

# Treat Different Negatives Differently: Enriching Loss Functions with Domain and Range Constraints for Link Prediction

Nicolas Hubert<sup>1,2</sup>[0000–0002–4682–422X], Pierre Monnin<sup>3</sup>[0000–0002–2017–8426],  
Armelle Brun<sup>2</sup>[0000–0002–9876–6906], and Davy Monticolo<sup>1</sup>[0000–0002–4244–684X]

<sup>1</sup> Université de Lorraine, ERPI, Nancy, France

<sup>2</sup> Université de Lorraine, CNRS, LORIA, Nancy, France

<sup>3</sup> Université Côte d’Azur, Inria, CNRS, I3S, Sophia-Antipolis, France  
{nicolas.hubert,armelle.brun,davy.monticolo}@univ-lorraine.fr  
pierre.monnin@inria.fr

**Abstract.** Knowledge graph embedding models (KGEMs) are used for various tasks related to knowledge graphs (KGs), including link prediction. They are trained with loss functions that consider batches of true and false triples. However, different kinds of false triples exist and recent works suggest that they should not be valued equally, leading to specific negative sampling procedures. In line with this recent assumption, we posit that negative triples that are semantically valid w.r.t. signatures of relations (domain and range) are high-quality negatives. Hence, we enrich the three main loss functions for link prediction such that all kinds of negatives are sampled but treated differently based on their semantic validity. In an extensive and controlled experimental setting, we show that the proposed loss functions systematically provide satisfying results which demonstrates both the generality and superiority of our proposed approach. In fact, the proposed loss functions (1) lead to better MRR and Hits@10 values, and (2) drive KGEMs towards better semantic correctness as measured by the Sem@ $K$  metric. This highlights that relation signatures globally improve KGEMs, and thus should be incorporated into loss functions. Domains and ranges of relations being largely available in schema-defined KGs, this makes our approach both beneficial and widely usable in practice.

**Keywords:** Knowledge Graph Embeddings · Link Prediction · Schema-based Learning · Loss Functions.

## 1 Introduction

A knowledge graph (KG) is a collection of triples  $(h, r, t)$  where  $h$  (head) and  $t$  (tail) are two entities of the graph, and  $r$  is a predicate that qualifies the nature of the relation holding between them. In this work, we do not consider literals. KGs are inherently incomplete, incorrect, or overlapping and thus major refinement tasks include entity matching and link prediction [29]. The latter is the focus of

this paper. Link prediction (LP) aims at completing KGs by leveraging existing facts to infer missing ones. In the LP task, one is provided with a set of incomplete triples, where the missing head (resp. tail) needs to be predicted. This amounts to holding a set of triples where, for each triple, either the head  $h$  or the tail  $t$  is missing.

The LP task is often addressed using knowledge graph embedding models (KGEMs). They represent entities and relations as low-dimensional vectors in a latent embedding space that preserves as much as possible graph structural information and graph properties [15]. A plethora of KGEMs has been proposed in the literature. They usually differ w.r.t. their scoring function, *i.e.* how they model interactions between entities. Entities and relations embeddings are learned throughout several epochs, in an optimization process relying on loss functions. Loss functions aims at maximizing the score output by KGEMs for *positive* triples, *i.e.*, triples that exist in the graph, and minimizing the score of *negative* triples, *i.e.*, triples that are absent from the graph. A recent segment of the literature started investigating the influence of negative triples in the training of KGEMs [17]. Indeed, their generation – called negative sampling – is usually performed by corrupting true triples, *i.e.*, replacing their head or tail with another entity randomly chosen in the graph. Enhanced sampling mechanisms were then envisioned and, for example, involve selecting entities of the same type as the entity to replace [17].

The latter proposal comes within the scope of works considering the underpinning semantics of KGs as additional information to improve results w.r.t. the LP task. Over the past few years, semantic information has been incorporated in various parts of KGEMs, *e.g.*, as mentioned, the negative sampling procedure [14,18], but also the model itself [6,19,23,28,32], or the loss function [5,7,10,20]. Existing works proposing to include semantic information into loss functions showcase promising results. However, they are restricted to specific loss functions [7,10], or consider ontological axioms [7,20] that do not include domain and range axioms (or relation signatures) which are widely available in KGs. Interestingly, such axioms could be leveraged to generate different kinds of negative triples, *i.e.*, negative triples that respect relation signatures (called semantically valid) and those that do not (called semantically invalid). Additionally, to the best of our knowledge, negative triples were only studied in the sampling procedure, where specific ones are chosen to train on and the others are discarded. The possibility to sample all kinds of negatives but differently consider them within loss functions was left unassessed. This twofold observation motivates our first research question:

**RQ1** how main loss functions used in LP can incorporate domain and range constraints to differently consider negative triples based on their semantic validity?

Precedent work also pointed out the performance gain of incorporating ontological information as measured by rank-based metrics such as MRR and Hits@K [5,7,10,20]. However, while such approaches include semantic information as KGEM inputs, the semantic capabilities of the resulting KGEM are

left unassessed, even though this would provide a fuller picture of its performance [11,13]. Hence, our second research question:

**RQ2** what is the impact of incorporating relation signatures into loss functions on the overall KGEM performance?

To address both questions, we propose signature-driven loss functions, *i.e.* loss functions containing terms that depend on some background knowledge (BK) about types of entities and domains and ranges of relations. To broaden the impact of our approach, our work is concerned with the three most encountered loss functions in the literature: the pairwise hinge loss (PHL) [3], the 1-N binary cross-entropy loss (BCEL) [8], and the pointwise logistic loss (PLL) [27] (further detailed in Section 3). For each of them, we propose a tailored signature-driven version. The considered BK is available in many schema-defined KGs [9], which makes these newly introduced loss functions widely usable in practice. Furthermore, the impact of loss functions is evaluated using both rank-based metrics and Sem@K [11,13] – a metric that measures the consistency of KGEMs predictions for the LP task with relation signatures, *i.e.*, the semantic correctness of predictions.

To summarize, the main contributions of this work are:

- We propose signature-driven versions for the three mostly used loss functions for the LP task, leveraging BK about relation domains and ranges.
- We evaluate our approach in terms of traditional rank-based metrics, and also w.r.t. Sem@K, which gives more insight into the benefits of our proposal.
- We show that the designed signature-driven loss functions provide, in most cases, better performance w.r.t. both rank-based metrics and Sem@K. Consequently, our findings strongly indicate that signature information should be systematically incorporated into loss functions.

The remainder of the paper is structured as follows. Related work is presented in Section 2. In Section 3, we detail the signature-driven loss functions proposed in this work. Dataset descriptions and experimental settings are provided in Section 4. Key findings are presented in Section 5 and are further discussed in Section 6. Lastly, Section 7 sums up the main findings and outlines future research directions.

## 2 Related Work

This section firstly relates to former contributions that make use of semantic information to enhance model results regarding the LP task. Emphasis is placed on how semantic information can be incorporated in the loss functions (Section 2.1) – a research avenue which remains relatively unexplored compared to incorporating semantic information in other parts of the learning process, *e.g.* in the negative sampling or in the interaction function. Secondly, a brief background on the mainstream loss functions is provided (Section 2.2). This is to help position our contributions w.r.t. the vanilla loss functions used in practice.

## 2.1 Semantic-Enhanced Approaches

A significant body of the literature proposes approaches that incorporate semantic information for performing LP with KGEMs, with the purpose of improving KGEM performance w.r.t. traditional rank-based metrics.

The most straightforward way to do so is to embed semantic information in the model itself. For instance, AutoETER [23] is an automated type representation learning mechanism that can be used with any KGEM and that learns the latent type embedding of each entity. In TaRP [6], type information and instance-level information are simultaneously considered and encoded as prior probabilities and likelihoods of relations, respectively. TKRL [32] bridges type information with hierarchical information: while type information is utilized as relation-specific constraints, hierarchical types are encoded as projection matrices for entities. TKRL allows entities to have different representations in different types. Similarly, in [28], the proposed KGEM allows entities to have different vector representations depending on their respective types. TransC [19] encodes each concept of a KG as a sphere and each instance as a vector in the same semantic space. The relations between concepts and instances (`rdf:type`), and the relations between concepts and sub-concepts (`rdfs:subClassOf`) are based on the relative distance within this shared semantic space.

A few works incorporate semantic information to constrain the negative sampling (NS) procedure and generate meaningful negative triples [18,14,31]. For instance, type-constrained negative sampling (TCNS) [18] replaces the head or the tail of a triple with a random entity belonging to the same type (`rdf:type`) as the ground-truth entity. Jain *et al.* [14] go a step further and use ontological reasoning to iteratively improve KGEM performance by retraining the model on inconsistent predictions. It is noteworthy that our approach adopts an orthogonal direction of such works by proposing to sample all kinds of negatives but to treat them differently in the loss function when training.

A few work actually propose to include semantic information in the learning and optimization process. In [10], entities embeddings of the same semantic category are enforced to lie in a close neighborhood of the embedding space. However, their approach only fits single-type KGs. In addition, the only mainstream model benchmarked in this work is TransE [3], and only the pairwise hinge loss is used. Likewise, d’Amato *et al.* [7] solely consider the pairwise hinge loss, and their approach is benchmarked w.r.t. to translational models only. Moreover, BK is injected in the form of `equivalentClass`, `equivalentProperty`, `inverseOf`, and `subClassOf` axioms, similarly to [20] who incorporate `equivalentProperty` and `inverseOf` axioms as regularization terms in the loss function. However, the aforementioned axioms are rarely provided in KGs [9]. Cao *et al.* [5] propose a new regularizer called Equivariance Regularizer, which limits overfitting by using semantic information. However, their approach is data-driven and does not rely on a schema. In contrast, the approach presented in Section 3 leverages domain and range constraints which are available in most schema-defined KGs.

Finally, all the aforementioned semantic-driven approaches are only evaluated w.r.t. rank-based metrics. However, semantic-driven approaches would

benefit from a semantic-oriented evaluation. To the best of our knowledge, the work around Sem@K [11,12,13] is the only one to provide appropriate tools for measuring KGEM semantic correctness. Hence, our experiments will also be evaluated with this metric.

## 2.2 Loss Functions for the Link Prediction Task

Few works revolve around the influence of loss functions on KGEM performance [1,22,21]. Mohamed *et al.* [22] point out the lack of consideration regarding the impact of loss functions on KGEM performance. Experimental results provided in [1] indicate that no loss function consistently provides the best results, and that it is rather the combination between the scoring and loss functions that impacts KGEM performance. In particular, some scoring functions better match with specific loss functions. For instance, Ali *et al.* show that TransE can outperform state-of-the-art KGEMs when configured with an appropriate loss function. Likewise, Mohamed *et al.* [22] show that the choice of the loss function significantly influence KGEM performance. Consequently, they provide an extensive benchmark study of the main loss functions used in the literature. Namely, their analysis relies on a commonly accepted categorization between pointwise and pairwise loss functions. The main difference between pointwise and pairwise loss functions lies in the way the scoring function, the triples, and their respective labels are considered all together. Under the pointwise approach, the loss function relies on the predicted scores for triples and their actual label values, which is usually 1 for positive triple and 0 (or  $-1$ ) for negative triples. In contrast, pairwise loss functions are defined in terms of differences between the predicted score of a true triple and the score of a negative counterpart.

In our approach (Section 3), we consider the three most commonly used loss functions for performing LP [24]: the pairwise hinge loss (PHL) [3], the 1-N binary cross-entropy loss (BCEL) [8], and the pointwise logistic loss (PLL) [27]. Their vanilla formulas are recalled in Equations (1), (2), and (3).

$$\mathcal{L}_{PHL} = \sum_{t \in \mathcal{T}^+} \sum_{t' \in \mathcal{T}^-} [\gamma + f(t') - f(t)]_+ \quad (1)$$

where  $\mathcal{T}$ ,  $f$ , and  $[x]_+$  denote a batch of triples, the scoring function, and the positive part of  $x$ , respectively.  $\mathcal{T}$  is further split into a batch of positive triples  $\mathcal{T}^+$  and a batch of negative triples  $\mathcal{T}^-$ .  $\gamma$  is a configurable margin hyperparameter specifying how much the scores of positive triples should be separated from the scores of corresponding negative triples.

$$\mathcal{L}_{BCEL} = -\frac{1}{|\mathcal{E}|} \sum_{t \in \mathcal{T}} \ell(t) \log(f(t)) + (1 - \ell(t)) \log(1 - f(t)) \quad (2)$$

where  $\ell(t) \in \{0, 1\}$  denotes the true label of  $t$  and  $\mathcal{T}$  is a batch with all possible  $(h, r, *)$ .  $|\mathcal{E}|$  is the number of entities in the KG.

$$\mathcal{L}_{PLL} = \sum_{t \in \mathcal{T}} \log(1 + \exp^{-\ell(t) \cdot f(t)}) \quad (3)$$

where  $\ell(t) \in \{-1, 1\}$  denotes the true label of  $t$ .

### 3 Signature-driven Loss Functions

Building on the limits of previous work (Section 2.1), we propose signature-driven loss functions that extend the three most frequently used loss functions [24] and leverage BK about domains and ranges of relations, which are provided in many KGs used in the literature.

The purpose of the proposed loss functions is to distinguish *semantically valid negatives* from *semantically invalid ones*. The former are defined as triples  $(h, r, t)$  respecting both the domain and range of the relation  $r$ , *i.e.*,

$$\text{type}(h) \cap \text{domain}(r) \neq \emptyset \wedge \text{type}(t) \cap \text{range}(r) \neq \emptyset$$

whereas the latter violate at least one of the constraints, *i.e.*

$$\text{type}(h) \cap \text{domain}(r) = \emptyset \vee \text{type}(t) \cap \text{range}(r) = \emptyset.$$

$\text{domain}(r)$  (resp.  $\text{range}(r)$ ) is defined as the expected type as head (resp. tail) for the relation  $r$ . For example, the relation `presidentOf` expects a `Person` as head and a `Country` as tail. Starting from a positive triple (`EmmanuelMacron`, `presidentOf`, `France`) which represents a true fact, (`BarackObama`, `presidentOf`, `France`) and (`EmmanuelMacron`, `presidentOf`, `Germany`) are examples of semantically valid negative triples, whereas (`Adidas`, `presidentOf`, `France`) and (`EmmanuelMacron`, `presidentOf`, `Christmas`) are examples of semantically invalid negative triples. In this work, entities are multi-typed. Therefore,  $\text{type}(e)$  returns the set of types (a.k.a. classes) that the entity  $e$  belongs to.

We introduce a loss-independent  $\epsilon$  factor, which is dubbed as the *semantic factor* and aims at bringing the scores of semantically valid negative triples closer to the positive ones. This common semantic factor fitting into different loss functions shows the generality of our approach that can possibly be extended to other loss functions. Interestingly, this factor also allows to take into account to some extent the Open World Assumption (OWA) under which KGs are represented. Under the OWA, triples that are not represented in a KG are either false or missing positive triples. In traditional training procedures, these triples are indiscriminately considered negative, which corresponds to the Closed World Assumption. On the contrary, our proposal considers semantically invalid triples as true negative while semantically valid triples (and possibly missing positive or false negative under the OWA) are closer to true positive triples. This assumes that entity types are complete and correct.

$\mathcal{L}_{\text{PHL}}$  defined in Equation (1) relies on the margin hyperparameter  $\gamma$ . Increasing (resp. decreasing) the value of  $\gamma$  will increase (resp. decrease) the margin that will be set between the scores of positive and negative triples. However, this unique  $\gamma$  treats all negative triples indifferently: the same margin will separate the scores of semantically valid and semantically invalid negative triples from the score of the positive triple they both originate from. We suggest that

the scores of these two kinds of negative triples should be treated differently. Hence, our approach redefines  $\mathcal{L}_{PHL}$  as follows (Equation (4)):

$$\mathcal{L}_{PHL}^S = \sum_{t \in \mathcal{T}^+} \sum_{t' \in \mathcal{T}^-} [\gamma \cdot \ell(t') + f(t') - f(t)]_+ \quad (4)$$

where  $\ell(t') = \begin{cases} 1 & \text{if } t' \text{ is semantically invalid} \\ \epsilon & \text{otherwise} \end{cases}$

The loss function in Equation (4) now has a superscripted  $S$  to make it clear this is the signature-driven version of the vanilla  $\mathcal{L}_{PHL}$  as defined in Equation (1). A choice of  $\epsilon < 1$  leads the KGEM to apply a higher margin between scores of positive and semantically invalid triples than between positive and semantically valid ones. For a given positive triple, this allows to keep the scores of its semantically valid negative counterparts relatively closer compared to the scores of its semantically invalid counterparts. Intuitively, when the KGEM outputs wrong predictions, more of them are still expected to meet the domain and range constraints imposed by relations. Thus, wrong predictions are assumed to be more meaningful, and, in a sense, semantically closer to the ground-truth triple.

$\mathcal{L}_{BCEL}$  defined in Equation (2) is adapted to  $\mathcal{L}_{BCEL}^S$  by redefining the labelling function  $\ell$ . In particular, when dealing with a KG featuring typed entities and providing information about domains and ranges of relations, the labelling function  $\ell$  is no longer binary. Instead, the labels of semantically valid negative triples can be fixed to some intermediate value between the label value of positive triples and of semantically invalid negative triples, which leads to the labelling function  $\ell$  defined in Equation (5):

$$\ell(t') = \begin{cases} 1 & \text{if } t' \in \mathcal{T}^+ \\ \epsilon & \text{if } t' \in \mathcal{T}^- \text{ and } t' \text{ is semantically valid} \\ 0 & \text{if } t' \in \mathcal{T}^- \text{ and } t' \text{ is semantically invalid} \end{cases} \quad (5)$$

where the semantic factor  $\epsilon$  is a tunable hyperparameter denoting the label value of semantically valid negative triples. The intuition underlying the refinement of the labelling function  $\ell$  is to voluntarily cause some confusion between semantically valid negative triples and positive triples. By bridging their respective label values, it is expected that the KGEM will somehow consider the former as “less negative triples” and assign them a higher score compared to positive triples.

$\mathcal{L}_{PLL}$  defined in Equation (3) could be adapted to  $\mathcal{L}_{PLL}^S$  similarly to  $\mathcal{L}_{BCEL}^S$ . In other words, the labelling function  $\ell$  could also output an intermediate label value  $\epsilon$  for semantically valid negative triples. Although this approach provides very good results in terms of Sem@K values, it does not provide consistently good results across datasets. Furthermore, obtained results in terms of MRR and Hits@K can be far below the ones obtained with the vanilla model (see supplementary materials for further details). That is why, here, to treat semantically valid and invalid negative triples differently, instead of modifying the labelling



function  $\ell$ , the semantic factor  $\epsilon$  for  $\mathcal{L}_{PLL}^S$  defines the probability with which semantically valid negative triples are considered as positive triples and therefore are labelled the same way. For example, with  $\epsilon = 0.05$ , at each training epoch and for each batch, the semantically valid negative triples of the given training batch will be considered positive with a probability of 5%.

It is noteworthy that our approach can be applied in practice to KGs with or without types, domains and ranges. Indeed, in the absence of such background knowledge, our signature-driven loss functions reduce to their respective vanilla counterparts. Recall that our approach does not focus on negative sampling or complex negative sample generators such as KBGAN [4], NSCaching [34], and self-adversarial negative sampling [26]. Although these works are related to ours, they constrain negative sampling upstream. On the contrary, our approach does not constrain the sampling of negative triples but rather dynamically distributes the negative triples into different parts of the loss functions, based on their semantic validity.

## 4 Experimental Setting

### 4.1 Evaluation Metrics

In our experiments, KGEM performance is assessed w.r.t. MRR, Hits@ $K$  and Sem@ $K$ , with  $K = 10$ . **Mean Reciprocal Rank (MRR)** corresponds to the arithmetic mean over the reciprocals of ranks of the ground-truth triples. MRR is bounded in the  $[0, 1]$  interval, where the higher the better. **Hits@ $K$**  accounts for the proportion of ground-truth triples appearing in the first  $K$  top-scored triples. This metric is bounded in the  $[0, 1]$  interval and its values increases with  $K$ , where the higher the better. **Sem@ $K$** [11,13] accounts for the proportion of triples that are semantically valid in the first  $K$  top-scored triples:

$$\text{Sem@}K = \frac{1}{|\mathcal{B}|} \sum_{q \in \mathcal{B}} \frac{1}{K} \sum_{q' \in \mathcal{S}_q^K} \text{compatibility}(q, q') \quad (6)$$

where, given a ground-truth triple  $q = (h, r, t)$ ,  $\mathcal{S}_q^K$  is the list of the top- $K$  candidate triples scored by a KGEM (*i.e.*, by predicting the tail for  $(h, r, ?)$  or the head for  $(?, r, t)$ ). A candidate triple  $q'$  is assessed w.r.t.  $q$  by the compatibility function that checks whether the predicted head (resp. tail) belongs to the domain (resp. range) of the relation. In this work, class hierarchies are considered: if a relation has a given class as domain (resp. range), entities from its subclasses are considered semantically valid. Sem@ $K$  is bounded in the  $[0, 1]$  interval.

### 4.2 Datasets and Models

Even though our approach could be applied in KGs with or without relation signatures, we evaluate our proposal in an ideal experimental setting to precisely qualify the interest of considering domains and ranges. Firstly, all relations appearing in the training set have a defined domain and range. Secondly, both the



head  $h$  and tail  $t$  of train triples have another semantically valid counterpart for negative sampling. These two conditions guarantee that each positive train triple can be paired with at least one semantically valid triple. Finally, validation and test sets contain triples whose relation have a well-defined domain (resp. range), as well as more than 10 semantically valid candidates as head (resp. tail). This ensures Sem@ $K$  is not unduly penalized and can be calculated on the same set of entities as Hits@ $K$  and MRR until  $K = 10$ .

To ensure these requirements, we filtered FB15k237-ET, DBpedia93k, and YAGO4-19k [13] so that they comply with the aforementioned criteria. In the following, their filtered versions are referred to as FB15k187, DBpedia77k, and Yago14k, respectively. Table 1 provides statistics for these datasets and reflects the diversity of their characteristics. In this work, several KGEMs are considered: TransE [3], TransH [30], and DistMult [33] using  $\mathcal{L}_{PHL}$ ; ComplEx [27] and SimpleE [16] using  $\mathcal{L}_{PLL}$ ; ConvE [8], TuckER [2], and RGCN [25] using  $\mathcal{L}_{BCEL}$ . Datasets and codes are available in our GitHub repository.<sup>4</sup>

**Table 1.** Datasets used in the experiments. These are filtered versions of the standard FB15k237-ET, DBpedia93k, and YAGO4-19k.

| Dataset    | $ \mathcal{E} $ | $ \mathcal{R} $ | $ \mathcal{T}_{train} $ | $ \mathcal{T}_{valid} $ | $ \mathcal{T}_{test} $ | Split ratios  |
|------------|-----------------|-----------------|-------------------------|-------------------------|------------------------|---------------|
| FB15k187   | 14,305          | 187             | 245,350                 | 15,256                  | 17,830                 | 88%/5.5%/6.5% |
| DBpedia77k | 76,651          | 150             | 140,760                 | 16,334                  | 32,934                 | 74%/9%/17%    |
| Yago14k    | 14,178          | 37              | 18,263                  | 472                     | 448                    | 95%/2.5%/2.5% |

### 4.3 Implementation Details

For the sake of comparison, MRR, Hits@ $K$  and Sem@ $K$  are computed after training models from scratch. KGEMs used in the experiments were implemented in PyTorch. After training KGEMs for a large number of epochs, we noticed the best achieved results were found around epoch 400 or below. Consequently, a maximum of 400 epochs of training was set, as in LibKGE<sup>5</sup> (except RGCN which is trained during 4,000 epochs due to lower convergence to the best achieved results). Except when using the BCEL which does not require negative sampling, Uniform Random Negative Sampling [3] was used to pair each train triple with two corresponding negative triples: one which is semantically invalid, and one which is semantically valid. Regarding BCEL, each positive triple is scored against negative triples formed with all other entities in the graph. Hence, negative triples comprise both semantically valid and invalid triples. It should be noted that the best epsilon values for each combination of model and dataset were found on the validation sets. Once the best epsilon values are found, they

<sup>4</sup> <https://github.com/nicolas-hbt/semantic-lossfunc/>

<sup>5</sup> <https://github.com/uma-pi1/kge/>

remain fixed for all triples. In order to ensure fair comparisons between models, embeddings are initialized with the same seed and each model is fed with exactly the same set of negative triples at each epoch. Grid-search based on predefined hyperparameters was performed. Best hyperparameters and the full hyperparameter space are reported on GitHub.<sup>4</sup>

## 5 Results

### 5.1 Global Performance

**Table 2.** Rank-based and semantic-based results. Bold fonts indicate which model performs best w.r.t. a given metric. Suffixes V and S indicate whether the model is trained under the vanilla or signature-driven version of the loss function, respectively. Hits@10 and Sem@10 are abbreviated to H@10 and S@10. Underlined cells indicate results that are more specifically referred to in Section 5.1. We use the symbol  $\dagger$  when the comparison of results in a duel (*e.g.* TransE-V vs. TransE-S) is statistically significant ( $\alpha = .05$ ) for a given metric and dataset.

|            | FB15k187                        |                                 |                                 | DBpedia77k                      |                                 |                                 | Yago14k                         |                                 |                                 |
|------------|---------------------------------|---------------------------------|---------------------------------|---------------------------------|---------------------------------|---------------------------------|---------------------------------|---------------------------------|---------------------------------|
|            | MRR                             | H@10                            | S@10                            | MRR                             | H@10                            | S@10                            | MRR                             | H@10                            | S@10                            |
| TransE-V   | .260                            | .446                            | .842                            | .274                            | .438                            | .936                            | .868                            | <b>.945</b>                     | .795                            |
| TransE-S   | <b>.315<math>\dagger</math></b> | <b>.497<math>\dagger</math></b> | <b>.973<math>\dagger</math></b> | <b>.275</b>                     | <b>.440</b>                     | <b>.985<math>\dagger</math></b> | <b>.876</b>                     | .944                            | <b>.968<math>\dagger</math></b> |
| TransH-V   | .266                            | .450                            | .855                            | .270                            | .437                            | .907                            | .836                            | .944                            | .581                            |
| TransH-S   | <b>.319<math>\dagger</math></b> | <b>.501<math>\dagger</math></b> | <b>.973<math>\dagger</math></b> | <b>.274<math>\dagger</math></b> | <b>.442<math>\dagger</math></b> | <b>.980<math>\dagger</math></b> | <b>.857<math>\dagger</math></b> | <b>.945</b>                     | <b>.831<math>\dagger</math></b> |
| DistMult-V | .291                            | .457                            | .824                            | .295                            | .405                            | .784                            | .904                            | <b>.930</b>                     | .409                            |
| DistMult-S | <b>.332<math>\dagger</math></b> | <b>.504<math>\dagger</math></b> | <b>.971<math>\dagger</math></b> | <b>.300<math>\dagger</math></b> | <b>.416<math>\dagger</math></b> | <b>.901<math>\dagger</math></b> | <b>.912<math>\dagger</math></b> | .929                            | <b>.449<math>\dagger</math></b> |
| ComplEx-V  | .280                            | .416                            | .472                            | <b>.309<math>\dagger</math></b> | <b>.415<math>\dagger</math></b> | .769                            | <b>.925</b>                     | <b>.932</b>                     | .333                            |
| ComplEx-S  | <b>.316<math>\dagger</math></b> | <b>.476<math>\dagger</math></b> | <b>.796<math>\dagger</math></b> | .297                            | .409                            | <b>.897</b>                     | .923                            | .931                            | <b>.667<math>\dagger</math></b> |
| Simple-V   | .261                            | .387                            | .462                            | <b>.259<math>\dagger</math></b> | <b>.346<math>\dagger</math></b> | <b>.883<math>\dagger</math></b> | <b>.926</b>                     | <b>.931</b>                     | .355                            |
| Simple-S   | <b>.268<math>\dagger</math></b> | <b>.409<math>\dagger</math></b> | <b>.759<math>\dagger</math></b> | .230                            | .302                            | .850                            | .924                            | .927                            | <b>.769<math>\dagger</math></b> |
| ConvE-V    | .273                            | .470                            | .973                            | .273                            | .382                            | .935                            | <b>.934</b>                     | <b>.942</b>                     | .904                            |
| ConvE-S    | <b>.283<math>\dagger</math></b> | <b>.476<math>\dagger</math></b> | <b>.996<math>\dagger</math></b> | <b>.283<math>\dagger</math></b> | <b>.405<math>\dagger</math></b> | <b>.985<math>\dagger</math></b> | .933                            | .940                            | <b>.997<math>\dagger</math></b> |
| TuckER-V   | .316                            | .516                            | .985                            | .311                            | .410                            | .912                            | .923                            | .927                            | .781                            |
| TuckER-S   | <b>.320<math>\dagger</math></b> | <b>.522<math>\dagger</math></b> | <b>.996<math>\dagger</math></b> | <b>.312</b>                     | <b>.421<math>\dagger</math></b> | <b>.969<math>\dagger</math></b> | <b>.931<math>\dagger</math></b> | <b>.943<math>\dagger</math></b> | <b>.929<math>\dagger</math></b> |
| RGCN-V     | .241                            | .386                            | .775                            | .194                            | .297                            | .872                            | .911                            | .923                            | .349                            |
| RGCN-S     | <b>.260<math>\dagger</math></b> | <b>.415<math>\dagger</math></b> | <b>.860<math>\dagger</math></b> | <b>.197<math>\dagger</math></b> | <b>.320<math>\dagger</math></b> | <b>.957<math>\dagger</math></b> | <b>.927<math>\dagger</math></b> | <b>.934<math>\dagger</math></b> | <b>.828<math>\dagger</math></b> |

Table 2 displays KGEM performance, datasets and evaluation metrics of interest. We performed t-tests (when prediction-related data follow a normal distribution according to Shapiro test) and Wilcoxon tests (when they do not) at the significance level  $\alpha = .05$ . Table 2 clearly shows that, with the sole exception of Simple on DBpedia77k, the signature-driven loss functions  $\mathcal{L}_{PHL}^S$ ,  $\mathcal{L}_{BCEL}^S$ , and  $\mathcal{L}_{PLL}^S$  all lead to significant improvement in terms of Sem@10.

Importantly, in some cases the relative gain in Sem@10 compared to the corresponding vanilla loss function is huge (underlined in Table 2): +137%, +117%, and +100% for RGCN, SimpleE, and ComplEx on the smaller Yago14k dataset, respectively, and +69% and +64% for ComplEx and SimpleE on FB15k187, respectively. These gains in terms of Sem@10 are observed regardless of the loss function at hand, which demonstrates that our designed signature-driven loss functions can all drive KGEM semantic correctness towards more satisfying results. It is worth noting that, in most cases, they also lead to better KGEM performance as measured by MRR and Hits@10. We observe that in 19 out of 24 ( $\approx 79\%$ ) one-to-one comparisons between the same KGEM trained with vanilla vs. signature-driven loss functions, better MRR values are reported for the model trained under the signature-driven loss function. On the remaining comparisons ( $\approx 21\%$ ), only two highlight statistically significant losses in terms of MRR. However, they are minimal and often for the benefit of significantly better Sem@10 values. This observation raises the question whether better MRR and Hits@ $K$  should be pursued at any cost, or whether a small drop w.r.t. to these metrics is acceptable if this leads to a significantly better KGEM semantic correctness (see Section 6.3). Plus, these promising results imply that even if the intended purpose is to only maximize MRR and Hits@ $K$  values, taking the available signature information into consideration is strongly advised: this does not only improve KGEM semantic correctness, but also provide performance gains in terms of MRR and Hits@ $K$ .

In the following, we provide a finer-grained results’ analysis, which focuses on the different loss functions and datasets used in the experiments. Although we previously showed the effectiveness of our approach, the benefits brought from considering BK in the form of relation domains and ranges differ across loss functions. In particular, the gains achieved using  $\mathcal{L}_{PHL}^S$  and  $\mathcal{L}_{BCEL}^S$  are substantial. With these loss functions, we observe a systematic improvement w.r.t. Sem@10. Gains are also reported w.r.t. rank-based metrics in the vast majority of cases. The only exception is on Yago14k where semantic losses are sometimes slightly outperformed but still competitive w.r.t. their vanilla counterparts. However, the difference in terms of MRR and Hits@10 values is negligible and not statistically significant. This may come from the reduced number of triples in Yago14k, which results in the additional signature information improving Sem@ $K$  but not helping discriminate gold entities from others. Therefore, incorporating BK about relation domains and ranges into  $\mathcal{L}_{PHL}^S$  and  $\mathcal{L}_{BCEL}^S$  is a viable approach that provides consistent gains both in terms of MRR, Hits@10 and Sem@10. Regarding the benefits from doing so under  $\mathcal{L}_{PLL}^S$ , we can notice a slight decline in MRR and Hits@10 values in some cases. However, the other side of the coin is that achieved Sem@10 values are substantially higher: except for SimpleE on DBpedia77k, Sem@10 values increase in a range from +17% to +117% for the remaining one-to-one comparisons. As such, our approach using  $\mathcal{L}_{PLL}^S$  also provides satisfactory results, as long as a slight drop in rank-based metrics is acceptable if it comes with the benefit of significantly better KGEM semantic correctness. Besides, it is worth noting the following points: our hyperparam-

eter tuning strategy relied on the choice of the best  $\epsilon$  value on Yago14k – for computational limitations. The  $\epsilon$  value which was found to perform the best on Yago14k was subsequently used in all the remaining scenarios. A more thorough tuning of  $\epsilon$  on the other datasets would have potentially provided even more satisfying results under the  $\mathcal{L}_{PLL}^S$ , thus strengthening the value of our approach.

## 5.2 Ablation Study

In this section, KGEMs are tested on three buckets of relations that feature narrow (B1), intermediate (B2), and large (B3) sets of semantically valid heads or tails, respectively. The cut-offs have been manually defined and are provided in supplementary materials<sup>4</sup>. The analysis of B1 allows us to gauge the impact of signature-driven loss functions on relations for which it is harder to predict semantically valid entities. Results reported in Table 3 for B1 clearly demonstrate that the impact of injecting BK into loss functions is exacerbated for them, thus supporting the value of our approach in a sparse and difficult setting. One might think that the better MRR values achieved with  $\mathcal{L}_{PHL}^S$ ,  $\mathcal{L}_{BCEL}^S$ , and  $\mathcal{L}_{PLL}^S$  are highly correlated to the better Sem@10 values. This is partially true, as for a relation in B1, placing all semantically valid candidates at the top of the ranking list is likely to uplift the rank of the ground-truth itself. However, we can see that RGCN-V and RGCN-S have almost equal MRR values on Yago14k, while Sem@10 values of RGCN-S are much higher than RGCN-V (+254%). Similar findings hold for ComplEx on Yago14k, ConvE on DBpedia77k, TransE on DBpedia77k and Yago14k. This shows that in a number of cases, signature-driven loss functions improve the semantic correctness of KGEM for small relations while leaving its performance untouched in terms of rank-based metrics. Results on B2 and B3 are provided in supplementary materials<sup>4</sup>. In particular, it can be noted that the relative benefit of our approach w.r.t. MRR and Hits@10 is more limited on such buckets. Regarding Sem@K, results achieved with the vanilla loss functions are already high, hence the relatively lower gain brought by injecting ontological BK. These already high Sem@K values may be explained by a higher number of semantically valid candidates.

## 6 Discussion

### 6.1 Treating Different Negatives Differently (RQ1)

The proposed loss functions  $\mathcal{L}_{PHL}^S$ ,  $\mathcal{L}_{BCEL}^S$ , and  $\mathcal{L}_{PLL}^S$  provide adequate training objective for KGEMs, as evidenced in Table 2. Most importantly, in Section 3 we clearly show how the inclusion of signature information into  $\mathcal{L}_{PHL}^S$ ,  $\mathcal{L}_{BCEL}^S$ , and  $\mathcal{L}_{PLL}^S$  can be brought under one roof thanks to a commonly defined semantic factor. Although this semantic factor operates at different levels depending on the loss function, its common purpose is to differentiate how semantically valid and semantically invalid negative triples should be considered compared to positive triples, whereas traditional approaches treat all negative triples indifferently.

**Table 3.** Rank-based and semantic-based results on the bucket of relations that feature a narrow set of semantically valid heads or tails (B1). The cut-offs have been manually defined and are provided in supplementary materials.

|            | FB15k187    |             |             | DBpedia77k  |             |             | Yago14k     |             |             |
|------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
|            | MRR         | H@10        | S@10        | MRR         | H@10        | S@10        | MRR         | H@10        | S@10        |
| TransE-V   | .535        | .727        | .647        | .498        | .578        | .460        | .914        | <b>.979</b> | .620        |
| TransE-S   | <b>.646</b> | <b>.805</b> | <b>.937</b> | <b>.539</b> | <b>.626</b> | <b>.910</b> | <b>.922</b> | .975        | <b>.924</b> |
| TransH-V   | .541        | .734        | .661        | .475        | .555        | .425        | .897        | .972        | .436        |
| TransH-S   | <b>.655</b> | <b>.814</b> | <b>.936</b> | <b>.541</b> | <b>.643</b> | <b>.828</b> | <b>.923</b> | <b>.981</b> | <b>.684</b> |
| DistMult-V | .589        | .735        | .628        | .466        | .462        | .244        | .949        | <b>.959</b> | .304        |
| DistMult-S | <b>.667</b> | <b>.802</b> | <b>.929</b> | <b>.498</b> | <b>.547</b> | <b>.451</b> | <b>.965</b> | .958        | <b>.372</b> |
| ComplEx-V  | .530        | .567        | .116        | <b>.424</b> | <b>.425</b> | .161        | .955        | .956        | .133        |
| ComplEx-S  | <b>.637</b> | <b>.723</b> | <b>.537</b> | .421        | .399        | <b>.198</b> | <b>.961</b> | .956        | <b>.423</b> |
| SimplE-V   | .507        | .553        | .136        | <b>.396</b> | <b>.370</b> | <b>.273</b> | <b>.959</b> | .882        | <b>.932</b> |
| SimplE-S   | <b>.576</b> | <b>.671</b> | <b>.505</b> | .324        | .259        | .206        | .958        | <b>.883</b> | .930        |
| ConvE-V    | .549        | .779        | .973        | .518        | <b>.569</b> | .789        | .969        | <b>.972</b> | .915        |
| ConvE-S    | <b>.562</b> | <b>.783</b> | <b>.986</b> | .518        | .566        | <b>.927</b> | .969        | .965        | <b>.960</b> |
| TuckER-V   | .597        | .811        | .969        | .519        | .568        | .740        | .949        | .970        | .846        |
| TuckER-S   | <b>.598</b> | <b>.815</b> | <b>.973</b> | <b>.526</b> | <b>.582</b> | <b>.797</b> | <b>.964</b> | .970        | <b>.892</b> |
| RGCN-V     | .510        | .629        | .468        | .386        | .387        | .254        | .963        | .959        | .141        |
| RGCN-S     | <b>.549</b> | <b>.705</b> | <b>.682</b> | <b>.396</b> | <b>.415</b> | <b>.398</b> | <b>.966</b> | <b>.967</b> | <b>.499</b> |

Besides, our approach that transforms vanilla loss functions into signature-driven ones can also work for other loss functions such as the pointwise hinge loss or the pairwise logistic loss, as presented in [22], by including a semantic factor  $\epsilon$  as well. The tailoring of these losses and the experiments are left for future work.

Recall that compared to complex negative sampler such as KBGAN [4], NSCaching [34], and self-adversarial NS [26], we do not introduce any potential overhead due to the need for maintaining a cache [34] or training an intermediate adversarial learning framework [4,26] for generating high-quality negatives. Instead, negative triples dynamically enter a different part of the loss function depending on their semantic validity. In future work, we will compare the performance and algorithmic complexity of our approach w.r.t. NS procedures, thereby highlighting the potential cost savings in computational resources and execution time compared to sophisticated NS. Our approach is also agnostic to the underlying NS procedure, and can work along with simple uniform random NS [3] as well as more complex procedures [4,34,26]. Besides, our approach can be applied even in the absence of BK. In this case, signature-driven loss functions reduce to their vanilla version.

## 6.2 Impact of Signature-Driven Losses on Performance (RQ2)

Mohamed *et al.* [21] investigate the effects of specific choices of loss functions on the scalability and performance of KGEMs w.r.t. rank-based metrics. In this present work, we also assess the semantic capabilities of such models. Based on

the results provided and analyzed in Section 3, incorporating BK about relation domains and ranges into the loss functions clearly contribute to a better KGEM semantic correctness, as evidenced by the huge increase frequently observed w.r.t.  $\text{Sem}@K$ . Although it seems intuitive that our proposed approach will result in better predicted semantics, it should not be taken for granted. Hubert *et al.* demonstrate that in the special case of LP on a single target relation (*e.g.*, recommender systems), incorporating schema-based information during training (in their case, during negative sampling) actually decreases the semantic correctness of KGEMs [12]. This result leads us to posit that injecting ontological information during training does not necessarily lead to a better semantic correctness. The improvement might depend on (1) where this information is consumed (*e.g.*, in the loss function, during negative sampling) and (2) what the task at hand is.

It should also be noted that this increase is not homogeneously distributed across relations: relations with a smaller set of semantically valid entities as heads or tails are more challenging w.r.t.  $\text{Sem}@K$  (see Table 3 and results on B2 and B3 in supplementary materials<sup>4</sup>). For such relations, the relative gain from signature-driven loss functions is more acute. In addition, signature-driven loss functions also drive most of the KGEMs towards better MRR and  $\text{Hits}@K$  values. As shown in Table 2, this is particularly the case when using the  $\mathcal{L}_{PHL}^S$  and  $\mathcal{L}_{BCEL}^S$ . This result is particularly interesting, as it suggests that when semantic information about entities and relations is available, there are benefits in using it, even if the intended goal remains to enhance KGEM performance w.r.t. rank-based metrics only.

In addition, it has been noted that including signature information in loss functions has the highest impact on small relations (B1) (Table 3), both in terms of rank-based metrics (MRR,  $\text{Hits}@10$ ) and semantic correctness ( $\text{Sem}@10$ ). For relations with a larger pool of semantically valid entities, the impact is still positive w.r.t.  $\text{Sem}@10$ , but sometimes at the expense of a small drop in terms of rank-based metrics. If the latter metrics are the sole optimization objective, it would be reasonable to design an adaptive training strategy in which vanilla and signature-driven loss functions are alternatively used depending on the current relation and the number of semantically valid candidates as head or tail.

### 6.3 On the Evaluation of KGEM Predictions for LP

It should be noted that  $\text{Sem}@K$  measures the capability of models to predict semantically correct triples w.r.t. relation signatures and but does not measure falsehood, contrary to MRR or  $\text{Hits}@K$ . It can be argued that in specific use-cases, such as e-commerce recommender systems, models should predict the expected item (high MRR/ $\text{Hits}@K$ ) but also predict semantically valid items (high  $\text{Sem}@K$ ) for a good user experience. In other applications such as healthcare, aerospace, or security, users expect to notice when models are failing in order to be able to take over. This corresponds to high MRR/ $\text{Hits}@K$  and low  $\text{Sem}@K$  so that errors are clearly noticeable.

Our experiments and these use cases illustrate the need of both  $\text{Hits}@K$  and  $\text{Sem}@K$  metrics to precisely qualify model performance w.r.t. the needs of the

considered application (*e.g.*, e-commerce, healthcare). This opens perspectives on sampling and differently considering different kinds of negative triples to tailor specific aspects of model performance to the requirements of the application. For instance, based on the results of our experiments, we could envision differently considering negatives for an e-commerce application to obtain a high  $\text{Sem}@K$  while an aerospace application would require the contrary.

## 7 Conclusion

In this work, we focus on the main loss functions used for link prediction in knowledge graphs. Building on the assumption that negative triples are not all equally good for learning better embeddings, we propose to differentiate them based on their semantic validity w.r.t. the domain and range of relations by including relation signature information into loss functions. A wide range of KGEMs are subsequently trained under both the vanilla and signature-driven loss functions. In our experiments on three public KGs with different characteristics, the proposed signature-driven loss functions lead to promising results: in most cases, they do not only lead to better MRR and Hits@10 values, but also drive KGEMs towards better semantic correctness as measured with  $\text{Sem}@10$ . This advocates for the further injection of semantic information into loss functions whenever such information is available. In future work, we will study how the proposed loss functions can accommodate other types of ontological constraints as well as literal nodes.

## References

1. Ali, M., Berrendorf, M., Hoyt, C.T., Vermue, L., Galkin, M., Sharifzadeh, S., Fischer, A., Tresp, V., Lehmann, J.: Bringing light into the dark: A large-scale evaluation of knowledge graph embedding models under a unified framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **44**(12), 8825–8845 (2022). <https://doi.org/10.1109/TPAMI.2021.3124805>
2. Balazevic, I., Allen, C., Hospedales, T.M.: Tucker: Tensor factorization for knowledge graph completion. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*. pp. 5184–5193. Association for Computational Linguistics (2019). <https://doi.org/10.18653/v1/D19-1522>
3. Bordes, A., Usunier, N., García-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: *Conference on Neural Information Processing Systems (NeurIPS)*. pp. 2787–2795 (2013)
4. Cai, L., Wang, W.Y.: KBGAN: adversarial learning for knowledge graph embeddings. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*. pp. 1470–1480. Association for Computational Linguistics (2018). <https://doi.org/10.18653/v1/n18-1133>



5. Cao, Z., Xu, Q., Yang, Z., Huang, Q.: ER: equivariance regularizer for knowledge graph completion. In: Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022. pp. 5512–5520. AAAI Press (2022)
6. Cui, Z., Kapanipathi, P., Talamadupula, K., Gao, T., Ji, Q.: Type-augmented relation prediction in knowledge graphs. In: Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021. pp. 7151–7159. AAAI Press (2021)
7. d’Amato, C., Quatraro, N.F., Fanizzi, N.: Injecting background knowledge into embedding models for predictive tasks on knowledge graphs. In: The Semantic Web - 18th International Conference, ESWC 2021, Virtual Event, June 6-10, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12731, pp. 441–457. Springer (2021). [https://doi.org/10.1007/978-3-030-77385-4\\_26](https://doi.org/10.1007/978-3-030-77385-4_26)
8. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2d knowledge graph embeddings. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018. pp. 1811–1818. AAAI Press (2018)
9. Ding, B., Wang, Q., Wang, B., Guo, L.: Improving knowledge graph embedding using simple constraints. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers. pp. 110–121. Association for Computational Linguistics (2018). <https://doi.org/10.18653/v1/P18-1011>
10. Guo, S., Wang, Q., Wang, B., Wang, L., Guo, L.: Semantically smooth knowledge graph embedding. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers. pp. 84–94. The Association for Computer Linguistics (2015). <https://doi.org/10.3115/v1/p15-1009>
11. Hubert, N., Monnin, P., Brun, A., Monticolo, D.: Knowledge Graph Embeddings for Link Prediction: Beware of Semantics! In: Proceedings of the Workshop on Deep Learning for Knowledge Graphs (DL4KG 2022) co-located with the 21th International Semantic Web Conference (ISWC 2022). Virtual Conference, online (Oct 2022)
12. Hubert, N., Monnin, P., Brun, A., Monticolo, D.: New strategies for learning knowledge graph embeddings: The recommendation case. In: Knowledge Engineering and Knowledge Management - 23rd International Conference, EKAW 2022, Bolzano, Italy, September 26-29, 2022, Proceedings. Lecture Notes in Computer Science, vol. 13514, pp. 66–80. Springer (2022). [https://doi.org/10.1007/978-3-031-17105-5\\_5](https://doi.org/10.1007/978-3-031-17105-5_5)
13. Hubert, N., Monnin, P., Brun, A., Monticolo, D.: Sem@k: Is my knowledge graph embedding model semantic-aware? (2023)
14. Jain, N., Tran, T., Gad-Elrab, M.H., Stepanova, D.: Improving knowledge graph embeddings with ontological reasoning. In: The Semantic Web - ISWC 2021 - 20th

- International Semantic Web Conference, ISWC 2021, Virtual Event, October 24–28, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12922, pp. 410–426. Springer (2021). [https://doi.org/10.1007/978-3-030-88361-4\\_24](https://doi.org/10.1007/978-3-030-88361-4_24)
15. Ji, S., Pan, S., Cambria, E., Marttinen, P., Yu, P.S.: A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Networks Learn. Syst.* **33**(2), 494–514 (2022). <https://doi.org/10.1109/TNNLS.2021.3070843>
  16. Kazemi, S.M., Poole, D.: Simple embedding for link prediction in knowledge graphs. In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3–8, 2018, Montréal, Canada*. pp. 4289–4300 (2018)
  17. Kotnis, B., Nastase, V.: Analysis of the impact of negative sampling on link prediction in knowledge graphs. *arXiv preprint 1708.06816* (2017)
  18. Krompaß, D., Baier, S., Tresp, V.: Type-constrained representation learning in knowledge graphs. In: *The Semantic Web - 14th International Semantic Web Conference (ISWC)*. vol. 9366, pp. 640–655. Springer (2015)
  19. Lv, X., Hou, L., Li, J., Liu, Z.: Differentiating concepts and instances for knowledge graph embedding. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. pp. 1971–1979. Association for Computational Linguistics (2018). <https://doi.org/10.18653/v1/d18-1222>
  20. Minervini, P., Costabello, L., Muñoz, E., Nováček, V., Vandenbussche, P.: Regularizing knowledge graph embeddings via equivalence and inversion axioms. In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18–22, 2017, Proceedings, Part I*. Lecture Notes in Computer Science, vol. 10534, pp. 668–683. Springer (2017). [https://doi.org/10.1007/978-3-319-71249-9\\_40](https://doi.org/10.1007/978-3-319-71249-9_40)
  21. Mohamed, S.K., Muñoz, E., Nováček, V.: On training knowledge graph embedding models. *Information* **12**(4) (2021). <https://doi.org/10.3390/info12040147>
  22. Mohamed, S.K., Nováček, V., Vandenbussche, P., Muñoz, E.: Loss functions in knowledge graph embedding models. In: *Proceedings of the Workshop on Deep Learning for Knowledge Graphs (DL4KG2019) Co-located with the 16th Extended Semantic Web Conference 2019 (ESWC 2019), Portoroz, Slovenia, June 2, 2019*. *CEUR Workshop Proceedings*, vol. 2377, pp. 1–10. CEUR-WS.org (2019)
  23. Niu, G., Li, B., Zhang, Y., Pu, S., Li, J.: Autoeter: Automated entity type representation with relation-aware attention for knowledge graph embedding. In: *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16–20 November 2020*. *Findings of ACL*, vol. EMNLP 2020, pp. 1172–1181. Association for Computational Linguistics (2020). <https://doi.org/10.18653/v1/2020.findings-emnlp.105>
  24. Rossi, A., Barbosa, D., Firmani, D., Matinata, A., Merialdo, P.: Knowledge graph embedding for link prediction: A comparative analysis. *ACM Transactions on Knowledge Discovery from Data* **15**(2), 14:1–14:49 (2021)
  25. Schlichtkrull, M.S., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings*. Lecture Notes in Computer Science, vol. 10843, pp. 593–607. Springer (2018). [https://doi.org/10.1007/978-3-319-93417-4\\_38](https://doi.org/10.1007/978-3-319-93417-4_38)
  26. Sun, Z., Deng, Z., Nie, J., Tang, J.: Rotate: Knowledge graph embedding by relational rotation in complex space. In: *7th International Conference on Learning Representations, ICLR (2019)*

27. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: *Proceedings of the 33rd International Conference on Machine Learning, ICML*. vol. 48, pp. 2071–2080 (2016)
28. Wang, P., Zhou, J., Liu, Y., Zhou, X.: Transet: Knowledge graph embedding with entity types. *Electronics* **10**(12), 1407 (2021)
29. Wang, Q., Mao, Z., Wang, B., Guo, L.: Knowledge graph embedding: A survey of approaches and applications. *IEEE Trans. Knowl. Data Eng.* **29**(12), 2724–2743 (2017)
30. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*. pp. 1112–1119 (2014)
31. Weyns, M., Bonte, P., Steenwinkel, B., Turck, F.D., Ongenaë, F.: Conditional constraints for knowledge graph embeddings. In: *Proc. of the Workshop on Deep Learning for Knowledge Graphs (DL4KG@ISWC)*. vol. 2635 (2020)
32. Xie, R., Liu, Z., Sun, M.: Representation learning of knowledge graphs with hierarchical types. In: *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016*, New York, NY, USA, 9–15 July 2016. pp. 2965–2971. IJCAI/AAAI Press (2016)
33. Yang, B., Yih, W., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. In: *3rd International Conference on Learning Representations, ICLR* (2015)
34. Zhang, Y., Yao, Q., Shao, Y., Chen, L.: Nscaching: Simple and efficient negative sampling for knowledge graph embedding. In: *35th IEEE International Conference on Data Engineering, ICDE 2019*, Macao, China, April 8–11, 2019. pp. 614–625. IEEE (2019). <https://doi.org/10.1109/ICDE.2019.00061>

## A Modified Versions of $\mathcal{L}_{BCEL}^S$ and $\mathcal{L}_{PLL}^S$

In Table 4, we report results achieved with KGEMs trained under  $\mathcal{L}_{BCEL}^{S'}$  and  $\mathcal{L}_{PLL}^{S'}$ , where the superscript  $S'$  denotes a different way to include semantic information. In fact,  $\mathcal{L}_{PLL}^{S'}$  makes use of the modified labelling function as used in  $\mathcal{L}_{BCEL}^S$  and defined in Equation (5) of the paper. Conversely,  $\mathcal{L}_{BCEL}^{S'}$  uses the binary (unmodified) labelling function  $\ell$  but adopts the same procedure as  $\mathcal{L}_{PLL}^S$ : semantically valid negative triples are considered as positive with probability  $\epsilon$  %. Hyperparameters for  $\mathcal{L}_{BCEL}^S$ ,  $\mathcal{L}_{PLL}^S$ ,  $\mathcal{L}_{BCEL}^{S'}$ , and  $\mathcal{L}_{PLL}^{S'}$  are reported in <https://github.com/nicolas-hbt/semantic-lossfunc/>. Results achieved with all the aforementioned loss functions are provided in Table 4. It shows that the signature-driven loss functions presented in the paper are the best performing ones.

**Table 4.** Rank-based and semantic-based results on FB15k187, DBpedia77k, and Yago14k. Bold fonts indicate which model performs best w.r.t. a given metric. Suffixes S and S' indicate whether the model is trained under the best (as presented in the paper) or the worst (as presented here) signature-driven version of the loss function, respectively.

|            | FB15k187    |             |             | DBpedia77k  |             |             | Yago14k     |             |             |
|------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
|            | MRR         | H@10        | S@10        | MRR         | H@10        | S@10        | MRR         | H@10        | S@10        |
| ComplEx-S  | <b>.316</b> | <b>.476</b> | <b>.796</b> | <b>.297</b> | <b>.409</b> | <b>.897</b> | <b>.923</b> | <b>.931</b> | <b>.667</b> |
| ComplEx-S' | .227        | .384        | .777        | .252        | .350        | .918        | .907        | .930        | .603        |
| SimpleE-S  | <b>.268</b> | <b>.409</b> | .759        | .230        | <b>.302</b> | <b>.850</b> | <b>.924</b> | <b>.927</b> | <b>.769</b> |
| SimpleE-S' | .169        | .288        | <b>.827</b> | .230        | .297        | .583        | .885        | .915        | .290        |
| ConvE-S    | <b>.283</b> | <b>.476</b> | <b>.996</b> | <b>.283</b> | <b>.405</b> | <b>.985</b> | .933        | .940        | <b>.997</b> |
| ConvE-S'   | .271        | .472        | .975        | .273        | .383        | .935        | .933        | <b>.941</b> | .894        |
| TuckER-S   | <b>.320</b> | <b>.522</b> | <b>.996</b> | <b>.312</b> | <b>.421</b> | <b>.969</b> | <b>.931</b> | <b>.943</b> | <b>.929</b> |
| TuckER-S'  | .316        | .517        | .983        | .311        | .412        | .912        | .918        | .938        | .867        |
| RGCN-S     | <b>.260</b> | <b>.415</b> | <b>.860</b> | <b>.197</b> | <b>.320</b> | <b>.957</b> | <b>.927</b> | <b>.934</b> | <b>.828</b> |
| RGCN-S'    | .243        | .391        | .780        | .146        | .246        | .862        | .912        | .922        | .385        |

## B Bucket Analysis

Relations are separated into three non-intersecting buckets : relations that feature narrow (B1), intermediate (B2), and large (B3) sets of semantically valid heads or tails, respectively. Cut-offs are manually defined for placing a given relation in its corresponding bucket. Such buckets are reported in <https://github.com/nicolas-hbt/semantic-lossfunc/>. Results achieved on B1 are reported in the paper, while results for buckets B2 and B3 for DBpedia77k, FB15k187, and Yago14k are reported in <https://github.com/nicolas-hbt/semantic-lossfunc/>.