Pattern-Aware and Noise-Resilient Embedding Models

Mojtaba Nayyeri[1,3], Sahar Vahdati[2,3], Emanuel Sallinger[5,3], Mirza Mohtashim Alam[1,3], Hamed Shariat Yazdi[3], and Jens Lehmann[3]

[1] University of Bonn, Bonn, Germany
nayyeri@cs.uni-bonn.de, shariatyazdi@gmail.com
[2] InfAI Lab, Dresden, Germany (vahdati,mohtasim)@infai.org
[3] University of Oxford, Oxford, United Kingdom emanuel.sallinger@cs.ox.ac.uk
[4] TU Wien, Vienna, Austria sallinger@dbai.tuwien.ac.at
[5] Fraunhofer IAIS, Dresden, Germany jens.lehmann@iais.fraunhofer.de

Abstract. Knowledge Graph Embeddings (KGE) have become an important area of Information Retrieval (IR), in particular as they provide one of the state-of-the-art methods for Link Prediction. Recent work in the area of KGEs has shown the importance of relational patterns, i.e., logical formulas, to improve the learning process of KGE models significantly. In separate work, the role of noise in many knowledge discovery and IR settings has been studied, including the KGE setting. So far, very few papers have investigated the KGE setting considering both relational patterns and noise. Not considering both together can lead to problems in the performance of KGE models. We investigate the effect of noise in the presence of patterns. We show that by introducing a new loss function that is both pattern-aware and noise-resilient, significant performance issues can be solved. The proposed loss function is model-independent which could be applied in combination with different models. We provide an experimental evaluation both on synthetic and real-world cases.

Keywords: Knowledge Graph · Embedding · Noise · Relational Pattern.

1 Introduction

Knowledge Graph Embeddings (KGE) have become an important area of information retrieval, in particular as they provide one of the state-of-the-art methods for Link Prediction. Embedding models typically receive Knowledge Graphs as a set of *correct* edges , i.e., triples in the form of (*subject*, *relation*, *object*) representing a fact such as (*Student1*, *isPhDin*, *Group1*). The role of KG embedding models is to take the symbolic representation (i.e., edges) and embed it into a vector space. For learning such embeddings, a typical scenario is to use existing edges as positive samples. Also, negative samples are generated by applying random corruptions to positive samples. As described so far, we see the ideal scenario, yet once knowledge graph embeddings are applied in real word settings, we see two factors: the appearance of *patterns* between relations, and the

existence of *noise*. Let us first consider the former. Recent work in the KGE area has shown the importance of *relational patterns*, i.e., logical formulas, to improve the learning process of KGE models significantly [9]. Relations between entities of a KG often form particular patterns which can, e.g., be represented as logical rules. Such patterns can be either stated directly based on domain knowledge or inferred statistically based on the data. For example, if we know the following statistical pattern from the data, namely that $(Student, isPhDin, Group)$ and $(Group, ledBy, Leader)$ in most cases implies $(Student, supervisedBy, Leader)$, we may expect our KGE model to infer such a fact as well. Such relational patterns can be injected into the learning process, or be statistically inferred during the learning process of a KGE model [7,8,21,5]. The other factor when we hit the real world, apart from patterns, is the existence of *noise*. The role of noise has been studied in many knowledge-based settings [10], including the KGE setting [24]. It is very hard in reality to distinguish noise (i.e., incorrect edges) from correct edges [20,12,25]. Noise in the presence of patterns in many off-the-shelf KGE models actually propagates along relational patterns. One of the reasons for this can be that the original edges are actually not correct, but noise. Another reason is that noise may, via relational patterns, lead to the creation of edges creating further contradictions. We frequently encounter situations where we have a number of such contradictions coming from incorrect triples, i.e., noise, as well as contradictions coming from edges learned via patterns. So far, very few papers have investigated the KGE setting considering *both* relational patterns and noise, which, allows one to overcome significant problems in the performance of KGEs. We investigate the effect of noise in the presence of patterns and, specifically, show that, noise on a particular edge (triple) will, via patterns, affect the score of other edges. We introduce a new loss function UNITE that is both noise-resilient and pattern-aware, i.e., allows to provide good performance even in the presence of noise and relational patterns. This new loss function is model independent and can be employed across KGE models. We provide an experimental evaluation, both on synthetic KGs where noise is explicitly introduced based on known patterns, and a large-scale real-world evaluation, where noise is randomly introduced on mostly unknown patterns.

2 Motivating Example

To directly illustrate the destructive effect of noise in combination with relational patterns, let us consider an example. In the lower part of Figure 1, we see a Knowledge Graph describing academic research groups (shown in yellow and marked as "group" in the diagram), students (shown in orange) and group leaders (shown in blue). We have three relations shown as directed edges, namely that a group can be led by a group leader, a student can be supervised by a group leader, and a student can work in, or be doing a PhD in a group. We here zoom-in into one portion of the graph to highlight typical relationships – other areas are shown in gray color in the background (for 22 triples in our example).

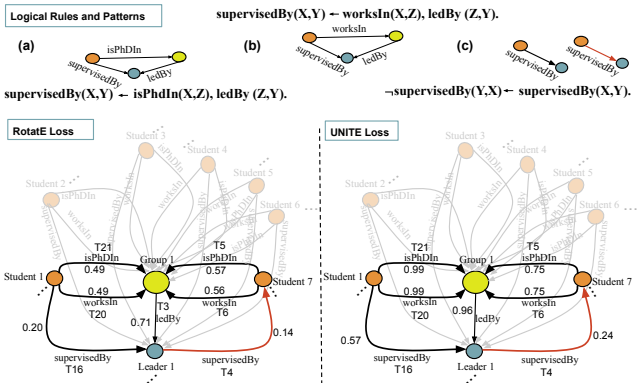

Fig. 1. Comparison of scoring triple correctness between RotatE and the UNITE – with three relational patterns in a KG in presence of noise.

Noise Noise Noise

	Hit@1 w/o noise	Hit@1 w/ noise	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	T18	T19	T20	T21	T22
UNITE	1.00	1.00	0.028	0.966	0.965	0.242	0.750	0.750	0.630	0.943	0.941	0.623	0.949	0.948	0.573	1.000	0.998	0.573	0.999	0.998	0.000	0.994	0.993	0.9
AdverserialLoss	1.00	0.66	0.000	0.719	0.715	0.148	0.570	0.565	0.470	0.728	0.724	0.428	0.770	0.771	0.325	0.876	0.872	0.202	0.998	0.996	0.225	0.495	0.490	1.0
MarginRankingLoss	1.00	0.50	0.464	0.492	0.373	0.366	0.655	0.311	0.554	0.606	1.000	0.818	0.767	0.759	0.594	0.250	0.870	0.000	0.434	0.499	0.392	0.664	0.836	0.48

Fig. 2. Loss functions and noise with repeated patterns of Figure 1.

The key aspects of the problem are relational patterns on the one hand, and noise on the other hand. One simple example of noise is shown through the red edge in Figure 1, which does not hold in the model world, but is present in our data set. On the upper side of Figure 1, we see the relational patterns of our model world where (a) and (b) are composition, and (c) antisymmetry. Figure 2 shows ranks (Hit@1) of the triples from TransE model for this scenario with three loss functions including ours. The second and third columns are the setting without and with noise, respectively. In Figure 2, we see the three edges marked as noise, i.e., incorrect edges. We record the resulting classification as correct (blue) and incorrect (red). While there are differences between the scores for our three noise tuples with the three loss functions, one could still consider them adequate, as the classification is in all cases negative. That is, for edges that represent noise, the scores are not truly problematic. The reason for the lower evaluation metrics becomes apparent on those edges shown in Figure 2 such as T3 or T16 which do not represent noise, but are related to noise via relational patterns. We see in these cases that the scores under margin ranking loss are in most cases the worst, followed by adversarial loss, while UNITE loss is clearly the least affected. Intuitively, UNITE dampens the effect of noise propagating along relational patterns. This is in particular highlighted when looking at the lower left part of Figure 1 (where scores under the adversarial loss are annotated above

edges) comparing it to the lower right part of Figure 1 (where scores under the UNITE loss are annotated above edges).

3 Related Work

Here, we review highlights of related contributions considering both the score and loss functions of models which fall in the same category as ours, namely distance models, i.e., translation-based, or rotation-based models.

Score Functions. The score function of a KGE model ($f_r(s,o)$) takes the embeddings of a triple, i.e., $(\mathbf{s},\mathbf{r},\mathbf{o})$ and returns a value – often denoted as $f^r_{s,o}$ – indicating the extent to which a triple is plausible in the embedding space. We consider one baseline (TrasnE) from translation-based embedding model and one state-of-the-art (RotatE) rotation-based KGE model.

TransE [1] model takes a vector representation of a triple $(\mathbf{s},\mathbf{r},\mathbf{o})$ and models its relation $(\mathbf{r})$ as a translation from subject $(\mathbf{s})$ to object $(\mathbf{o})$, i.e., $\mathbf{s}+\mathbf{r}\approx\mathbf{o}$. The score function of TransE is formulated as $\|\mathbf{s}+\mathbf{r}-\mathbf{o}\|$.

RotatE [21] forces $\mathbf{s}_j\mathbf{r}_j \approx \mathbf{o}_j$ to hold for all $j\in\{0,\ldots,d\}$ per each given triple $(\mathbf{s},\mathbf{r},\mathbf{o})$. The model performs a rotation of the j-th element s_j of the subject vector $\mathbf{s}$ by the j-th element $r_j = e^{i\theta_{r_j}}$ of a relation vector $\mathbf{r}$ to get the j-th element o_j of the object vector $\mathbf{t}$, where θ_{r_j} is the phase of the relation r.

Loss Functions. The loss function of a KGE ($\mathcal{L}$) is an optimization to adjust the embedding vectors. The losses of TransE and RotatE models have been reported strongly outperforming the other possible KGEs [19,21].

Margin Ranking Loss (MRL) has been primarily designed for training TransE and its variants. The loss optimization aims to put a margin (inspired by SVM [2]) between each positive sample (s,r,o) and its corresponding negative sample (s',r,o'), obtained by corruption in s, or o [1]. Let γ is a margin and $[x]_+ = max(0,x)$. MRL is defined as $\mathcal{L} = \sum_{(s,r,o)\in S^+}\sum_{(s',r,o')\in S^-_{(s,r,o)}}[\gamma + f_r(s,o) - f_r(s',o')]_+$. Despite the major use of MRL, it suffers from the margin sliding problem [15]. Therefore, there are many solutions which do not fulfil the translation, i.e., $\mathbf{s}+\mathbf{r}\neq\mathbf{o}$ [26]. The loss of the RotatE model is called *Adversarial Loss* which is defined as $\mathcal{L} = -\sum_{(s,r,o)\in S^+}\Big(\log\sigma(\gamma - f_r(s,o)) + \sum_{(s',r,o')\in S^-} p(s',r,o')\log\sigma(f_r(s',o')-\gamma)\Big)$, where $\sigma(.)$ is the Sigmoid function, $p(s',r,o') = \frac{exp(\alpha f_r(s',o'))}{\sum exp(\alpha f_r(s',o'))}$ is the probability of the triple (s',r,o') to be true negative, and α is the temperature of sampling. Other loss functions are also designed for particular usages [14,15].

Noise and KGE Models The sensitivity of KG embeddings to sparse and unreliable data is discussed in [16] without considering noise and relational patterns. GTransE [11] deals with uncertainty (meaning triple incorrectness) on KGs using dynamic and static weighting. In a different work, puTransE [22] proposes an approach to make KGEs semantically and structurally aware of noise, but it did not consider loss functions in particular. Node similarity Preserving (NSP) [17] proposes a loss function without considering multi-relational

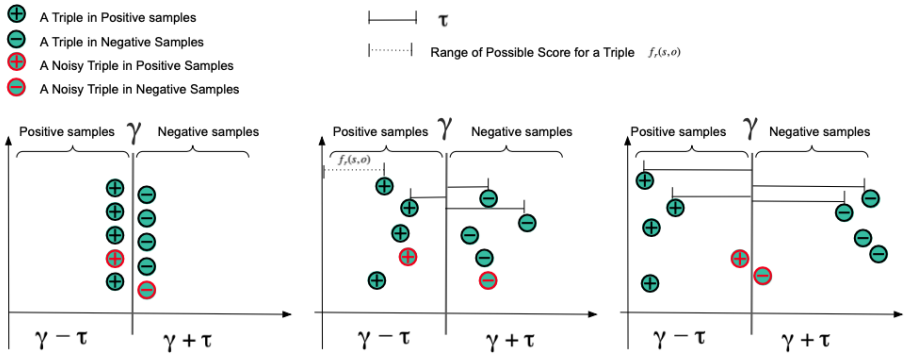

Fig. 3. Learning steps of UNITE. Initial state (left), intermediate state (middle) and final state (right).

KGs. Graph Denoising Policy Network (GDPNet) [23] focuses on an inductive approach from reinforcement learning on noise in scholarly KGs. Most of the other works have been focused on negative sampling, or feature selection for noise detection [20,12,25]. We focus on the role of loss functions in the existence of incorrect triples as noisy data and considers relational patterns.

Relational Patterns and KGE Models Early literature conjectured KGE models are evaluated in rules encoding [9,4,3,18]. Only some rule injection frameworks such as Ruge [8], KALE [7] and few embedding models such as RotatE [21] and its special case TorusE [5] have considered the issue of relational patterns. Specially, a recent work has theoretically and empirically proven that the score function of the RotatE model is capable of inferring various relational patterns including symmetric/antisymmetric, inverse and composition patterns. This inference ability is when for any pattern in the form of *premise* $\rightarrow$ *conclusion*, the model approves correctness of *conclusion* when the correctness of *premise* is confirmed. Overall, the existence of noise in KGs in the context of relational patterns has not been addressed.

4 Method

Given a KGE model, a typical optimization framework for link prediction consists of the following steps: (1) initialization of embedding vectors, (2) setting criteria (e.g., margin, square error), and (3) optimizing a loss function in an iterative way to enforce that embedding vectors satisfy the criteria. In order to unify the power of implicit patterns shaped inside the underlying KG and mitigate the negative effect of noise, we adapt these steps in a proposed optimization framework named UNITE model.

In step (2), i.e., setting criteria, we consider a point $\gamma \in [0, 1]$ as discriminator for separation of positive and negative samples. Relative to γ, the correctness/incorrectness of a triple (s, r, o) is measured by its distance $(\tau^r_{s,o})$ to

the discriminator. Note that $\tau^r_{s,o}$ is initialized to zero originally, and during the learning process will, in general, increase. UNITE aims at adjusting the distance as well as embedding vectors in an iterative process to reduce the degree of correctness of implicit noisy triples while the model learns from patterns that the triple is wrongly labeled as positive or negative.

To illustrate the process of step (3), i.e., optimization, we introduce Figure 3 which guides us through this method section, and the components of which we will introduce step by step in this section. Overall, we see three states of the learning process: On the left side of Figure 3 the initial state, on the right side the final state and an intermediate state in the middle. The boundary γ, illustrated in each state in the figure, separates positive and negative samples. The distance to γ is given by $\tau^r_{s,o}$, and illustrated in Figure 3. In the rest of this section, We first design a loss function and explain the process to perform optimization for positive samples. This will allow us to understand the areas to the left of γ in Figure 3. The same procedure is explained for negative samples, providing an intuition for the areas to the right of γ. Finally, the two designed parts are united by proposing the optimization framework UNITE, which allows us to understand Figure 3.

4.1 UNITE Loss for Positive Samples

As briefly introduced in the previous section, apart from the embedding itself, the primary values to be optimized during the optimization phase is the distance $\tau^r_{s,o}$. We will now describe, bottom-up, how the loss function and the optimization problem is defined for positive samples. Let us first define domain and range of $\tau^r_{s,o} : S^+ \rightarrow [0, \infty]$. More specifically, by constraints of the optimization problem introduced later, the effective range of $\tau^r_{s,o}$ will actually be constrained to be $[0, \gamma]$. Within our framework, we will apply a probability function P to $\tau^r_{s,o}$, and use the notation $P^r_{s,o} = P(\tau^r_{s,o})$. The specific choice of such a probability function is up to the user of the framework. In our evaluation, we use a Gaussian function with the variance optimized as a hyper-parameter. Precise definitions will be given in the evaluation section, where experiments for particular configurations of the framework are performed. Intuitively, values $P^r_{s,o}$ for positive samples have the following meaning: 1) a triple with $P^r_{s,o} = 0$, has the highest probability of being correct ("positive"); 2) a triple with $P^r_{s,o} = 1$, has the lowest probability of being correct ("unknown"); 3) a triple with $P^r_{s,o}$ between 0 and 1 describes the extent to which a triple is considered as positive or unknown. In the beginning of the learning process, $\tau^r_{s,o}s$ are randomly assigned to a very small value ($\tau^r_{s,o}s \approx 0$). Therefore, $P^r_{s,o}s$ are very high in the beginning of the learning process ($P^r_{s,o} \approx 1$). The ultimate objective is to minimize the loss function. The embedding vectors and $\tau^r_{s,o}s$ are optimized in an iterative process. We define the *loss function* as:

$$\mathcal{L}^+ = \prod_{(s,r,o) \in S^+} P^r_{s,o} \tag{1}$$

where S^+ is the set of all triples in the KG. We define the *objective function of the optimization problem* as follows:

$$\min_{\{(\mathbf{s},\mathbf{r},\mathbf{o})\}\in\mathbf{S}^+,\tau^r_{s,o}} \mathcal{L}^+ \tag{2}$$

where $\mathbf{S}^+$ is the embedding of all entities and relations. We apply a second probability function $\mathcal{Q}$ for the purpose of defining constraints for our optimization problem. The specific choice of such a probability function is again up to the user of the framework. In our evaluation, we use a Sigmoid function. As before, precise definitions will be given in the evaluation section. We now give the *constraint* of our optimization problem

$$\mathcal{Q}(\gamma - f^r_{s,o}) \geq \mathcal{Q}(\tau^r_{s,o}). \tag{3}$$

Observe that this constraint effectively limits $\tau^r_{s,o}$ to be no larger than γ, and it forces the score $f^r_{s,o}$ to be in the range $[0, \gamma]$ as well. This yields the following optimization problem using the objective function from Equation 2 and the constraint from Equation 3:

$$\begin{cases} \min_{\{(\mathbf{s},\mathbf{r},\mathbf{o})\}\in\mathbf{S}^+,\tau^r_{s,o}} \prod_{(s,r,o)\in S^+} P^r_{s,o}\,, \\ \text{s.t. } \mathcal{Q}(\gamma - f^r_{s,o}) \geq \mathcal{Q}(\tau^r_{s,o}). \end{cases} \tag{4}$$

Considering the fact that $\min \mathcal{L}$ is equivalent to $\min \log(\mathcal{L})$, the following optimization problem is solved instead of Equation 4:

$$\begin{cases} \min_{\{(\mathbf{s},\mathbf{r},\mathbf{o})\}\in\mathbf{S}^+,\tau^r_{s,o}} \sum_{(s,r,o)\in S^+} \log P^r_{s,o}\,, \\ \text{s.t. } \mathcal{Q}(\gamma - f^r_{s,o}) \geq \mathcal{Q}(\tau^r_{s,o}). \end{cases} \tag{5}$$

This essentially makes the mathematical operations applied simpler, while still solving the same optimization problem.

4.2 UNITE Loss for Negative Samples

Existing KGE models mostly generate negative samples by corruption of positive samples. In this paper, we consider one of the simplest corruption techniques namely *uniform negative sampling* used in [1]. To this end, either subject (s) or object (o) of a given positive triple (s, r, o) is replaced by an entity (s' or o'). A candidate entity (s' or o') is selected randomly using uniform distribution. Let the set $S^-_{(s,r,o)}$ denotes all such corruptions of the triple (s, r, o), and let S^- denote the overall set of all the negative samples. Due to randomness of negative sample generation, there is always uncertainty in the negative samples. The definition of the optimization problem for negative samples follows similar principles as the one described before for positive samples: 1) a triple with $P^r_{s',o'} = 0$, has the highest probability of being incorrect ("negative"); 2) a triple with $P^r_{s',o'} = 1$, has the lowest probability of being incorrect ("unknown"); 3) a triple with $P^r_{s',o'}$ between 0 and 1 describes the extent to which a triple is considered as negative or unknown. The constraint for negative samples is $\mathcal{Q}(f^r_{s',o'} - \gamma) \geq \mathcal{Q}(\tau^r_{s',o'})$ where S^- is the set of all negative samples and $\mathbf{S}^-$ is the set of all embeddings of entities and relations participating in the negative sample set S^-.

4.3 United Optimization

We now merge the two optimization problems previously formulated for positive and negative samples, completing the overall situation illustrated in Figure 3. There are two possible assumptions for uniting the formulations of positive and negative samples, namely: independent uncertainty and dependent uncertainty.

Independent uncertainty: this assumption indicates that although a negative sample (s', r, o') is generated by corruption of either subject or object of the positive sample (s, r, o), the degrees of uncertainty for positive and negative samples are independent (UNITE-I). Therefore, we set two different parameters for the positive and its negative sample i.e., $\tau^r_{s,o}$ for positive and $\tau^r_{s',o'}$ for negative samples. This can be achieved by simply combining the two optimization problems we introduced so far without any further modification, yielding the following optimization problems:

$$\begin{cases} \min_{\{(\mathbf{s},\mathbf{r},\mathbf{o})\}\in\mathbf{S}^+,\{(\mathbf{s}',\mathbf{r},\mathbf{o}')\}\in\mathbf{S}^-,\tau^r_{s,o},\tau^r_{s',o'}} \\ \quad \sum_{(s',r,o')\in S^-} \log P^r_{s',o'} + \sum_{(s,r,o)\in S^+} \log P^r_{s,o} \\ \text{s.t. } \mathcal{Q}(f^r_{s',o'} - \gamma) \geq \mathcal{Q}(\tau^r_{s',o'}) \ , \{(s', r, o')\} \in S^-, \\ \quad \mathcal{Q}(\gamma - f^r_{s,o}) \geq \mathcal{Q}(\tau^r_{s,o}),\ \{(s, r, o)\} \in S^+. \end{cases} \tag{6}$$

Dependent uncertainty: we set the same parameters for the positive and its negative sample to measure the degree of uncertainty dependently. Therefore, $\tau^r_{s,o}$ is used for both of the positive and its corresponding negative samples (UNITE-D). The formulation of this optimization is:

$$\begin{cases} \min_{\{(\mathbf{s},\mathbf{r},\mathbf{o})\}\in\mathbf{S}^+,\tau^r_{s,o}} \sum_{(s,r,o)\in S^+} \log P^r_{s,o}\,, \\ \text{s.t. } \mathcal{Q}(f^r_{s',o'} - \gamma) \geq \mathcal{Q}(\tau^r_{s,o})\,, \\ \{(s', r, o')\} \in S^-_{(s,r,o)},\ \{(s, r, o)\} \in S^+, \\ \mathcal{Q}(\gamma - f^r_{s,o}) \geq \mathcal{Q}(\tau^r_{s,o}),\ \{(s, r, o)\} \in S^+ \end{cases} \tag{7}$$

For both positive and negative samples, $\tau^r_{s,o}$ of the positive samples is used. Finally, in order to solve the optimization problems given by Equations 6 and 7, we bring the constraints to the objectives, as is usually done, and solve the unconstrained optimization problems using stochastic gradient descent.

The Role of $\boldsymbol{\tau}$. Let us focus on the negative samples in Figure 3 which are distributed in the right side of γ. This is effectively enforced by the constraint introduced in negative constrains which consequently enforce the eventual scores of negative samples to be bigger than γ. To understand the role of τ in determining the plausibility of triples in presence of noise, for example let r be a symmetric rule in the form of *premise* $\Longleftrightarrow$ *conclusion*, where *premise*, (a triple) and *conclusion* (a triple) have a common relation with different entities in head and tail positions i.e. $(s, r, o) \Longleftrightarrow (o, r, s)$. If we exemplify this on *colleagueOf* relation, then $(S, colleagueOf, O) \Longleftrightarrow (O, colleagueOf, S)$ In case $(S, colleagueOf, O)$ is a correct triple in a KG, then the plausibility

of $(O, colleagueOf, S)$ can be defined with different conditions: (a) it is a correct triple in positive samples, or (b) it is not in positive samples nor in negative samples, or (c) it is in negative samples (false negative) which creates a conflict with what the model is enforced to learn. Here we focus on case (c) when $(O, colleagueOf, S)$ is a false negative sample (noise) in the training set. If we use RotatE for this case (as one of the best reported distance-based model) with the score function $f^r_{s,o} = \|\mathbf{s} \circ e^{\theta_\mathbf{r}} - \mathbf{o}\|$ which is proven to be capable of inferring symmetric relations when $\theta_r = 0$ or π [21], therefore the triple $(O, colleagueOf, S)$ will be learned (inferred) as a positive triple. This poses a conflict between what the model infers about the plausibility of this triple from the patterns (positive), and what the model sees about the plausibility of this triple in the training set (negative). Here τ comes into play with an important role to resolve this conflict by giving the high uncertainty value to the false negative sample $(O, colleagueOf, S)$. For simplicity of explanation, let $\mathcal{Q}$ be a linear function (in negative constraint) and $\gamma = 0$, so we have $f^{colleagueOf}_{(O,S)} \geq \tau^{colleagueOf}_{(O,S)}$. Since $\theta_r = 0$ or π and $(S, colleagueOf, O)$ is positive, $f^{colleagueOf}_{(O,S)} \approx 0$ (is positive) , therefore, $\tau^{colleagueOf}_{(O,S)}$ is constrained to be close to zero (is positive). Therefore, the main optimization problem of Equation 6 considers $\tau \approx 0$ as an optimal solution. Therefore, the triple gets high uncertainty value based on τ i.e.$P^{colleagueOf}_{O,S} \approx 1$. Giving such high uncertainty to $(O, colleagueOf, S)$ enables the model to keep θ_r still close to 0 or π which preserves the ability of inferring a symmetric pattern. The opposite of this scenario is the case where $(S, colleagueOf, O)$ gets the score to be negative by the model because $(O, colleagueOf, S)$ is in the negative set and r is a symmetric relation ($\theta_r = 0, \pi$). However, this scenario is less likely to happen because there are other patterns and triples that by using them the model recognizes $(S, colleagueOf, O)$ as positive during learning process.

5 Evaluation

In this section, we evaluate the UNITE framework in the context of two other loss functions: margin ranking loss (baseline) and adversarial loss (state-of-the-art).

Evaluation Metrics. Three statistic measurement metrics are considered: Mean Rank (MR), Mean Reciprocal Rank (MRR) and Hits@K. In order to calculate MR, a corrupted version of the test set is created: (1) replacing the subject of triples by possible other entities, and (2) replacing the object of triples by all possible entities. For each triple (s, r, o) in the test set S, a sample set S_i for the i^{th} test triple is generated such that $S^s_i = (?, r, o), ? \in KG$. The same holds for the second round. Scores of all the generated triples (including the test triple (s, r, o)) are computed and sorted in S^s_i and S^o_i. Let $Rank_i = (Rank^s_i + Rank^o_i)/2$ denote the general rank of the i-th triples, the Mean Rank (MR) is computed as $MR = \frac{\sum Rank_i}{n_t}$ where n_t is the number of test triples. Hits@K is the rate of correct triples appearing in top K position. The filtered setting [1] is used for evaluation of our model.

Datasets. We used four standard benchmarks (statistics in Table 1) with the assumption that they contain implicit noise. AMIE [6] was used for rule mining. We contaminate FB15K by more than 100,000 random corrupted triples (about 20% noise) to evaluate the ability of UNITE to recognize and be resilient to noise. We keep the ratio relatively high (20%) however for other datasets, we stayed with lower ratio because it was compatible with the statistics of patterns in the whole dataset. The same ratio of noise is considered for WN18. For FB15K-237, we generate 5% random noise.

Dataset	Inv.	Imp.	Eql.	Sym.	#train	#valid	#test
FB15K	67,757	3,259	8,771	7,740	483,142	50,000	59,071
FB15K-237	4,645	578	861	-	272,115	17,535	20,466
WN18	116,464	-	-	-	141,442	5,000	5,000
WN18RR	-	-	-	-	6,084	3,034	3,134

Table 1. Dataset Statistics. Number of triples and patterns.

Training Setting. We train TransE and RotatE using optimization framework Equation 6 (UNITE-I) and 7 (UNITE-D). In our experiments, we use $P^r_{s,o} = \exp -\sigma {\tau^r_{s,o}}^2$ (Gaussian) and $\mathcal{Q}(x) = \frac{1}{1+\exp -x}$ (Sigmoid). The Sigmoid function is strictly monotone, therefore we use a linear function ($\mathcal{Q}$) instead to enforce the constraints. The embedding dimension d is set to be 200 and 10 negative sample are generated ($n = 10$). The hyper-parameter of the Gaussian function (σ) is fixed to 1000. The batch size is set to 1024 for training on FB15k and FB15k-237, and 512 for WN18 and WN18RR. The corresponding values for γ is from the set $\{0, 5, 10, 15, 20, 25, 30\}$.

5.1 Experimental Results

The results of our experiments have been divided into multiple parts. First, we evaluate UNITE with and without artificially generated noise. We additionally report the results of different models/losses with/without pattern injection. We also inject patterns (only for inverse) by adding a regularization term to the objective as in [13]. As shown in Table 2, UNITE achieves improvements in FB15K and FB15K-237 in terms of MRR and Hits@10 with and without noise. UNITE achieves an MRR of 74 for FB15K whereas MRL and adversarial loss achieve 62 and 73, respectively. The relatively good results of adversarial loss is due to its noise resiliency ability, however our model is particularly designed for being pattern-aware in presence of noise. In our evaluation setup with randomly generated noise, UNITE significantly outperforms margin ranking loss by more than 10%. As visible, the results of the RotatE model when trained with adversarial loss stay very close to UNITE. Our assessment for these results can be summarized as three points: 1) **lack of diversity in types of rules**: despite

Dataset			FB15K		FB15K-237		WN18	
			MRR	Hits@10	MRR	Hits@10	MRR	Hits@10
Adv. Loss	w/o noise	w/ RotatE	73.3	87.9	32.1	50.8	-	-
		inject TransE	66.7	86.3	30.8	50.3	-	-
		w/o RotatE	72.3	87.8	32.3	51.2	94.9	96.3
		inject TransE	68	86	31.3	51.3	70.7	95.2
	w/- noise	w/ RotatE	63.3	82.8	31.2	51.1	-	-
		inject TransE	32.3	75.7	30.7	50.2	-	-
		w/o RotatE	61.2	81.9	32.3	51.2	94.8	96.2
		inject TransE	32.2	74.8	31.3	49.8	69.8	94.7
MR Loss	w/o noise	w RotatE	62.3	81.1	-	-	-	-
		inject TransE	45.8	74.3	-	-	-	-
		w/o RotatE	60.8	80.7	27.9	47.3	94.2	94.3
		inject TransE	46.9	74.3	27.7	47.3	50.1	94.8
	w/- noise	w/ RotatE	52.3	72.8	-	-	91.9	93.8
		inject TransE	33.3	61.2	-	-	50.4	94.8
		w/o RotatE	50.3	72.7	27.9	46.8	92.3	94.2
		inject TransE	32.2	60.1	26.7	47.9	45.2	95.2
UNITE Loss	w/o noise	w/ RotatE	74.3	88.8	33.3	51.9	-	-
		inject TransE	67.9	86.8	31.1	51.2	-	-
		w/o RotatE	73.3	88.8	21.2	50.8	95.3	96.2
		inject TransE	69.3	86.3	31.3	51.1	75.9	95.8
	w/- noise	w/ RotatE	64.2	83.8	33.3	51.8	-	-
		inject TransE	33.3	76.2	30.9	50.8	-	-
		w/o RotatE	61.1	83.3	32.2	51.8	94.9	95.8
		inject TransE	32.2	76.2	30.9	51.2	74.8	95.8

Table 2. Evaluation Results. Comparisons of results for Adversarial Loss, Margin Ranking Loss and UNITE are depicted for FB15k and FB15k-237, and WN18.

the existence of patterns in FB15K, it lacks diversity of patterns; 2) **relatively small ratio of other pattern types to inverse**: the majority of the patterns are inverse relations and the rest belong to symmetric, implication and equivalence patterns Table 1; 3) **test set leakage**: not only the diversity but also the ratio of overall grounding of patterns has dropped dramatically in the case of FB15K-237. Considering the above points, of particular interest for future work will be in-depth studies on the existence of complex patterns, and advances in the area of pattern extraction, especially focusing on complex patterns. Both of the optimizations, UNITE-I and UNITE-D, perform closely – in all the previous evaluations, we only reported UNITE-I. UNITE-D gets a lower MR in both datasets of WN18 and FB15k with same result in Hits@1 on WN18. In FB15k for Hits@1, UNITE-I gets 82.2% while UNITE-D is performing with 81.7%. In Table 2 we reported the relevant results for WN18. TransE trained by UNITE obtains 76% on MRR, which significantly outperforms TransE with adversarial loss with MRR of 71%. In the case of WN18RR, as discussed before, the rule mining system that we used, AMIE, did not extract any patterns. Despite the

pattern-free characteristic of WN18RR, RotatE trained with UNITE was able to achieve 57% in hits@10 whereas MRL could only reach 37%.

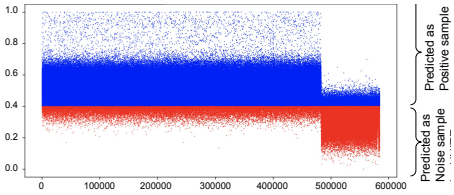

Fig. 4. Distribution of Scores in FB15K.

Correctness Prediction. In Figure 4, we illustrate the distribution of scores predicted by UNITE for FB15K. The X axis shows the index of triples and the Y axis presents normalized scores. Predicted correct triples (in blue) by UNITE are separated on 0.4 from predicted noisy triples (in red). Out of 483,142 triples in the train set of FB15k, 101,123 triples have been made noisy through a random procedure (21%). As shown, our model is able to assign a low score to most of the 21% generated noise triples. The triples in the upper right (in blue) are assumed to fall in this category. The lower left part of the plot shows the existence of noise (in red) in the positive samples. We manually validated several triples in Table 3.

Triple/Wikipedia	**Prediction**
`Harvey_Weinstein`($/m/05hj_k$) isSiblingOf `Bob_Weinstein`($/m/06q8hf$)	1.0 (Originally Positive - Identified Positive
- ($/m/07s9rl0$) hasSameGenre ($/m/06fvc$)	0.169 (Originally Positive - Identified Noise)

Table 3. Validation test shows correct and noisy triples identified by UNITE.

6 Conclusion

In this paper, we investigated KGE models in the presence of both relational patterns and noise. We introduced the new loss function UNITE and its variations UNITE-I and UNITE-D. We evaluated UNITE both within translation-based models (TransE) and rotation-based models (RotatE), and in synthetic and real-world scenarios. As future work, we plan to further investigate the effect of non-uniformly distributed random noise. This will need some advances in the area of detecting and extracting more complex relational patterns than current methods can do, but would be able to shed more light on the situation in which noise is specifically affecting patterns in large real-world scenarios.

Acknowledgements. We acknowledge the support of the EU projects TAILOR (GA 952215), Cleopatra (GA 812997), the BmBF project MLwin, ScaDS.AI (01/S18026A-F), WWTF (Vienna Science and Technology Fund) grant VRG18-013, the EPSRC grant EP/M025268/1, and the EU Horizon 2020 grant 809965.

References

1. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Advances in Neural Information Processing Systems. pp. 2787–2795 (2013)
2. Cortes, C., Vapnik, V.: Support vector machine. Machine learning **20**(3), 273–297 (1995)
3. Du, J., Qi, K., Shen, Y.: Knowledge graph embedding with logical consistency. In: Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data, pp. 123–135. Springer (2018)
4. Du, J., Qi, K., Wan, H., Peng, B., Lu, S., Shen, Y.: Enhancing knowledge graph embedding from a logical perspective. In: Joint International Semantic Technology Conference. pp. 232–247. Springer (2017)
5. Ebisu, T., Ichise, R.: Toruse: Knowledge graph embedding on a lie group. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
6. Galárraga, L.A., Teflioudi, C., Hose, K., Suchanek, F.: Amie: association rule mining under incomplete evidence in ontological knowledge bases. In: Proceedings of the 22nd international conference on World Wide Web. pp. 413–422 (2013)
7. Guo, S., Wang, Q., Wang, L., Wang, B., Guo, L.: Jointly embedding knowledge graphs and logical rules. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. pp. 192–202 (2016)
8. Guo, S., Wang, Q., Wang, L., Wang, B., Guo, L.: Knowledge graph embedding with iterative guidance from soft rules. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
9. Gutiérrez-Basulto, V., Schockaert, S.: From knowledge graph embedding to ontology embedding? an analysis of the compatibility between vector space representations and rules. In: Sixteenth International Conference on Principles of Knowledge Representation and Reasoning (2018)
10. Heindorf, S., Potthast, M., Stein, B., Engels, G.: Vandalism detection in wikidata. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management. pp. 327–336 (2016)
11. Kertkeidkachorn, N., Liu, X., Ichise, R.: Gtranse: Generalizing translation-based model on uncertain knowledge graph embedding. In: Annual Conference of the Japanese Society for Artificial Intelligence. pp. 170–178. Springer (2019)
12. Luo, S., Fang, W.: Potential probability of negative triples in knowledge graph embedding. In: International Conference on Neural Information Processing. pp. 48–58. Springer (2018)
13. Minervini, P., Costabello, L., Muñoz, E., Nováček, V., Vandenbussche, P.Y.: Regularizing knowledge graph embeddings via equivalence and inversion axioms. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 668–683. Springer (2017)
14. Nayyeri, M., Vahdati, S., Zhou, X., Yazdi, H.S., Lehmann, J.: Embedding-based recommendations on scholarly knowledge graphs. In: European Semantic Web Conference. pp. 255–270. Springer (2020)
15. Nayyeri, M., Zhou, X., Vahdati, S., Izanloo, R., Yazdi, H.S., Lehmann, J.: Let the margin slide±for knowledge graph embeddings via a correntropy objective function. In: 2020 International Joint Conference on Neural Networks (IJCNN). pp. 1–9. IEEE (2020)
16. Pujara, J., Augustine, E., Getoor, L.: Sparsity and noise: Where knowledge graph embeddings fall short. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 1751–1756 (2017)

17. Qiu, Z., Hu, W., Wu, J., Tang, Z., Jia, X.: Noise-resilient similarity preserving network embedding for social networks. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence. pp. 3282–3288. AAAI Press (2019)
18. by Representation, K.G.: Ranking diagnoses for inconsistent knowledge graphs by representation learning. In: Semantic Technology: 8th Joint International Conference, JIST 2018, Awaji, Japan, November 26–28, 2018, Proceedings. vol. 11341, p. 52. Springer (2018)
19. Ruffinelli, D., Broscheit, S., Gemulla, R.: You {can} teach an old dog new tricks! on training knowledge graph embeddings. In: International Conference on Learning Representations (2020)
20. Shan, Y., Bu, C., Liu, X., Ji, S., Li, L.: Confidence-aware negative sampling method for noisy knowledge graph embedding. In: 2018 IEEE International Conference on Big Knowledge (ICBK). pp. 33–40. IEEE (2018)
21. Sun, Z., Deng, Z.H., Nie, J.Y., Tang, J.: Rotate: Knowledge graph embedding by relational rotation in complex space. arXiv preprint arXiv:1902.10197 (2019)
22. Tay, Y., Luu, A.T., Hui, S.C.: Non-parametric estimation of multiple embeddings for link prediction on dynamic knowledge graphs. In: Thirty-First AAAI Conference on Artificial Intelligence (2017)
23. Wang, L., Yu, W., Wang, W., Cheng, W., Zhang, W., Zha, H., He, X., Chen, H.: Learning robust representations with graph denoising policy network. arXiv preprint arXiv:1910.01784 (2019)
24. Xie, R., Liu, Z., Lin, F., Lin, L.: Does william shakespeare really write hamlet? knowledge representation learning with confidence. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
25. Zhao, Y., Liu, J.: Scef: A support-confidence-aware embedding framework for knowledge graph refinement. arXiv preprint arXiv:1902.06377 (2019)
26. Zhou, X., Zhu, Q., Liu, P., Guo, L.: Learning knowledge embeddings by combining limit-based scoring loss. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. pp. 1009–1018. ACM (2017)