0.01}, the batch size among {120, 256, 500, 960, 1200, 2400}, and the embedding dimension dim among {100, 150, 200, 256, 500}. For TransE-ADL, TransH-ADL and TransD-ADL, we select the scale factor λ in {1.0, 2.0, 3.0, 4.0, 5.0} and the center of margin γ from 1.0 to 15.0 (interval of 1.0). For TorusE-ADL, we select γ in {1000, 2000, 2500, 2600, 2700, 2800, 2900, 3000} and λ in {100, 150, 200, 500, 1100, 1100, 1200, 2000}, and refer to TorusE's experiment

Table 3
The results of Link prediction on WN18 and FB15K

Models	WN18				FB15K			
	Mean Rank		Hits@10(%)		Mean Rank		Hits@10(%)	
	raw	filt	raw	filt	raw	filt	raw	filt
SE[40]	1011	985	68.5	80.5	273	162	28.8	39.8
RESCAL[41]	1180	1163	37.2	52.8	828	683	28.4	44.1
TransR(unif)[24]	232	219	78.3	91.7	226	78	43.8	65.5
TransR(bern)[24]	238	225	79.8	92.0	198	77	48.2	68.7
TransSpare(unif)[15]	233	221	79.6	93.4	216	66	50.3	78.4
TransSpare(bern)[15]	223	211	80.1	93.2	190	82	53.7	79.9
DistMult[42]	987	902	79.2	93.6	224	97	51.8	82.4
HolE[43]	387	361	80.4	94.9	209	75	54.9	73.9
TransE(unif)[13]	263	251	75.4	89.2	243	125	34.9	47.1
TransE(bern)[13]	291	282	81.4	94.6	198	103	49.8	65.8
TransE-RS(unif)[16]	362	348	80.3	93.7	161	62	53.1	72.3
TransE-RS(bern)[16]	385	371	80.4	93.7	161	63	53.2	72.1
TransE-SS(unif)[17]	285	279	83.1	94.4	170	**39**	54.3	78.7
TransE-SS(bern)[17]	276	263	**83.6**	95.0	155	54	55.8	76.5
TransE-ADL(unif)(OUR)	159	147	82.4	**95.3**	189	41	55.1	**85.5**
TransE-ADL(bern)(OUR)	**154**	**143**	82.6	**95.3**	**152**	46	**56.0**	82.1
TransH(unif)[14]	318	303	75.4	86.7	311	84	42.5	58.5
TransH(bern)[14]	401	388	73.0	82.3	212	87	45.7	64.4
TransH-RS(unif)[16]	401	389	81.2	94.7	163	64	53.4	72.6
TransH-RS(bern)[16]	371	357	80.3	94.5	178	77	53.6	75.0
TransH-SS(unif)[17]	182	170	81.8	95.1	166	54	55.3	82.5
TransH-SS(bern)[17]	184	173	82.1	95.1	177	61	54.6	83.5
TransH-ADL(unif)(OUR)	190	178	82.5	**95.2**	197	**45**	54.7	**85.9**
TransH-ADL(bern)(OUR)	**179**	**167**	**82.7**	**95.2**	**155**	**45**	**55.6**	83.9
TransD(unif)[25]	242	229	79.2	92.5	211	67	49.4	74.2
TransD(bern)[25]	224	212	79.6	92.2	194	91	53.4	77.3
TransD-SS(unif)[17]	267	250	83.0	95.0	201	70	53.8	82.0
TransD-SS(bern)[17]	248	237	**83.1**	**95.3**	176	69	55.3	83.9
TransD-ADL(unif)(OUR)	**223**	**211**	82.7	95.2	202	**49**	54.9	**85.9**
TransD-ADL(bern)(OUR)	229	217	**83.1**	**95.3**	**168**	59	**55.7**	84.2

Table 4
The results of Link prediction on TorusE-based model

Models	WN18				FB15k			
	MRR(%)	Hits@1(%)	Hits@3(%)	Hits@10(%)	MRR(%)	Hits@1(%)	Hits@3(%)	Hits@10(%)
TorusE[22]	94.7	94.3	95.0	95.4	73.3	**67.4**	77.1	83.2
TorusE-ADL(OUR)	**94.8**	**94.3**	**95.0**	**95.6**	**74.7**	66.1	**81.7**	**86.8**

by setting the batch size to 10000 and the dim to 100. The details of the optimal parameter configuration are shown in the table 5.

Table3 shows the experimental results for the three models TransE-ADL, TransH-ADL, and TransD-ADL at WN18 and FB15K. We compare models with the same scoring function and identify the best experimental results with bold black. The experimental results show that the ADL-based models have a different degree of improvement in Mean Rank and Hits@10 compared to the other loss-based models. Specifically, for Hits@10, TransE-ADL and TransH-ADL (bern,filt) improve by 0.1% and 0.1% on WN18, and by 6.8% and 2.4% respectively on FB15K. TransD-ADL also improves the performance of link prediction compared to TransD-SSL, especially on FB15K with Hits@10 by 3.9% and 0.3% respectively. The link prediction results of TorusE-ADL in WN18 and FB15K are shown in Table4. It can be seen from Table4 that, with respect to indexes MRR and Hits@10, our TorusE-ADL outperformed. This indicates that, with the KGE learned by our proposed self adaptive double-limited loss, our models have the strongest generalization ability under the more challenging setting.

4.2.2. Results on WN18RR and FB15K-237

The datasets WN18 and FB15K have a large number of inverse triplets of the form (h, r, t) and (h, r^{-1}, t), where r^{-1} is the inverse relation of r. In the link prediction task, the embedding model is more biased toward learning the

Table 5
Setting of experimental optimal parameters

Dataset	models	α	γ	k	λ	Batch size	dim	epoch
WN18	TransE-ADL	0.01	7	0.01	5	1200	500	3000
	TransH-ADL	0.01	7	0.01	4	120	256	3000
	TransD-ADL	0.001	6	0.01	4	1200	100	3000
	TorusE-ADL	0.0001	3000	0.01	1100	100	10000	2000
FB15K	TransE-ADL	0.001	7	0.01	1	500	256	3000
	TransH-ADL	0.001	8	0.01	1	500	256	3000
	TransD-ADL	0.001	8	0.01	1	120	256	3000
	TorusE-ADL	0.0005	2800	0.01	150	100	10000	3000
WN18RR	TransE-ADL	0.0001	7	0.01	7	100	50	3000
	TransH-ADL	0.0005	6	0.01	4	120	100	3000
	TransD-ADL	0.0001	7	0.01	7	1200	50	3000
	ComplEx-ADL	0.0001	0.6	0.01	0.8	2400	200	2000
FB15K-237	TransE-ADL	0.0003	6	0.01	2	500	256	3000
	TransH-ADL	0.0001	5	0.01	1	1200	50	3000
	TransD-ADL	0.0001	8	0.01	2	1200	256	3000
	ComplEx-ADL	0.0001	0.6	0.01	0.95	2400	200	2000

Table 6
The results of Link prediction on WN18RR and FB15K-237

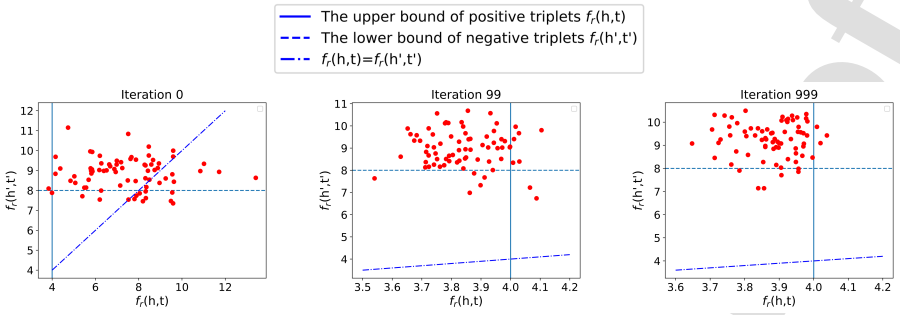
	WN18RR				FB15K-237			
Models	MRR(%)	Hits@1(%)	Hits@3(%)	Hits@10(%)	MRR(%)	Hits@1(%)	Hits@3(%)	Hits@10(%)
TransE[13]	20.7	2.2	36.1	47.8	27.9	19.3	30.5	44.9
TransE-RS[16]	20.8	2.3	36.3	47.8	28.2	19.4	31.2	46.1
TransE-SS[17]	20.9	2.5	**37.1**	47.9	28.4	**19.6**	31.7	47.0
TransE-ADL(OUR)	**22.7**	**5.6**	35.4	**50.8**	**28.8**	19.3	**32.2**	**47.4**
TransH[14]	19.8	0.7	36.3	46.3	26.7	17.7	29.9	44.5
TransH-RS[16]	18.1	0.9	36.9	47.6	27.3	17.6	30.6	46.4
TransH-SS[17]	20.1	1.0	37.3	47.8	28.5	17.8	31.2	46.7
TransH-ADL(OUR)	**20.6**	**1.1**	**37.6**	**49.8**	**28.7**	**19.3**	**32.1**	**47.4**
TransD[25]	20.3	4.2	35.5	46.0	30.5	20.5	31.7	47.7
TransD-RS[16]	21.0	3.7	35.6	47.1	31.8	23.1	35.5	50.3
TransD-SS[17]	**23.6**	4.3	35.8	49.6	**32.7**	**23.2**	**35.6**	**51.0**
TransD-ADL(OUR)	22.2	**4.8**	**36.1**	**49.6**	30.6	21.1	34.0	49.8
ComplEx[29]	40.1	36.2	42.5	47.1	24	15.2	26.4	42.3
ComplEx-SS[17]	41.3	37.8	44.5	50.6	24.7	15.7	27.3	43.4
ComplEx-ADL(OUR)	**46.2**	**42.7**	**47.8**	**52.5**	**27.3**	**19.0**	**29.9**	**43.5**

inverse relations[44]. To eliminate the effect of the inverse relation in WN18 and FB15K, [34], and [45] constructed new datasets WN18RR and FB15k-237 by retaining one of the inverse relations. Thus, we further evaluate the link predictions of our proposed models on WN18RR and FB15K-237.

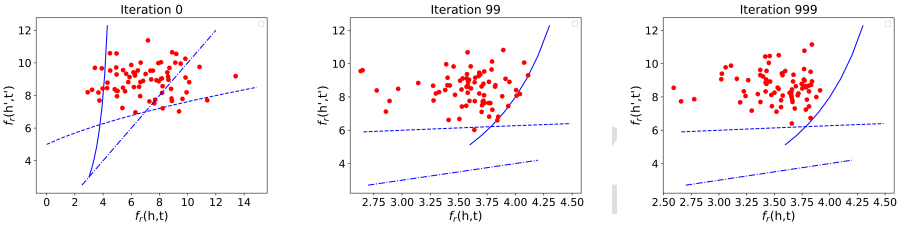
In this part, we compare our proposed TransE-ADL, TransH-ADL, TransD-ADL, and ComplEx-ADL with other loss-based models, and then evaluate our models with five evaluation metrics: MR, MRR, Hits@1, Hits@3, and Hits@10. For the parameter settings, we search the same parameter ranges on TransE-ADL, TransH-ADL, and TransD-ADL as in the experiments conducted on WN18 and FB15K. For ComplEx-ADL, we select the learning rate α in {0.0001, 0.0005, 0.001, 0.005}, set dim to 200 and the batch size to 2400. Referring to ComplEx-SSL, we use $-log()$ for the center of margin γ and the scale factor λ, respectively, i.e. $\gamma = -log(\gamma)$ and $\lambda = -log(\lambda)$, and select γ from 0.1 to 0.9 (with an interval of 0.1) and λ in {0.5, 0.6, 0.7, 0.8, 0.9, 0.95}The optimal parameter settings of our proposed model for the link prediction are recorded in Table5.

Table 7
The number of $f_r(h',t') - f_r(h,t) < 0$ in the test set

Model	TransE-SS	TransE-ADL	TransH-SS	TransH-ADL	TransD-SS	TransD-ADL
num	410	342	413	388	433	381



(a) TransE-SS



(b) TransE-ADL

Figure 3: Score distribution of pairs $\{(h,r,t),(h',r,t')\}$ in different training stages (on WN18)

Table6 shows the Link prediction results of our model with other models in WN18RR and FB15K-237. Similarly, we have marked the experimental results of the best model in the comparison model. From the table, we can find that the experimental performance of our models in WN18RR and FB15K-237 has also improved. Our proposed TransE-ADL and TransH-ADL improve the MRR metrics over the other loss-based models by 1.8% and 0.5% in WN18RR, and by 0.4% and 0.2% in FB15K237. Our proposed ComplEx-ADL obtains a significant performance improvement over other ComplEx-based models on both datasets.

4.3. Discussion

We analyze the scores of the triplets from two perspectives. We first observe the change in scores for the L_{ADL}-based and L_{SS}-based models in the training phase and then analyze the distribution of scores for positive and negative triplets in the testing phase.

4.3.1. Distribution of scores in the training phase

Fig.2 expresses the regions where we expect the pairs of positive triplets and corresponding negative triplets to be scored. However, the exact distribution of pairs during training is not clear. Therefore, we first record the distribution of positive-negative scores during training and analyze the differences in the scoring regions between TransE-ADL and TransE-SS.

To reduce the influence of other factors on the observed experiments, we recorded the positive-negative scores of TransE-ADL and TransE-SS during the training of the WN18 dataset and set the batch size, dim, and learning rate of both models to 75, 50 and 0.01. In addition, the other parameters were set to $\gamma_1 = 4$, $\gamma_2 = 8$, $\lambda = 1$ for TransE-SS ,and $\gamma = 5$, $k = 0.01$, $\lambda = 2$ for TransE-ADL. During training, the distribution of triplet scores for TransE-ADL is shown in the left column of, while that for TransE-SS is shown in the right column. During training, as shown in Figure 3, the distribution of scores for TransE-SS is shown above, while the distribution of scores for the triplet of TransE-ADL is shown below. In preliminaries, we classify positive and negative triplets into simple pairs ($\{f_r(h,t) \le b_{up}, f_r(h',t') \ge b_{low}\}$), semi-hard pairs ($\{f_r(h,t) < f_r(h',t'), f_r(h,r) > b_{up}\}$ or $\{f_r(h,t) < f_r(h',t'), f_r(h',t') < b_{low}\}$) and hard pairs($\{f_r(h,t) \ge f_r(h',t')\}$). As can be seen from Figure 3, in the initial state (when iteration number=0) all three types of triplet pairs exist simultaneously. As training progresses, the number of hard and semi-hard pairs decreases and the number of simple pairs increases, e.g. (when iteration number=99). When the loss function converges (iteration number=999), the number of hard pairs is zero and the number of semi-hard pairs in TransE-ADL is almost zero, while some semi-hard pairs still exist in TransE-SS. Therefore, we can believe that ADL fits the scores of positive and negative triplets better than SSL, and that the ADL-based model is more reasonable than that of the SSL-based model.

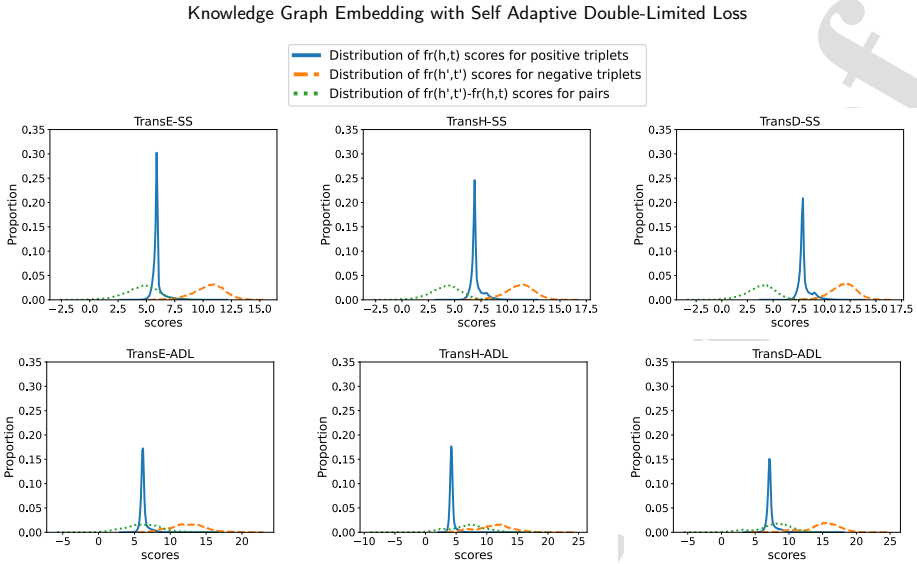


Figure 4: Distribution of triplets in testing stages (on FB15K)

4.3.2. Distribution of scores in the testing phase

To verify whether our proposed model can effectively avoid the problem of overlapping scoring regions for positive and negative triplets, we further analyze the score distributions of triplets. For FB15K, we use the training set to train the KGE and record the score distribution of the test set. We first generate negative triplets for the training and test sets with the 'bern' negative sampling method. For each pair of positive and negative triplets from test set, we calculate the score $f_r(h,t)$ for positive triplets, the score $f_r(h',t')$ for negative triplets and the margin-score $f_(h',t') - f_r(h,t)$ for pairs. In addition, we set the scoring interval to 0.1 and computed the proportion of triplet scores in (s-0.05, s+0.05] as the probability of score. The parameter settings of the SSL-based models refer to SSL[17]. The γ of TransE-ADL, TransH-ADL, and TransD-ADL are 7, 8, and 8, respectively, and other parameter settings are shown in Table 5.

According to these settings, the SSL-based models expect the scores of the positive and negative triplets as $f_r(h,t) < \gamma_1$ and $f_r(h',t') > \gamma_2$, while the ADL-based models expect $f_r(h,t) < \gamma - \frac{\lambda f_r(h,t)}{k+f_r(h',t')}$ and $f_r(h',t') > \gamma + \frac{\lambda f_r(h,t)}{k+f_r(h',t')}$. In Figure 4, the peaks of the $f_r(h,t)$ distribution (solid lines) for TransE-SS, TransH-SS and TransD-SS are 0.3, 0.25 and 0.22, respectively, while for TransE-ADL, TransH-ADL and TransD-ADL are 0.17, 0.18 and 0.16. In addition, comparing the peaks of the marginal-scores distribution (dot lines) for the SSL-based models and the ADL-based models, it can be found that the peaks of ADL get higher marginal scores than SSL. It demonstrates that the ADL-based can effectively alleviate the over-concentration of scores for positive triplets and distinguish positive triplets from negative triplets more clearly.

To get a more intuitive view of the number of pairs (dot lines) for $f_r(h',t') - f_r(h,t) \leq 0$, we counted the pairs that satisfy positive score higher than negative score, as shown in Table 7. the statistics show that TransE-ADL, TransH-ADL and TransD-ADL were compared to TransE-SS, TransH-SS and TransD-SS, the number of pairs satisfying the condition $f_r(h',t') - f_r(h,t) \leq 0$ is reduced by 68, 25 and 52, respectively. In summary, ADL-based models weaken the highly concentrated problem of triplet scores, increase the distance between positive and negative elements in a triplet, and reduce the number of $fr(h',t') - fr(h,t) < 0$. From these phenomena, we conjecture that the self adaptive double-limited loss function correctly distinguishes some of the hard positive-negative pairs in the KGs, thus enhancing the ability of the model's link prediction.

4.4. Performance analysis on different γ and λ

The loss function of ADL involves two critical parameters, γ, which is a center of margin to split positive and corresponding negative triplets, and λ, which is a scale factor. To investigate the sensitivity of these two parameters, we conduct a series of experiments on WN18 and FB15K.

To explore the effect of the hyperparameters γ and λ on the model, we set the values of γ to {0.5, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0, 11.0, 12.0, 13.0, 14.0, 15.0} or λ to {0.5, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0}, in that order. The control variable method was used to control the experiment's variables as a single variable, either γ or λ. The other parameters of the model are set with refer to table 5. We perform link prediction experiments using different

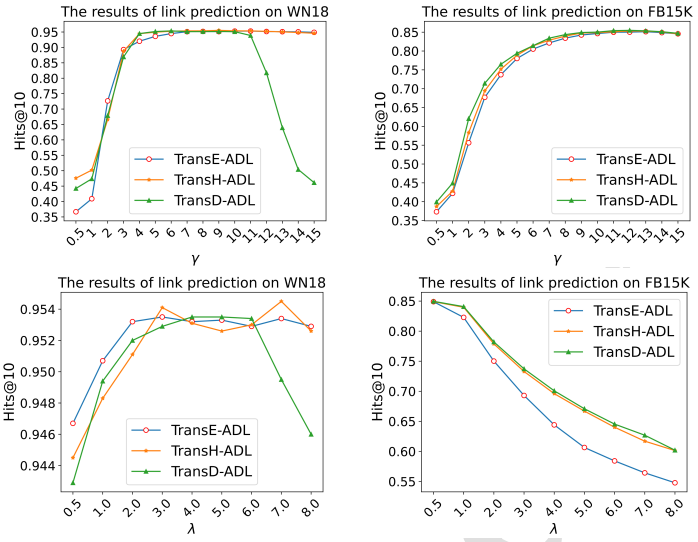


Figure 5: The impact of different γ and λ

γ and λ, then record the results of hits@10 and plot them. As shown in Fig.5, we can observe the following: 1) For WN18, the best margin of center γ is between 4 and 8, while the best λ is the same value as the best γ for TransE-ADL, TransH-ADL, and TransD-ADL. 2) For FB15K, the best γ is higher than 7, but the best λ is lower than 1. 3)The optimal value of λ in FB15K is smaller than that in WN18, indicating that the scoring of triplets requires a higher score on FB15K, further reflecting that FB15K has more complex relationships triplets than WN18. From Fig.5, we can also find that the ADL-based model is very sensitive to the hyperparameters γ and λ. When the optimal combination of γ and λ is found, our model can effectively improve the link prediction. But when the optimal combination of γ and λ is not exactly found, the performance of link prediction performs poorly and fluctuates to some extent. This causes our model to spend a lot of time on hyperparameter search, especially when compared to the MRL-based model.

4.5. Defect analysis

From the previous sections, we know that our proposed ADL can adaptively adjust the margin according to the difficulty of differentiating positive and negative triplets thus improving the model performance in link prediction. Although the ADL loss function can solve the hard differentiation problem between positive and negative triplets in KGE, it generates some drawbacks.

Generally, the ADL is a piecewise linear function (i.e. relu), which has the advantages of simplicity of architecture, good interpretability, and light computational load. But in comparison with those sigmoid based loss functions, its accuracy is slightly lower.

Specifically, the ADL loss can be split into two parts: $L_{pos} = f_r(h,t) - \gamma + \frac{\lambda f_r(h,t)}{k+f_r(h',t')}$ and $L_{neg} = \frac{\lambda f_r(h,t)}{k+f_r(h',t')} + \gamma - f_r(h',t')$. From Section 3.4, it is known that, whichever part is greater than 0, each element of positive and negative triplets has a nonzero gradient. And, we expect the positive triplet score to be as large as possible, while the negative triplet score $f_r(h,t)$ to be as large as possible, while the negative triplet score $f_r(h',t')$ of to be as large as possible. Thus, we analyze two extreme cases that may exist in the gradient update:

(1) Assume that $f_r(h',t') \to +\infty$, then $L_{pos} \approx f_r(h,t) - \gamma$. If $L_{pos} > 0$, the gradient for $f_r(h,t)$ is $1 + \frac{\lambda}{k+f_r(h',t')} \approx 1$, and the gradient for $f_r(h',t')$ is $-\frac{\lambda f_r(h,t)}{(k+f_r(h',t'))^2} \approx 0$.

(2) Assume that $f_r(h,t) = 0$, then $L_{neg} \approx \gamma - f_r(h',t')$. If $L_{neg} > 0$, the gradient for $f_r(h',t')$ is $-\frac{\lambda f_r(h,t)}{(k+f_r(h',t'))^2} - 1 = -1$, and the gradient for $f_r(h,t)$ is $\frac{\lambda}{k+f_r(h',t')} > \frac{\lambda}{k+\gamma} > 0$.

From case (1), we can find that, when the negative triplet score is positive infinity, the gradient is extremely small and does not affect the elements in the negative triplet. However, in case (2), when the positive triplet score is 0, there is possibly a large gradient on the elements in the positive triplet. We hope the update will stop when the positive triplet score is 0, but in this case, it still updates due to a large gradient. It is a defect of this loss function. We conjecture this is the reason why the Hits@1 of TorusE-ADL is slightly worse than that of TorusE in the dataset FB15K, as show in the table 4.

Moreover, the ADL loss, as a double-limited function, has one more hyperparameter than the MRL and is sensitive

to the hyperparameters γ and λ, which increases the parameter search time, as discussed in detail in Subsection 4.4.

5. Conclusion

In this paper, we design a new loss framework, ADL, which relies on the center of the margin to adjust the upper and lower bounds adaptively. First, to achieve a larger margin for the more indistinguishable positive-negative pairs, we design a discriminator to calculate the degree of indiscernibility of pairs and adjust the upper and lower bounds according to the result of the discriminator. Then, we propose TransE-ADL, TransH-ADL,TransD-ADL, TorusE-ADL, and ComplEx-ADL based on several popular KGE models. Finally, the experimental results in link prediction show that our proposed models significantly improve prediction performance compared with the corresponding KGE models.

References

[1] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka, T. M. Mitchell, Toward an architecture for never-ending language learning, in: Twenty-Fourth AAAI conference on artificial intelligence, 2010.

[2] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, 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, 2008, pp. 1247–1250.

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

[4] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer, et al., Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia, Semantic web 6 (2015) 167–195.

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

[6] S. He, K. Liu, Y. Zhang, L. Xu, 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), 2014, pp. 1092–1103.

[7] S. Zhu, X. Cheng, S. Su, Knowledge-based question answering by tree-to-sequence learning, Neurocomputing 372 (2020) 64–72.

[8] F. Zhang, N. J. Yuan, D. Lian, X. Xie, 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, 2016, pp. 353–362.

[9] B. Hu, Y. Ye, Y. Zhong, J. Pan, M. Hu, Transmkr: Translation-based knowledge graph enhanced multi-task point-of-interest recommendation, Neurocomputing (2021).

[10] M. E. Nissen, Knowledge-based knowledge management in the reengineering domain, Decision Support Systems 27 (1999) 47–65.

[11] S. Szumlanski, F. Gomez, Automatically acquiring a semantic network of related concepts, in: Proceedings of the 19th ACM international conference on Information and knowledge management, 2010, pp. 19–28.

[12] R. Socher, D. Chen, C. D. Manning, A. Ng, Reasoning with neural tensor networks for knowledge base completion, in: NIPS, 2013.

[13] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, Advances in neural information processing systems 26 (2013).

[14] Z. Wang, J. Zhang, J. Feng, Z. Chen, Knowledge graph embedding by translating on hyperplanes, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 28, 2014.

[15] G. Ji, K. Liu, S. He, J. Zhao, Knowledge graph completion with adaptive sparse transfer matrix, in: Thirtieth AAAI conference on artificial intelligence, 2016.

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

[17] X. Zhou, L. Niu, Q. Zhu, X. Zhu, P. Liu, J. Tan, L. Guo, Knowledge graph embedding by double limit scoring loss, IEEE Transactions on Knowledge and Data Engineering (2021).

[18] A. Rossi, D. Barbosa, D. Firmani, A. Matinata, P. Merialdo, Knowledge graph embedding for link prediction: A comparative analysis, ACM Transactions on Knowledge Discovery from Data (TKDD) 15 (2021) 1–49.

[19] D. Q. Nguyen, K. Sirts, L. Qu, M. Johnson, Stranse: a novel embedding model of entities and relationships in knowledge bases, arXiv preprint arXiv:1606.08140 (2016).

[20] W. Zhang, B. Paudel, W. Zhang, A. Bernstein, H. Chen, Interaction embeddings for prediction and explanation in knowledge graphs, in: Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, 2019, pp. 96–104.

[21] Z. Zhang, J. Cai, Y. Zhang, J. Wang, Learning hierarchy-aware knowledge graph embeddings for link prediction, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, 2020, pp. 3065–3072.

[22] T. Ebisu, R. Ichise, Toruse: Knowledge graph embedding on a lie group, in: Thirty-second AAAI conference on artificial intelligence, 2018.

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

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

[25] G. Ji, S. He, L. Xu, K. Liu, 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), 2015, pp. 687–696.

[26] Q. Wang, Z. Mao, B. Wang, L. Guo, Knowledge graph embedding: A survey of approaches and applications, IEEE Transactions on Knowledge and Data Engineering 29 (2017) 2724–2743.

[27] M. Nickel, V. Tresp, H.-P. Kriegel, A three-way model for collective learning on multi-relational data, in: Icml, 2011.

[28] B. Yang, W.-t. Yih, X. He, J. Gao, L. Deng, Embedding entities and relations for learning and inference in knowledge bases, arXiv preprint arXiv:1412.6575 (2014).

[29] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, G. Bouchard, Complex embeddings for simple link prediction, in: International conference on machine learning, PMLR, 2016, pp. 2071–2080.

[30] H. Liu, Y. Wu, Y. Yang, Analogical inference for multi-relational embeddings, in: International conference on machine learning, PMLR, 2017, pp. 2168–2178.

[31] I. Balažević, C. Allen, T. M. Hospedales, Tucker: Tensor factorization for knowledge graph completion, arXiv preprint arXiv:1901.09590 (2019).

[32] R. Socher, D. Chen, C. D. Manning, A. Ng, Reasoning with neural tensor networks for knowledge base completion, in: Advances in neural information processing systems, 2013, pp. 926–934.

[33] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, M. Welling, Modeling relational data with graph convolutional networks, in: European semantic web conference, Springer, 2018, pp. 593–607.

[34] T. Dettmers, P. Minervini, P. Stenetorp, S. Riedel, Convolutional 2d knowledge graph embeddings, in: Thirty-second AAAI conference on artificial intelligence, 2018.

[35] M. Nayyeri, S. Vahdati, J. Lehmann, H. S. Yazdi, Soft marginal transe for scholarly knowledge graph completion, arXiv preprint

arXiv:1904.12211 (2019).

[36] M. Nayyeri, X. Zhou, S. Vahdati, H. S. Yazdi, J. Lehmann, Adaptive margin ranking loss for knowledge graph embeddings via a correntropy objective function, arXiv preprint arXiv:1907.05336 (2019).

[37] B. Shi, T. Weninger, Proje: Embedding projection for knowledge graph completion, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 31, 2017.

[38] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proceedings of the thirteenth international conference on artificial intelligence and statistics, JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[39] A. Bordes, X. Glorot, J. Weston, Y. Bengio, Joint learning of words and meaning representations for open-text semantic parsing, in: Artificial Intelligence and Statistics, PMLR, 2012, pp. 127–135.

[40] A. Bordes, J. Weston, R. Collobert, Y. Bengio, Learning structured embeddings of knowledge bases, in: Twenty-Fifth AAAI Conference on Artificial Intelligence, 2011.

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

[42] B. Yang, W.-t. Yih, X. He, J. Gao, L. Deng, Learning multi-relational semantics using neural-embedding models, arXiv preprint arXiv:1411.4072 (2014).

[43] M. Nickel, L. Rosasco, T. Poggio, Holographic embeddings of knowledge graphs, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 30, 2016.

[44] F. Akrami, M. S. Saeef, Q. Zhang, W. Hu, C. Li, Realistic re-evaluation of knowledge graph completion methods: An experimental study, in: Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, 2020, pp. 1995–2010.

[45] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, M. Gamon, Representing text for joint embedding of text and knowledge bases, in: Proceedings of the 2015 conference on empirical methods in natural language processing, 2015, pp. 1499–1509.

LaTeX Source Files

Author Statement

Xiaoying Zou: Methodology, Validation, Writing - original draft
Xizhao Wang: Supervision, Writing - review & editing, Funding acquisition
Si Cen: Methodology, Supervision, Writing - review & editing
Guoquan Dai: Methodology, Supervision, Writing - review & editing
Chao Liu: Methodology, Supervision, Writing - review & editing

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: