

**SEM@K: IS MY KNOWLEDGE GRAPH EMBEDDING MODEL SEMANTIC-AWARE?**

Nicolas Hubert
Université de Lorraine, CNRS, LORIA, France
nicolas.hubert@univ-lorraine.fr

Pierre Monnin
Université Côte d’Azur, Inria, CNRS, I3S, France
pierre.monnin@inria.fr

Armelle Brun
Université de Lorraine, CNRS, LORIA, France
armelle.brun@loria.fr

Davy Monticolo
Université de Lorraine, ERPI, France
davy.monticolo@univ-lorraine.fr

**ABSTRACT**

Using knowledge graph embedding models (KGEMs) is a popular approach for predicting links in knowledge graphs (KGs). Traditionally, the performance of KGEMs for link prediction is assessed using rank-based metrics, which evaluate their ability to give high scores to ground-truth entities. However, the literature claims that the KGEM evaluation procedure would benefit from adding supplementary dimensions to assess. That is why, in this paper, we extend our previously introduced metric Sem@K that measures the capability of models to predict valid entities w.r.t. domain and range constraints. In particular, we consider a broad range of KGs and take their respective characteristics into account to propose different versions of Sem@K. We also perform an extensive study to qualify the abilities of KGEMs as measured by our metric. Our experiments show that Sem@K provides a new perspective on KGEM quality. Its joint analysis with rank-based metrics offers different conclusions on the predictive power of models. Regarding Sem@K, some KGEMs are inherently better than others, but this semantic superiority is not indicative of their performance w.r.t. rank-based metrics. In this work, we generalize conclusions about the relative performance of KGEMs w.r.t. rank-based and semantic-oriented metrics at the level of families of models. The joint analysis of the aforementioned metrics gives more insight into the peculiarities of each model. This work paves the way for a more comprehensive evaluation of KGEM adequacy for specific downstream tasks.

**Keywords** Knowledge Graph Embeddings · Link Prediction · Model Evaluation · Semantic-Oriented Metrics

1 **Introduction**

A knowledge graph (KG) is commonly seen as a directed multi-relational graph in which two nodes can be linked through potentially several semantic relationships. More formally, a knowledge graph $KG = (\mathcal{E}, \mathcal{R}, \mathcal{T})$ where $\mathcal{E}, \mathcal{R}$ and $\mathcal{T} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ are a set of entities (nodes), relations (edge labels) and triples, respectively. A KG is represented as a collection of such triples - a.k.a. facts - denoted as $(h, r, t) \in \mathcal{T}$ where $h \in \mathcal{E}$ and $t \in \mathcal{E}$ are two entities of the graph and are respectively named the head and tail of the triple, while $r \in \mathcal{R}$ is a predicate that qualifies the nature of the relationship holding between these entities. For instance, in the sample KG depicted in Fig. 1: $\mathcal{E} = \{\text{BarackObama,MichelleObama,EmmanuelMacron,USA,France}\}$ and $\mathcal{R} = \{\text{spouseOf,presidentOf,supports,livesIn}\}$.

KGs are inherently incomplete, incorrect, or overlapping and thus major refinement tasks include entity matching, question answering, and link prediction Paulheim [2017], Wang et al. [2017]. 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
Sem@K: Is my KGEM semantic-aware?

Figure 1: Excerpt of a KG containing some influential political figures and relations holding between them

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 $T'$ where, for each triple, either the head $h$ or the tail $t$ is missing. This task can be subdivided into a head prediction phase - which consists in predicting the most plausible head $h$ for each $(?, r, t)$ -  and a tail prediction phase - which consists in predicting the most plausible tail $t$ for each $(h, r, ?)$. In the sample KG depicted in Fig. 1, an example of triple to be predicted during the tail prediction phase would be $(\text{EmmanuelMacron}, \text{livesIn}, ?)$, where the expected tail to be inferred is France. Training a Knowledge Graph Embedding Model (KGEM) firstly requires corrupting existing triples by replacing either their head $h$ or their tail $t$ with another entity to generate negative counterparts. This procedure is called negative sampling Bordes et al. [2013], Krompaß et al. [2015], Jain et al. [2021a]. Secondly, the KGEM iteratively learns to assign higher scores to true triples than to their negative counterparts.

The performance of KGEM for LP is evaluated using rank-based metrics such as Hits@$K$, Mean Rank (MR), and Mean Reciprocal Rank (MRR) that assess whether ground-truth entities are indeed given higher scores Wang et al. [2017], Rossi et al. [2021]. However, various works recently raised some caveats about such metrics Berrendorf et al. [2020], Hoyt et al. [2022], Tiwari et al. [2021]. For instance, they are not well-suited for drawing comparisons across datasets Berrendorf et al. [2020]. More importantly, they only provide a partial picture of KGEM performance Berrendorf et al. [2020]. Indeed, LP can lead to nonsensical triples, such as $(\text{BarackObama}, \text{isFatherOf}, \text{USA})$, being predicted as highly plausible facts, although they violate constraints on the domain and range of relations Jain et al. [2021a], Wang et al. [2019]. KGEMs with such issues may nevertheless reach a satisfying performance in terms of rank-based metrics.

Few works propose to go beyond the mere traditional quantitative performance of KGEMs and address their ability to capture the semantics of the original KG, e.g., domain and range constraints, hierarchy of classes Paulheim [2018], Jain et al. [2021b], Monnin et al. [2022]. According to Berrendorf et al. Berrendorf et al. [2020], this would give a more complete picture of the performance of a KGEM. This is why we advocate for additional qualitative and semantic-oriented metrics to supplement traditional rank-based metrics and propose Sem@$K$ to address this need. The relevance of using semantic-oriented metrics – more specifically Sem@$K$ – is clearly visible in Fig. 2: Sem@$K$ provides a supplementary dimension to the evaluation procedure and allows to confidently choose between two models. They are equally good in terms of rank-based metrics but Model A predicts entities that are semantically valid w.r.t. the range constraint.

More specifically, in this work, our goal is to assess the ability of popular KGEMs to capture the semantic profile of relations in a LP task, i.e., whether KGEMs predict entities that respects domain and range of relations. Henceforth, we refer to this aspect as the semantic awareness of KGEMs. To do so, we build on Sem@$K$, a semantic-oriented metric that we previously introduced Hubert et al. [2022a,b] In Hubert et al. [2022a], Sem@$K$ was specifically defined for the recommendation task which was seen as predicting tails for a unique target relation. Sem@$K$ was then extended in Hubert et al. [2022b] to the more generic LP task, where not only tails but also heads are corrupted and all relations are considered. In the present work, we deepen the study of the semantic awareness of KGEMs by proposing different versions of Sem@$K$ that take into account the different characteristics of KGs (e.g., hierarchy of types). Moreover, the semantic awareness of a wider range of KGEMs is analyzed – especially convolutional models. Likewise, a broader array of KGs is used, in order to benchmark the semantic capabilities of KGEMs on mainstream LP datasets. Thus, the following research questions are addressed:

- **RQ1**: how semantic-aware agnostic KGEMs are?
- **RQ2**: how the evaluation of KGEM semantic awareness should adapt to the typology of KGs?
Accordingly, the main contributions of this work are:

- A distance-based function $\delta$ is used to define the scoring function $f$ of such KGEMs: $f(h, r, t) = \delta(h, r, t)$. A large array of geometric models is purely translational and is based on TransE Bordes et al. [2013] and its extensions. TransE was the earliest introduced geometric model for link prediction. It enforces the sum of the head and relation embeddings to lie in a close neighborhood of the tail embedding. The distance-based function $\delta$ is usually the $L1$ or $L2$ norm. TransE does not properly handle 1-to-N, N-to-1, nor N-to-N relations Wang et al. [2017] and yet has been found to be very efficient in multi-relational settings Chowdhury et al. [2019].

- Several LP approaches have been proposed to complete KGs. Symbolic approaches relying on rule-based Galárraga et al. [2013, 2015], Lajas et al. [2020], Meilicke et al. [2019], Ott et al. [2021] or path-based reasoning Lao et al. [2011], Xiong et al. [2017], Das et al. [2018] are somewhat popular but are not considered in the present work. Instead, KGEMs are the focus of this paper.

- In particular, this work is concerned with assessing the semantic awareness of agnostic KGEMs. The semantic awareness of such models is defined as their ability to score higher entities whose types belong to the domain and range of relations. Agnostic KGEMs are defined as KGEMs that rely solely on the structure of the KG to learn entity and relation representations. In this respect, these models differ according to several criteria, such as the nature of the embedding space or the type of scoring function Wang et al. [2017, 2021a], Ji et al. [2022], Rossi et al. [2021]. In this work, KGEMs are considered with respect to three main families of models that are traditionally distinguished in the literature.

- Geometric models are additive models that consider relations as geometric operations in the latent embedding space. A head entity $h$ is spatially transformed using an operation $\tau$ that depends on the relation embedding $r$. Then the distance between the resulting vector and the tail entity $t$ is used as a measure for assessing the plausibility of a fact $(h, r, t)$. A distance-based function $\delta$ is used to define the scoring function $f$ of such KGEMs: $f(h, r, t) = \delta(h, r, t)$. A large array of geometric models is purely translational and is based on TransE Bordes et al. [2013] and its extensions. TransE was the earliest introduced geometric model for link prediction. It enforces the sum of the head and relation embeddings to lie in a close neighborhood of the tail embedding. The distance-based function $\delta$ is usually the $L1$ or $L2$ norm. TransE does not properly handle 1-to-N, N-to-1, nor N-to-N relations Wang et al. [2017] and yet has been found to be very efficient in multi-relational settings Chowdhury et al. [2019]. A myriad of translational models have been proposed since then, such as TransH Wang et al. [2014], TransR Lin et al. [2015], TransD Ji et al. [2015], TransA Jia et al. [2016] and TransG Xiao et al. [2016]. In addition to these purely translational models, some recent geometric models either replace or combine the translation operation with rotation-wise transformations to make their models even more expressive and deal with difficult relational patterns such as symmetric or anti-symmetric relations. A case in point is RotatE Sun et al. [2019], which considers relations as rotations in a complex latent space. $h$, $r$ and $t$ are all embedded in $\mathbb{C}^d$ where $d$ denotes the embeddings’ dimension. The use of the rotation operation allows RotatE to properly address many relational patterns. In particular, RotatE is able to take account for symmetry – which is not modeled correctly by TransE. In the wake of RotatE, a handful of other roto-translational models have subsequently been proposed in the literature, e.g. QuatE Zhang et al. [2019], DualE Cao et al. [2021] and HAKE Zhang et al. [2020a].

- Semantic matching models are named this way as they usually use a similarity-based function to define their own scoring function. Semantic matching models are also referred to as matrix factorization models or tensor decomposition models, in the sense that a KG can be seen as a 3D adjacency matrix, in other words a three-way tensor. This tensor

The remainder of the paper is structured as follows. Related work is presented in Section 2. Section 3 outlines the main motivations for assessing the semantic awareness of KGEMs. Section 4 subsequently presents $\text{Sem@}^K$, the semantic-oriented metric that fulfills this purpose. $\text{Sem@}^K$ comes in different flavors based on the typology of the datasets at hand and the intended use cases. In Section 5, we detail the datasets and KGEMs used in this work, before presenting the experimental findings in Section 6. A thorough discussion is provided in Section 7. Lastly, Section 8 outlines future research directions.

## 2 Related work

### 2.1 Link prediction using knowledge graph embedding models

Several LP approaches have been proposed to complete KGs. Symbolic approaches relying on rule-based Galárraga et al. [2013, 2015], Lajas et al. [2020], Meilicke et al. [2019], Ott et al. [2021] or path-based reasoning Lao et al. [2011], Xiong et al. [2017], Das et al. [2018] are somewhat popular but are not considered in the present work. Instead, KGEMs are the focus of this paper.

In particular, this work is concerned with assessing the semantic awareness of agnostic KGEMs. The semantic awareness of such models is defined as their ability to score higher entities whose types belong to the domain and range of relations. Agnostic KGEMs are defined as KGEMs that rely solely on the structure of the KG to learn entity and relation representations. In this respect, these models differ according to several criteria, such as the nature of the embedding space or the type of scoring function Wang et al. [2017, 2021a], Ji et al. [2022], Rossi et al. [2021]. In this work, KGEMs are considered with respect to three main families of models that are traditionally distinguished in the literature.

- **Geometric models** are additive models that consider relations as geometric operations in the latent embedding space. A head entity $h$ is spatially transformed using an operation $\tau$ that depends on the relation embedding $r$. Then the distance between the resulting vector and the tail entity $t$ is used as a measure for assessing the plausibility of a fact $(h, r, t)$. A distance-based function $\delta$ is used to define the scoring function $f$ of such KGEMs: $f(h, r, t) = \delta(h, r, t)$. A large array of geometric models is purely translational and is based on TransE Bordes et al. [2013] and its extensions. TransE was the earliest introduced geometric model for link prediction. It enforces the sum of the head and relation embeddings to lie in a close neighborhood of the tail embedding. The distance-based function $\delta$ is usually the $L1$ or $L2$ norm. TransE does not properly handle 1-to-N, N-to-1, nor N-to-N relations Wang et al. [2017] and yet has been found to be very efficient in multi-relational settings Chowdhury et al. [2019]. A myriad of translational models have been proposed since then, such as TransH Wang et al. [2014], TransR Lin et al. [2015], TransD Ji et al. [2015], TransA Jia et al. [2016] and TransG Xiao et al. [2016]. In addition to these purely translational models, some recent geometric models either replace or combine the translation operation with rotation-wise transformations to make their models even more expressive and deal with difficult relational patterns such as symmetric or anti-symmetric relations. A case in point is RotatE Sun et al. [2019], which considers relations as rotations in a complex latent space. $h$, $r$ and $t$ are all embedded in $\mathbb{C}^d$ where $d$ denotes the embeddings’ dimension. The use of the rotation operation allows RotatE to properly address many relational patterns. In particular, RotatE is able to take account for symmetry – which is not modeled correctly by TransE. In the wake of RotatE, a handful of other roto-translational models have subsequently been proposed in the literature, e.g. QuatE Zhang et al. [2019], DualE Cao et al. [2021] and HAKE Zhang et al. [2020a].

- **Semantic matching models** are named this way as they usually use a similarity-based function to define their own scoring function. Semantic matching models are also referred to as matrix factorization models or tensor decomposition models, in the sense that a KG can be seen as a 3D adjacency matrix, in other words a three-way tensor. This tensor

- **RQ3:** does the evaluation of KGEM semantic awareness offer different conclusions on the relative superiority of some KGEMs compared to rank-based metrics?

Accordingly, the main contributions of this work are:

- to evaluate KGEM semantic awareness on any kind of KG, we extend a previously defined semantic-oriented metric and tailor it to support a broader range of KGs.
- we perform an extensive study of the semantic awareness of state-of-the-art KGEMs on mainstream KGs. We show that most of the observed trends apply at the level of families of models.
- we perform a dynamic study of the evolution of KGEM semantic awareness vs. their performance in terms of rank-based metrics along training epochs. We show that a trade-off may exist for most KGEMs.
- our study supports the view that agnostic KGEMs are quickly able to infer the semantics of KG entities and relations.

The remainder of the paper is structured as follows. Related work is presented in Section 2. Section 3 outlines the main motivations for assessing the semantic awareness of KGEMs. Section 4 subsequently presents $\text{Sem@}^K$, the semantic-oriented metric that fulfills this purpose. $\text{Sem@}^K$ comes in different flavors based on the typology of the datasets at hand and the intended use cases. In Section 5, we detail the datasets and KGEMs used in this work, before presenting the experimental findings in Section 6. A thorough discussion is provided in Section 7. Lastly, Section 8 outlines future research directions.
can be decomposed into a set of low-dimensional vectors that actually represent the entity and relation embeddings. Semantic matching models are subdivided into bilinear and non-bilinear models. Bilinear models share the characteristic of capturing interactions between two entity vectors using multiplicative terms. RESCAL Nickel et al. [2011] is the earliest introduced bilinear model for LP. It models entities as vectors and relations as matrices. The components \( w_{i,j} \) of the relation matrix \( W_r \in \mathbb{R}^{d \times d} \) account for the interaction intensity between the \( i \)-th embedding component of \( e_i \in \mathbb{R}^d \) and the \( j \)-th embedding component of \( e_j \in \mathbb{R}^d \) under the relation \( r \). As pointed out by the original author of RESCAL Nickel et al. [2016a], the parameter complexity of this model can drastically increase when the embedding dimension is large. DistMult Yang et al. [2015] alleviates this scalability issue by enforcing all relation-matrices \( W_r \) to be diagonal. Doing so, DistMult trades computational advantages for less expressiveness. Indeed, DistMult is not able to model anti-symmetric relations. However, DistMult still achieves state-of-the-art performance in most cases Kadlec et al. [2017]. ComplEx Trouillon et al. [2016] relies on complex-valued representations for both entities and relations. Because the Hadamard product is used in the scoring function, ComplEx is not commutative in the complex space and can properly model anti-symmetric relations. Other bilinear models with distinctive features were later proposed, such as Analogy Liu et al. [2017], SimplE Kazemi and Poole [2018] and DihEdral Xu and Li [2019] that are supposedly more expressive models. Another category of semantic matching models leverage interactions between entities and relations using non-bilinear operations. For instance, HoIE Nickel et al. [2016b] uses the circular correlation operation, while Tucker Balazevic et al. [2019] relies on the Tucker decomposition.

**Neural network-based models** rely on neural networks to perform LP. In neural networks, the parameters (e.g., weights and biases) are organized into different layers, with usually non-linear activation functions between each of these layers. The first introduced neural-network based model for LP is Neural Tensor Network (NTN) Socher et al. [2013]. It can be seen as a combination of multi-layer perceptrons (MLPs) and bilinear models Ji et al. [2022]. NTN defines a distinct neural network for each relation. This choice of parameterization makes NTN similar to RESCAL in the sense that both models achieve great expressiveness at the expense of computational concerns. The most recent neural network-based models rely on more sophisticated layers to perform a broader set of operations. Convolutional models are by far the most representative family of such models. They use convolutional layers to learn deep and expressive features from the input data, which pass through such layers to undergo convolution with low-dimensional filters \( \omega \). The resulting feature maps subsequently go through dense layers to obtain a final plausibility score. Compared with fully connected neural networks, convolution-based models are able to capture complex relationships with fewer parameters by learning non-linear features. While ConvE Dettmers et al. [2018] reshapes head entity and relation embeddings before concatenating them into a unique input matrix to pass through convolutional layers, ConvKB Nguyen et al. [2018] does not perform any reshaping and also puts the tail embedding into the concatenated input matrix. Other models were later proposed, such as ConvR Jiang et al. [2019] and InteractE Vashishth et al. [2020a] that both process triples independently. Another branch of convolutional models also considers the local neighborhood around each central entity. These are based on Graph Neural Networks (GNNs). The most representative GNN-based model for LP is R-GCN Schlichtkrull et al. [2018]. The key idea is to accumulate messages from the local neighborhood of the central node over multi-hop relations. By doing so, R-GCN is better able to model a long range of relational dependencies. However, R-GCN does not outperform baselines for the LP task Ferrari et al. [2022]. Subsequent models have claimed superiority over R-GCN: SACN Shang et al. [2019] introduces a weighted GCN to adjust the amount of aggregated information from the local neighborhood and KBGAT Nathani et al. [2019] relies on attention mechanism to generate more accurate embeddings. More recently, ComplGCN Vashishth et al. [2020b] and DisenKGAT Wu et al. [2021] showcased impressive performance with regard to the LP task.

### 2.2 Combining embeddings and semantics

The possibility of using additional semantic information has been extensively studied in recent works Jain et al. [2021a], Krompaß et al. [2015], Niu et al. [2020], Cui et al. [2021], Xie et al. [2016], Lv et al. [2018], Wang et al. [2021b]. In general, the semantic information stems directly from an ontology, originally defined by Gruber as an "explicit specification of a conceptualization" Gruber [1993]. Ontologies formally describe a specific application domain of interest (e.g. education, pharmacology, etc.) in which several classes (or concepts) and relations are identified and formally specified. Ontologies support KG construction by providing a schema that specifies the nature of entities, the semantic profile of relations, and other constraints that give the KG a semantic coherence.

A significant part of the literature incorporates such semantic information to constrain the negative sampling procedure and generate meaningful negative triples Jain et al. [2021a], Krompaß et al. [2015]. For instance, type-constrained negative sampling (TCNS) Krompaß et al. [2015] replaces the head or the tail of a triple with a random entity belonging to the same type \( \text{rdfs:} \text{type} \) as the ground-truth entity. Jain et al. Jain et al. [2021a] go a step further and use ontological reasoning to iteratively improve KGEM performance by retraining the model on inconsistent predictions.
Semantic information can also be embedded in the model itself. In fact, some KGEMs leverage ontological information such as entity types and hierarchy.

Embedding models project entities and relations of a KG into a vector space. Thus, the semantics of the original KG may not be fully preserved Jain et al. [2021a], Paulheim [2018]. As stated by Paulheim Paulheim [2018], because embeddings are not meant to preserve the semantics of the KG, they are not interpretable and can severely hinder explainability in domains such as recommender systems. Consequently, Paulheim Paulheim [2018] advocates for semantic embeddings. Similarly, Jain et al. Jain et al. [2021b] perform a thorough evaluation of popular KGEMs to better assess whether embeddings can express similarities between entities of the same type. A key finding is that because of overlapping relations among entities of different types, fine-grained semantics cannot be properly reflected by embeddings. For instance, the task of finding semantically similar entities does not always provide satisfying results when working with entity embeddings Jain et al. [2021b].

2.3 Evaluating KGEM performance for link prediction

KGEM performance is evaluated in two stages. During the validation phase, KGEM performance is evaluated on the validation set $\mathcal{T}_{\text{valid}}$ after regular – often uniform – intervals of epochs. This way, the best epoch is identified. During the test phase, KGEM performance is ultimately evaluated on the test set $\mathcal{T}_{\text{test}}$ after retrieving the optimal model parameters achieved on the best epoch of validation. Whether during the validation or test phase, KGEM performance is evaluated the same way: sifting through every triple of the test (resp. validation) set, both head and tail predictions are performed. In the case of head prediction, this amounts to taking a triple $(h, r, t)$ from the test (resp. validation) set, hiding the ground-truth head entity – resulting in $(?, r, t)$ – and letting the KGEM assign a score to every possible entity as a candidate for the head position. These scores are finally ordered and reflect the plausibility of such facts. Tail prediction is performed analogously on $(h, r, ?)$.

In both cases, the rank of the ground-truth entity from the test (resp. validation) set is used to compute aggregated rank-based metrics based on the top-$K$ scored entities. The rank of the ground-truth entity can be determined in two different ways that depend on how observed facts – i.e. facts that already exist in the KG – are considered. In the raw setting, observed facts outranking the ground-truth are not filtered out, while this is the case in the filtered setting. For instance, assuming head prediction is performed on the given ground-truth triple $(BarackObama, livesIn, USA)$, a KGEM may assign a lower score to this triple than to the following triple: $(MichelleObama, livesIn, USA)$. The latter triple actually represents an observed fact. In the raw setting, this triple would not be filtered out from top-$K$ scored triples. This can cause the evaluation procedure to not properly assess the KGEM performance. This is why in practice, the filtered setting is commonly preferred. In the present work, the filtered setting is also used.

KGEM performance is almost exclusively assessed using the following rank-based metrics: Hits@$K$, Mean Rank (MR), and Mean Reciprocal Rank (MRR) Hoyt et al. [2022]. Disagreements exist as to how and when these metrics can be used and compared properly. In the following, we recall their definitions and discuss their limits.

**Hits@$K$** (Eq. (1)) accounts for the proportion of ground-truth triples appearing in the first $K$ top-scored triples:

$$\text{Hits@}K = \frac{1}{|B|} \sum_{q \in B} \mathbb{1} \left[ \text{rank}(q) \leq K \right]$$

where $B$ is the batch of ground-truth triples, rank$(q)$ is the position of the ground-truth triple $q$ in the sorted list of triples, and $\mathbb{1} \left[ \text{rank}(q) \leq K \right]$ yields 1 if $q$ is ranked between 1 and $K$, 0 otherwise. This metric is bounded in the $[0, 1]$ range and its values increase with $K$, where the higher the better.

**Mean Rank (MR)** (Eq. (2)) corresponds to the arithmetic mean over ranks of ground-truth triples:

$$\text{MR} = \frac{1}{|B|} \sum_{q \in B} \text{rank}(q)$$

This metric is bounded in the $[0, |\mathcal{E}|]$ interval, where $|\mathcal{E}|$ stands for the number of entities in the KG, where the lower the better.

**Mean Reciprocal Rank (MRR)** (Eq. (3)) corresponds to the arithmetic mean over reciprocals of ranks of ground-truth triples:

$$\text{MRR} = \frac{1}{|B|} \sum_{q \in B} \frac{1}{\text{rank}(q)}$$

Contrary to MR, MRR is a metric bounded in the $[0, 1]$ interval. Higher results indicate better performance. Because this metric does not use any threshold $K$ compared to Hits@$K$, it is less sensitive to outliers. In addition, it is often
used for performing early stopping and for tracking the best epoch during training Berrendorf et al. [2020], Hoyt et al. [2022].

As mentioned in Section 1, these metrics present some caveats. LP is often used to complete knowledge graphs, where the Open World Assumption (OWA) prevails. KGs are incomplete and, due to the OWA, an unobserved triple used as a negative one can still be positive. It follows that traditional evaluation methods based on rank-based metrics may systematically underestimate the true performance of a KGEM Wang et al. [2019].

In addition, the aforementioned rank-based metrics have intrinsic and theoretical flaws, as pointed out in several works Berrendorf et al. [2020], Hoyt et al. [2022], Tiwari et al. [2021]. For example, Hits@$K$ does not take into account triples whose rank is larger than $K$. As such, a model scoring the ground-truth in position $K + 1$ would be considered equally good as another model scoring the ground-truth in position $K + d$ with $d \gg 1$. It follows that Hits@$K$ is not a suitable metric for drawing comparisons between models Hoyt et al. [2022]. MR alleviates this concern as it does not consider any threshold $K$. Therefore, MR allows to compare KGEM performance on the same dataset. Nonetheless, MR is sensitive to the number of KG entities (see Eq. (2)) Berrendorf et al. [2020]: a MR of 10 indicates very good performance if the set of entities is in the thousands, but it would indicate poor performance if the set of entities is much more restricted. Therefore, MR does not allow comparisons across datasets.

Recent works recommend using adjusted version of the aforementioned metrics. The Adjusted Mean Rank (AMR) proposed in Ali et al. [2022] compares the mean rank against the expected mean rank under a model with random scores. In Berrendorf et al. [2020], Berrendorf et al. transform the AMR to define the Adjusted Mean Rank Index (AMRI) bounded in the $[0, 1]$ interval. This way, a value of 1 indicates an optimal performance of the model. A value of 0 indicates a model performance similar to a model assigning random scores. A negative value indicates that the model performs worse than random. However, all these attempts at producing a better evaluation framework still focus on the quantitative assessment of KGEMs, i.e. the improvement of already existing rank-based metrics.

In Section 2.2, several approaches incorporating the semantics of entities and relations into the embeddings were mentioned. However, in such cases, the use of semantic information such as entity types and the hierarchy of classes is intended to improve KGEM performance in terms of the aforementioned rank-based metrics only. The underlying semantics of KGs is considered as an additional source of information during training but the ability of KGEMs to generate predictions in accordance with these semantic constraints is never directly addressed. This encourages further assessment of the semantic capabilities of KGEMs, as firstly suggested in Hubert et al. [2022b]. In our work, we directly address this issue by assessing to what extent KGEMs are able to give high scores to triples whose head (resp. tail) belongs to the domain (resp. range) of the relation. When such information is not available – i.e. the KG does not rely on a schema containing rdfs:domain and rdfs:range properties – extensional constraints can still be used to evaluate the semantic capabilities of KGEMs, as detailed in Section 4.

3 Motivations and problem formulation

3.1 Motivating example

To motivate the use of Sem@$K$ to evaluate KGEM quality, this section builds upon a minimalist example which is representative of the issue encountered while benchmarking the performance of several KGEM on the same dataset. As depicted in Fig. 2, two KGEMs that have been trained on the whole training set are tested on a batch of test triples. These KGEMs are referred to as Model A and Model B. Without loss of generality, it is assumed that the test set only comprises the three triples shown in Fig. 2. For the sake of clarity, only the tail prediction pass and the top-5 ranked candidate entities are depicted in Fig. 2. It should be noted that the performance of both models are strictly equal in terms of MR, MRR and Hits@$K$ with $K \leq 5$: MR=8/3, Hits@1=1/3, Hits@3=2/3 and Hits@5=1. MRR can only be computed knowing the total number of entities in the KG but two models having the same MR on the same dataset have de facto the same MRR as well. Distinguishing between these two models by relying solely on traditional rank-based metrics is not possible. One might even draw the misleading conclusion that these models are equally good. But they are not: Model A gives high scores to semantically valid entities with regard to the range of relations, while Model B semantic awareness is very low. In other words, Model B does not capture the semantic profile of relations well. This case in point illustrates the need for using semantic-oriented metrics alongside common rank-based metrics, so as to better assess the overall quality of KGEMs.

3.2 Problem formulation

The traditional evaluation of KGEMs solely based on rank-based metrics can be flawed for several reasons. First, KGEMs benchmarked on the same test sets can exhibit very similar results. Using only rank-based metrics, the final choice only depends on the best achieved MRR and/or Hits@$K$. This raises the questions whether the chosen
Sem@K: Is my KGEM semantic-aware?

Figure 2: Motivating example. Tail prediction is performed for the three test triples contained in the upper insert. Model A and Model B output scores for each possible entity and only the top-five ranked tail candidates are depicted here. Model A and Model B have the same Hits@1, Hits@3 and Hits@5 values. But Model A has better semantic capabilities. Green, blue and white cells respectively denote the ground-truth entity, entities other than the ground-truth and semantically valid, and entities other than the ground-truth and semantically invalid model is actually the best one, or whether its slight superiority over other KGEMs can be due to other factors such as better hyperparameter tuning or better modeling of a relational pattern highly present in the test set. Moreover, using only rank-based metrics does not provide the full picture of KGEM quality for the downstream LP task, as some dimensions of KGEMs are left unassessed (see Section 2). In this work, contrary to the mainstream approach consisting in comparing KGEM performance exclusively in terms of rank-based metrics, the trained KGEMs are also evaluated in terms of Sem@K which measures the ability of KGEMs to predict semantically valid triples with respect to the domain and range of relations.

4 Measuring KGEMs semantic awareness with Sem@K

The standard LP evaluation protocol consists in reporting aggregated results, considering the rank-based metrics presented in Section 2.3. As mentioned above, these metrics only provide a partial picture of KGEM performance Berrendorf et al. [2020]. To give a more comprehensive assessment of KGEMs, we aim at assessing their semantic awareness using our proposed metric called Sem@K Hubert et al. [2022a,b]. In Hubert et al. [2022a], Sem@K was specifically defined for the recommendation task which was seen as predicting tails for a unique target relation. Sem@K was then extended in Hubert et al. [2022b] to the more generic LP task, where not only tails but also heads are corrupted and all relations are considered. In this work, this original formalization for LP is presented in Section 4.1, and enriched to take into account schemaless KGs (Section 4.3), or KGs with a class hierarchy (Section 4.4). As a consequence, Sem@K comes in 3 different versions (respectively denoted Sem@K[base], Sem@K[ext], and Sem@K[wup]) so as to adapt to KG typology. These distinct versions and their adequacy regarding the KG at hand are summarized in Table 1 and further detailed below. In the following, when no suffix is provided, it is assumed that we are concerned with Sem@K in general, regardless of the actual version.
### 4.1 Definition of Sem@K[base]

This version of Sem@K (Eq. (4)) accounts for the proportion of triples that are semantically valid in the first $K$ top-scored triples:

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

(4)

where, given a ground-truth triple $q = (h, r, t)$, $S^K_q$ is the list of the top-$K$ candidate triples scored by a given KGEM (i.e., by predicting the tail for $(h, r, ?)$ or the head for $(?, r, t)$). The function compatibility $(q, q')$ (Eq. (5)) assesses whether the candidate triple $q'$ is semantically compatible with its ground-truth counterpart $q$. In this work, by semantic compatibility we refer to the fact that the predicted head (resp. tail) belongs to the domain (resp. range) of the relation:

$$
\text{compatibility}(q, q') = \begin{cases} 
1, & \text{if } \text{types}(q'_h) \cap \text{domain}(r) \neq \emptyset \land \text{types}(q'_t) \cap \text{range}(r) \neq \emptyset \\
0, & \text{otherwise}
\end{cases}
$$

(5)

where types($e$) returns the types of entity $e$ and domain($r$) (resp. range($r$)) is the domain (resp. range) of the relation $r$. $q_r, q'_r,$ and $q'_t$ denote the ground-truth relation, the head and the tail of the ranked triple $q'$, respectively. It is noteworthy that with this formula, we allow entities to instantiate multiple types, and domains and ranges to be defined with multiple types. Sem@K is bounded in the $[0, 1]$ interval. Compared to Hits@K (Eq. (1)), Sem@K is non-monotonic: increasing $K$ can lead to either lower or higher Sem@K values.

### 4.2 A note on Sem@K, untyped entities, and the Open World Assumption

Traditional KGEM evaluation can be performed in all situations, regardless of whether entities are typed or whether the KG comes with a proper schema. When measuring the semantic capability of KGEMs – e.g. with Sem@K – some concerns arise. For instance, a fair question to ask is the following: how should untyped entities be considered? Indeed, in some KGs, some entities are left untyped. For example, in DBpedia the entity dbr:1..FC_Union_Berlin_players does not belong to any other class than owl:Thing. In some other cases as in DB15K and DB100K d’Amato et al. [2021], entities have incomplete typing.

Under the OWA, an untyped entity should not count in the calculation of Sem@K since it is not possible to determine whether this entity has no types or no known types. Although this seems to be a fair option, it raises a major issue: it makes possible to score different sets of entities with rank-based metrics and Sem@K, which is not desirable. If there are $M$ untyped entities in an ordered list of ranked entities, Hits@K, MR and MRR are calculated regardless of this fact, i.e. still taking into account the $M$ untyped entities. However, Sem@K cannot be calculated for these $M$ untyped entities. As such, MR, MRR and Hits@K would be calculated on the original entity set, whereas Sem@K would be computed on a different set of entities. This issue would be even more acute in the case of Sem@1 when the first ranked entity is untyped: Hits@1 and Sem@1 would be calculated on two different entities, which is not acceptable. Consequently, one strategy consists in removing untyped entities from the evaluation protocol, both regarding rank-based and semantic-based metrics. By doing so, consistency is ensured in the ranked list of entities across rank-based and semantic metrics evaluation.

### 4.3 Sem@K[ext] for schemaless KGs

Not all KGs come with a proper schema, e.g. relations do not appear in any rdfs:domain or rdfs:range clauses. In that particular situation, it can still be desirable to assess the semantic awareness of KGEMs. One approach is to maintain a list of all entities that have been observed as heads (resp. tails) of each relation $r$:

$$\text{domain}(r) = \{ e : \exists (e, r, t) \in T \}$$

(resp. range$(r) = \{ e : \exists (h, r, e) \in T \}$). Hence, in this case, Sem@K[ext] is defined as in Eq. (4) and (5) but using these definitions of domain and range. Therefore, an entity will be considered as a semantically valid head (resp. tail) with respect to the relation if this entity appears as a head (resp. tail) in any other triple observed in the KG with the same relation.
4.4 $\text{Sem@K}[\text{wup}]$: a hierarchy-aware version of $\text{Sem@K}$

Sem@K as previously defined equally penalizes all entities that are not of the expected type. However, KGs may be equipped with a class hierarchy that, in turn, can support a more fine-grained penalty for entities depending on the distance or similarity between their type and the expected domain (resp. range) in this hierarchy. To illustrate, consider Figure 3 that depicts a subset of the DBpedia ontology $\text{dbo}$ class hierarchy. Using the hierarchy-free version of Sem@K for the test triple ($\text{dbr:The_Social_Network}$, $\text{dbo:director}$, $\text{dbr:David_Fincher}$), predicting $\text{dbr:Friends}$ or $\text{dbr:Central_Park}$ as head would be penalized the same way in the compatibility function. However, it is clear that an entity of class $\text{dbo:TelevisionShow}$ is semantically closer to $\text{dbo:Film}$ – the domain of the relation $\text{dbo:director}$ – than an entity of class $\text{dbo:Park}$, and thus should be less penalized.

To leverage such a semantic relatedness between concepts in Sem@K, the compatibility function can be adapted accordingly:

$$\text{compatibility}(q, q') = \min \left( \max_{c \in \text{type}(q'_r)} \sigma(c, c'), \max_{c' \in \text{domain}(q'_r)} \sigma(c, c') \right)$$  \hspace{1cm} (6)$$

where $\sigma(c, c')$ measure the semantic similarity between the two classes $c$ and $c'$ based on the class hierarchy. It should be noted that in this formula $\text{type}(e)$ only returns the most specific classes instantiated by $e$.

Several similarity measures $\sigma$ have been proposed in the literature Rada et al. [1989], Wu and Palmer [1994], Leacock and Chodorow [1998], Resnik [1999], Li et al. [2003]. In this work, the Wu-Palmer similarity Wu and Palmer [1994] (Eq. 7) is used:

$$\sigma(c, c') = \frac{2 \times \delta(c \land c', \rho)}{\delta(c \land c', \rho) + \delta(c' \land c', \rho) + 2 \times \delta(c \land c', \rho)}$$  \hspace{1cm} (7)$$

where $\rho$ is the root of the hierarchy (e.g. $\text{owl:Thing}$), $\delta(c, c')$ is the number of edges linking $c$ to $c'$, and $c \land c'$ represents the least common subsumer of $c$ and $c'$. The Wu-Palmer similarity is well suited to a class hierarchy and provides a good indication of the semantic relatedness between the domain (resp. range) class and the classes of the chosen entity. This gives rise to the Sem@K$[\text{wup}]$ version.

Considering the example in Fig. 3, using the Wu-Palmer score in the calculation of Sem@K, a head prediction of $\text{dbr:Friends}$ and $\text{dbr:Central_Park}$ for the ground-truth triple ($\text{dbr:The_Social_Network}$, $\text{dbo:director}$, $\text{dbr:David_Fincher}$) are now differently penalized. The instance $\text{dbr:Friends}$ is of type $\text{dbo:TelevisionShow}$, so we have: $\sigma(\text{dbo:TelevisionShow}, \text{dbo:Film}) = 1/2$. The instance $\text{dbr:Central_Park}$ is of type $\text{dbo:Park}$, so we have: $\sigma(\text{dbo:Park}, \text{dbo:Film}) = 0$. This illustrates that incorporating the Wu-Palmer score into Sem@K calculation leads to more precise semantic comparisons that take into account the available class hierarchy. It should be noted that comparing the classes of a candidate entity with the expected class can result in the same penalization as the vanilla version of Sem@K. However, in most cases, two classes do not lie in a completely disjoint part of the hierarchy of classes. As such, the Wu-Palmer score between the classes of a candidate entity and the expected class is rarely 0.
5 Experimental setting

5.1 Datasets

In order to draw reliable and general conclusions, a broad range of KGs are used in our experiments. They have been chosen due to their mainstream adoption in recent research works around KGEMs for LP and the fact that they have different characteristics (e.g. entities, relations, classes, presence of a class hierarchy). In this section, the schema-defined and schemaless KGs used in the experiments are detailed. Note that in our experiments, all the schema-defined KGs come with a class hierarchy inherited from either Freebase Bollacker et al. [2008], DBpedia Auer et al. [2007], YAGO Suchanek et al. [2007], or schema.org. To meet requirements for evaluating KGEMs w.r.t. Sem@K (i.e. classes instantiated by entities, domain and range for relations), when necessary, subsets of schema-defined KGs are used, as explained in Section 5.1.1. Among the 4 schema-defined KGs presented hereafter, FB15K237-ET, DB93K and YAGO3-37K are derived from already existing KGs, while YAGO4-19K was specifically created and is made available on Zenodo2 and GitHub3. The other KGs are made available on GitHub4.

5.1.1 Schema-defined KGs

The statistics of the datasets FB15K237-ET, DB93K, YAGO3-37K and YAGO4-19K used in our experiments are provided in Table 2. As discussed in Section 4.2, to create an experimental evaluation setting as unbiased and flawless as possible, the schema-defined KGs used in the experiments are filtered to keep typed entities only. This way, Sem@K is calculated under the CWA.

FB15K237-ET derives from FB15K Bordes et al. [2013] – a dataset extracted from the cross-domain Freebase KG Bollacker et al. [2008]. In these experiments, we do not use FB15K, as it has been noted that this dataset suffers from test leakage, i.e., a large number of test triples can be obtained by simply inverting the position of the head and tail in the train triples Toutanova and Chen [2015]. The later introduced FB15K237 dataset Toutanova and Chen [2015] is a subset of the original FB15K without these inverse relations. To the best of our knowledge, there is no schema-defined version of FB15K237. However, a schema-defined version of FB15K is provided in Xie et al. [2016]. Consequently, we based ourselves on this version of FB15K and mapped the extracted entity types, relation domains and ranges to the entities and relations found in FB15K237. The resulting schema-defined version of FB15K237 is named FB15K237-ET and includes only typed entities. Besides, validation and test sets contain triples whose relation have a well-defined domain (resp. range), as well as more than 10 observed 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, at least until K = 10.

DB93K is a subset of DB100K, which was first introduced in Ding et al. [2018]. A slightly modified version of DB100K has been proposed in d’Amato et al. [2021]. Contrary to the initial version of DB100K, the latter version is schema-defined: entities are properly typed and most relations have a domain and/or a range. This second version is considered in the following experiments. However, some inconsistencies were found in the dataset. Some DBpedia entities only instantiate Wikidata5 or schema.org6 classes, while instantiation of classes from the DBpedia ontology actually exist. Moreover, some entities are only partially typed. It must also be noted that domains and ranges of relations have been extracted from DBpedia more than two years ago. DBpedia is a communautary and open-source project: DBpedia classification system relies on human curation, which sometimes implies a lack of coverage for some resources and updates for others. Consequently we associated all relations of DB100K to their most up-to-date domains and ranges7. Similarly for entities, we associated all entities to their most up-to-date classes. Finally, we removed all untyped entities as well as validation and test relations having less than 10 observed entities, so as not to unfairly penalize Sem@K results in the validation and test phases.

YAGO3-37K derives from the schema-defined YAGO39K dataset Lv et al. [2018] extracted from the cross-domain YAGO3 KG Suchanek et al. [2007]. Compared to the original YAGO39K, in our experiments only typed entities are kept. In addition, relations having less than 10 observed heads or tails in the training set are discarded from the validation and test splits, for the same reason that keeping them would not reflect the actual Sem@K values. It should be noted that in the YAGO3 ontology, most relations have very generic domains and ranges which are very close to the root of the ontology hierarchy. To produce a more challenging evaluation setting of the models’ semantic awareness, a

---

2https://doi.org/10.5281/zenodo.7526244
3https://github.com/pmonnin/YAGO4-LP
4https://github.com/nicolas-hbt/benchmark-sematk
5https://www.wikidata.org/
6https://schema.org/
7SPARQL queries were fired against DBpedia as of November 9, 2022
subset of hard relations was identified and only validation and test triples whose relation belongs to this subset are kept. The resulting dataset is named YAGO3-37K.

We built YAGO4-19K with several other subsets of the YAGO4 knowledge graph Tanon et al. [2020]. Similarly to other datasets, we focused on relations with a defined domain and range, and more than 10 triples to constitute the validation and test sets. We purposely favored difficult relations to feature in the validation and test sets. To enrich the training set, additional relations were added based on a manual selection. Selected relations in validation and test sets as well as additional relations in the training set can be found on the GitHub repository of the dataset. It should be noted that the class hierarchy associated with entities in YAGO4 is schema.org.

5.1.2 Extensional KGs

Another range of datasets used in these experiments do not come with an ontological schema. In particular, this means relations do not have a clearly-defined domain or range. Although Codex-S and Codex-M are based on the Wikidata schema which does possess property constraints linking subject types to value type constraints, we limit ourselves to KGs that represent this information with rdfs:domain and rdfs:range predicates. Consequently, in this work, we only report Sem@K[ext] results for Codex-S and Codex-M. The statistics of the datasets Codex-S, Codex-M and WN18RR used in our experiments are provided in Table 3.

WN18RR Dettmers et al. [2018] originates from WN18, which is a subset of the WordNet KG Miller [1995]. As for FB15K, Toutanova et al. Toutanova and Chen [2015] reported a huge test leakage in the original WN18 dataset. More specifically, 94% of the train triples have inverse relations that are linked to test triples. Dettmers et al. Dettmers et al. [2018] remove all inverse relations to propose WN18RR. In WN18RR, entities are nouns, verbs, and adjectives. Relations such as hypernym and derivationally related from hold between observed entities. Such relations are not linked to any rdfs:domain or rdfs:range predicates. Besides, relations such as derivationally related from can contain nouns, verbs or adjectives as both head or tail. It follows that it is not possible to infer any expected entity type for relations – in this case the word qualifier. This is why WN18RR is used in the extensional setting.

Codex-S Safavi and Koutra [2020] and Codex-M Safavi and Koutra [2020] are datasets extracted from Wikidata and Wikipedia. They cover a wider scope and purposely contain harder facts than most KGs Safavi and Koutra [2020]. Consequently, these datasets prove to be more challenging for the link prediction task. Compared to WN18RR, Codex-S and Codex-M contain entity types, relation descriptions and Wikipedia page extracts. Nonetheless, Wikidata does not contain rdfs:domain or rdfs:range predicates. The property constraints present in Wikidata are harder to manipulate than the rdfs:domain and rdfs:range clauses found in DBpedia. As such, in our experiments Codex-S and Codex-M are used in the extensional setting. Codex-S and Codex-M initially come with already generated hard negatives. In our experiments, we do not directly use these provided negative triples. Instead, we use the same Uniform Random Negative Sampling schema as for other datasets.

5.2 Baseline models

In this work, the semantic awareness of the most popular semantically agnostic KGEMs is analyzed. More specifically, the translational models TransE Bordes et al. [2013] and TransH Wang et al. [2014], the semantic matching models DistMult Yang et al. [2015], ComplEx Trouillon et al. [2016] and SimplE Kazemi and Poole [2018], and the convolu-
where will be made between pure convolutional KGEMs (ConvE, ConvKB) and GNNs (R-GCN, CompGCN). Although the Wang et al. [2014] is an extension of TransE. It allows entities to have distinct representations when involved TransH often showcases better performance than TransE with only slightly more parameters Wang et al. [2014]. The scoring function is by the relation-specific translation vector d.

ConvE subsequently uses 2D convolution over between all the aforementioned entity and relation embeddings:

$$f(h, r, t) = \frac{1}{2} \left( \langle e_h, e_r, e_t \rangle + \langle e_h, e_r^{-1}, e_t \rangle \right)$$

### Table 4: Summary of the KGEMs used in the experiments

<table>
<thead>
<tr>
<th>Model Family</th>
<th>Model</th>
<th>Scoring Function</th>
<th>Loss Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geometric</td>
<td>TransE</td>
<td>$|e_h + e_r - e_t|_p$</td>
<td>Pairwise Hinge</td>
</tr>
<tr>
<td></td>
<td>TransH</td>
<td>$|e_{h\perp} + d_r - e_{t\perp}|_p$</td>
<td>Pairwise Hinge</td>
</tr>
<tr>
<td>Semantic Matching</td>
<td>DistMult</td>
<td>$\langle e_h, W_r, e_t \rangle$</td>
<td>Pairwise Hinge</td>
</tr>
<tr>
<td></td>
<td>ComplEx</td>
<td>$Re \left( e_h \odot e_r \odot \bar{t} \right)$</td>
<td>Pointwise Logistic</td>
</tr>
<tr>
<td></td>
<td>SimplE</td>
<td>$\frac{1}{2} \left( \langle e_h, e_r, e_t \rangle + \langle e_h, e_r^{-1}, e_t \rangle \right)$</td>
<td>Pointwise Logistic</td>
</tr>
<tr>
<td>Convolutional</td>
<td>ConvE</td>
<td>$g \left( vec \left( g \left( \text{concat} \left( e_h, e_r \right) * \omega \right) \right) W \cdot e_t \right)$</td>
<td>Binary Cross-Entropy</td>
</tr>
<tr>
<td></td>
<td>ConvKB</td>
<td>$\text{concat} \left( g \left( \left( e_h, e_r, e_t \right) * \omega \right) \right) \cdot w$</td>
<td>Pointwise Logistic</td>
</tr>
<tr>
<td></td>
<td>R-GCN</td>
<td>DistMult decoder</td>
<td>Binary Cross-Entropy</td>
</tr>
<tr>
<td></td>
<td>CompGCN</td>
<td>ConvE decoder</td>
<td>Binary Cross-Entropy</td>
</tr>
</tbody>
</table>

TransE Bordes et al. [2013] is the earliest translational model. It learns representations of entities and relations such that for a triple $(h, r, t)$, $e_h + e_r \approx e_t$, where $e_h$, $e_r$, and $e_t$ are the head, relation and tail embeddings, respectively. The scoring function is

$$f(h, r, t) = d(e_h + e_r - e_t)$$

where $d$ is a distance function, usually the $L1$ or $L2$ norm.

TransH Wang et al. [2014] is an extension of TransE. It allows entities to have distinct representations when involved in different relations. Specifically, $e_h$ and $e_t$ are projected into relation-specific hyperplanes with projection matrices $W_r$. If $(h, r, t)$ holds, the projected entities $e_{h\perp} = e_h - w_r e_h$ and $e_{t\perp} = e_t - w_r e_t$ are expected to be linked by the relation-specific translation vector $d_r$. Thus, the scoring function is

$$f(h, r, t) = d(e_{h\perp} + d_r - e_{t\perp})$$

TransH often showcases better performance than TransE with only slightly more parameters Wang et al. [2014].

DistMult Yang et al. [2015] is a semantic matching model. It is characterized as such because it uses a similarity-based scoring function and matches the latent semantics of entities and relations by leveraging their vector space representations. More specifically, DistMult is a bilinear diagonal model that uses a trilinear dot product as its scoring function:

$$f(h, r, t) = \langle e_h, W_r, e_t \rangle$$

It is similar to RESCAL Nickel et al. [2011] – the very first semantic matching model – but restricts relation matrices $W_r \in \mathbb{R}^{d \times d}$ to be diagonal.

ComplEx Trouillon et al. [2016] is also a semantic matching model. It extends DistMult by using complex-valued vectors to represent entities and relations: $e_h, e_r, e_t \in \mathbb{C}^d$. As a result, ComplEx is better able to model antisymmetric relations than DistMult Sun et al. [2019]. Its scoring function uses the Hadamard product:

$$f(h, r, t) = Re \left( e_h \odot e_r \odot \bar{t} \right)$$

where $\bar{t}$ denotes the conjugate of $t$.

Simple Kazemi and Poole [2018] models each fact in both a direct and an inverse form. To do so, an entity $e$ is simultaneously represented by two vectors $e^h, e^i \in \mathbb{R}^d$. Depending on whether $e$ appears as head or tail in a given triple, either $e^h$ or $e^i$ is used. Consequently, the two entity representations $e^h$ and $e^i$ are learned independently. Likewise, each relation $r$ comes with a direct and an inverse vectors $e_r$ and $e_{r^{-1}}$. The scoring function reflects the interaction between all the aforementioned entity and relation embeddings:

$$f(h, r, t) = \frac{1}{2} \left( \langle e_h^h, e_r, e_t \rangle + \langle e_r^h, e_r^{-1}, e_t^h \rangle \right)$$

ConvE Dettmers et al. [2018] first reshapes entity and relation embeddings and then concatenates them into a 2D matrix $[h; r]$. To model the interactions between entities and relations, ConvE subsequently uses 2D convolution over...
embeddings and layers of nonlinear features. The output is ultimately scored against the tail embedding \( t \) using the dot product. More precisely, the following scoring function is used:

\[
    f(h, r, t) = g \left( \text{vec} \left( g \left( \text{concat} \left( \hat{e}_h, \hat{e}_r \right) \ast \omega \right) \right) \right) W \cdot e_t
\]

(13)

where \( g \) denotes a non-linear function, vec is the vectorization operation reshaping a tensor into a vector, concat is the concatenation operator, \( \ast \) and \( \cdot \) denote a convolution and a dot product, respectively, \( \hat{e} \) denotes a 2D reshaping of \( e \) and \( \omega \) is the set of convolutional filters.

**ConvKB** Nguyen et al. [2018] also represents entities and relations as same-sized vectors. However, ConvKB does not reshape the embeddings of entities and relations. Plus, ConvKB also considers the tail embedding in the concatenation operation, thus obtaining the 2D matrix \([h; r; t]\) after concatenation. Convolution by a set \( \omega \) of \( T \) filters of shape \( 1 \ast 3 \) is applied on this input. The resulting \( T \ast 3 \) feature map then passes through a dense layer with one neuron and a weight matrix \( W \). Finally, the following scoring function assesses the plausibility of a given triple \((h, r, t)\):

\[
    f(h, r, t) = \text{concat} \left( g \left( \left[ \hat{e}_h, \hat{e}_r, \hat{e}_t \right] \ast \omega \right) \right) \cdot w
\]

(14)

It has been claimed that the concatenation of a set of feature maps generated by convolution should increase the learning ability of latent features compared to ConvE Ji et al. [2022]. However, recent works point out the evaluation procedure used in the original implementation of ConvKB Shang et al. [2019], Sun et al. [2020], which may result in overly optimistic results on the benchmark datasets FB15K237 and WN18RR. Nonetheless, due to the popularity of this model, we choose to include it in our experiments.

**R-GCN** Schlichtkrull et al. [2018] extends the idea of applying graph convolutional networks (GCNs) to multi-relational data. R-GCN on local graph neighborhoods and applies a convolution operation to the neighbors of each entity. By aggregating the messages coming from all the neighbors of an entity, the embedding of the latter is updated in accordance. Each entity thus has a hidden representation which directly depends on the hidden representations of its neighbors. This process of accumulating messages (i.e. the hidden representations of neighboring entities) and aggregating them so as to update the hidden representation of the central node is performed for each layer of the R-GCN model. More formally, the hidden representation of the entity \( i \) in the layer \((l + 1)\) is defined as:

\[
    h^{(l+1)}_i = \sigma \left( W_{0}^{(l)} h_i^{(l)} + \sum_{r \in R} \sum_{j \in \mathcal{N}_r^i} \frac{1}{c_{i,r}} W_r^{(l)} h_j^{(l)} \right)
\]

(15)

where \( \sigma(\cdot) \) can be any element-wise activation function, \( \mathcal{N}_r^i \) denotes the set of neighboring entities of entity \( i \) considering the relation \( r \), \( W_r^{(l)} \) is the relation-specific weight matrix for layer \( l \) and \( c_{i,r} \) is a normalization constant. To update entity hidden representation taking into account its previous state, a self-connection is also incorporated into the activation function \( \sigma(\cdot) \). Such layers can be stacked multiple times in order to better learn interactions and dependencies across several relational steps. However, as applying Eq.(15) directly would dramatically increase the number of parameters for KGs having lots of different relation types, basis-decomposition and block-diagonal-decomposition are proposed in Schlichtkrull et al. [2018] to reduce model parameter size and prevent overfitting. In the case of basis decomposition which is used in the original paper, the relation-specific weight matrix \( W_r^{(l)} \) is decomposed into a linear combination of basis transformation \( V_b^{(l)} \) with coefficients \( a_{rb}^{(l)} \):

\[
    W_r^{(l)} = \sum_{b=1}^{B} a_{rb}^{(l)} V_b^{(l)}
\]

(16)

**CompGCN** Vashishth et al. [2020b] improves over R-GCN by not only learning entity representations but also relation representations. Concretely, CompGCN performs a composition operation \( \phi(\cdot) \) over each edge in the neighborhood of a central node. The composed embeddings are subsequently convolved with direction-specific weight matrices \( W_O \) and \( W_I \) – for original and inverse relations, respectively. In addition to the entity representation update, relation representations are also updated individually:

\[
    h^{(l+1)}_r = W^{(l)}_{rel} h^{(l)}_r
\]

(17)

where \( W_{rel} \) is a learnable transformation matrix which projects all the relations into the same embedding space as entities.

### 5.3 Implementation details and hyperparameters

For the sake of comparisons, MRR, Hits@K and Sem@K all need to rely on the same code implementation. More specifically, for R-GCN and CompGCN, existing implementations were reused and Sem@K values were calculated...
A major finding is that models performing well with respect to rank-based metrics are not necessarily the most necessary to support our claim, some of them are duplicated in the body text.

For the sake of clarity, the complete range of tables and plots are placed in Appendix B and C, respectively. When in the following, we perform an extensive analysis of the results obtained using the aforedescribed KGEMs and KGs. For the sake of clarity, the complete range of tables and plots are placed in Appendix B and C, respectively. Whenever necessary to support our claim, some of them are duplicated in the body text.

### 6 Results

In the following, we perform an extensive analysis of the results obtained using the aforedescribed KGEMs and KGs. For the sake of clarity, the complete range of tables and plots are placed in Appendix B and C, respectively. Whenever necessary to support our claim, some of them are duplicated in the body text.

#### 6.1 Semantic awareness of KGEMs

This section draws on the Sem@K values (see Tables 8 and 9) achieved at the best epoch in terms of MRR to provide an analysis of the semantic awareness of state-of-the-art KGEMs. In other words, for such models we only consider a snapshot of their best epochs in terms of rank-based metrics.

A major finding is that models performing well with respect to rank-based metrics are not necessarily the most competitive when it comes to their semantic capabilities. On YAGO3-37K (see Table 5) for instance, ConvE showcases impressive MRR and Hits@K values. However, it is far from being the best KGEM in terms of Sem@K, as it is outperformed by CompGCN, R-GCN and all translational models.

From a coarse-grained viewpoint, conclusions about the relative superiority of models with the distinct consideration of rank-based metrics and semantic awareness can be generalized at the level of models families. For example, semantic matching models (DistMult, ComplEx, SimplE) globally achieve better MRR and Hits@K values while their semantic capabilities are in most cases lower than the ones of translational models (TransE, TransH) – see Table 8 and Table 9 for detailed results w.r.t. rank-based and semantic-oriented metrics. A condensed view of the comparison between MRR and Sem@10 is also reported in Fig. 4 and Fig. 5. The respective hierarchies of such models for the benchmarked schema-defined and schemaless KGs are depicted in Fig. 6 and Fig. 7, respectively. It is clearly visible that KGEMs are grouped by family. In particular, GNNs and translational models showcase very promising semantic capabilities. GNNs are almost always the best regarding Sem@K[ext] values – not only for schemaless KGs (Fig. 7), but also for schema-defined KGs (see Table 8 for full results, and Fig. 4 for a quick glimpse). This means GNNs are more capable of predicting entities that have been observed as head (resp. tail) of a given relation. Translational models are very

---

Table 5: Results on YAGO3-37K. Bold fonts and gray cells denote the best achieved results and the worst achieved results among the models reported in the table, respectively. Full results are available in Appendix B, Table 8.

<table>
<thead>
<tr>
<th>Model Family</th>
<th>Model</th>
<th>MRR</th>
<th>Rank-based</th>
<th>Sem@K[base]</th>
<th>Sem@K[wup]</th>
<th>Sem@K[ext]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>H@1</td>
<td>H@3</td>
<td>H@10</td>
<td>S@1</td>
</tr>
<tr>
<td>Geometric</td>
<td>TransE</td>
<td>.184</td>
<td>.080</td>
<td>.198</td>
<td>.308</td>
<td>.989</td>
</tr>
<tr>
<td></td>
<td>TransH</td>
<td>.187</td>
<td>.091</td>
<td>.199</td>
<td>.515</td>
<td>.965</td>
</tr>
<tr>
<td>Convolutional</td>
<td>ConvE</td>
<td>.493</td>
<td>.350</td>
<td>.578</td>
<td>.775</td>
<td>.933</td>
</tr>
<tr>
<td></td>
<td>R-GCN</td>
<td>.117</td>
<td>.046</td>
<td>.110</td>
<td>.254</td>
<td>.963</td>
</tr>
<tr>
<td></td>
<td>CompGCN</td>
<td>.299</td>
<td>.269</td>
<td>.464</td>
<td>.663</td>
<td>.998</td>
</tr>
</tbody>
</table>

---

10 https://github.com/uma-pi1/kge
Figure 4: MRR and Sem@10 results achieved at the best epoch in terms of MRR for each model and on each schema-defined dataset

Figure 5: MRR and Sem@10 results achieved at the best epoch in terms of MRR for each model and on each schemaless dataset
Sem@$K$: Is my KGEM semantic-aware?

![Diagram](image)

(a) FB15K237-ET

(b) DB93K

(c) YAGO3-37K

(d) YAGO4-19K

Figure 6: Sem@$K$[base] comparisons between KGEMs on the 4 benchmarked schema-defined KGs. Colors indicate the family of models: blue, purple, green, and yellow cells denote GNNs, translational, convolutional, and semantic matching models, respectively. Regarding Sem@$K$, the relative hierarchy of models is consistent across KGs and we can clearly see that KGEMs are grouped by families of models competitive in terms of Sem@$K$[base]. In Fig. 6, we clearly see that they consistently rank among the best performing models regarding Sem@$K$[base]. Interestingly, the semantic matching models DistMult, ComplEx and SimplE perform relatively poorly. This observation holds regardless of the nature of the KG, as they systematically rank among the worst performing models for schema-defined (Fig. 6) and schemaless (Fig. 7) KGs.

Therefore, it seems that translational models are better able at recovering the semantics of entities and relations to properly predict entities that are in the domain (resp. range) of a given relation, while semantic matching models might sometimes be better at predicting entities already observed in the domain (resp. range) of a given relation (e.g. DistMult reaches very high Sem@$K$[ext] values on Codex-S, as evidenced in Fig. 7a). In cases when translational and semantic matching models provide similar results in terms of rank-based metrics, the nature of the dataset at hand – whether it is schema-defined or schemaless – might thus strongly influence the final choice of a KGEM.

Interestingly, CompGCN which is by far the most recent and sophisticated model used in our experiments, outperforms all the other models in terms of semantic awareness as, with very limited exceptions, it is the best in terms of Sem@$K$[base], Sem@$K$[ext] and Sem@$K$[wup]. In addition, it should be noted that R-GCN provides satisfying results as well. Except in a very few cases (e.g. outperformed by ComplEx on Codex-S and WN18RR in terms of Sem@1), R-GCN showcases better semantic awareness than semantic matching models. Most of the time, R-GCN also provides comparable or even higher semantic capabilities than translational models. In particular, Sem@$K$ values achieved with R-GCN are similar to the ones achieved with TransE and TransH (e.g. on FB15K237-ET and YAGO3-
Sem@\(K\): Is my KGEM semantic-aware?

![Diagram of KGEM families](image)

(a) Codex-S

(b) Codex-M

(c) WN18RR

Figure 7: Sem@\(K[\text{ext}]\) comparisons between KGEMs on the 3 benchmarked schemaless KGs. Colors indicate the family of models: blue, purple, green, and yellow cells denote GNNs, translational, convolutional, and semantic matching models, respectively. Regarding Sem@\(K\), the relative hierarchy of models is consistent across KGs and we can clearly see that KGEMs are grouped by families of models.

Our experimental results suggest that the structure of GNNs seems to be able to encode the latent semantics of the entities and relations of the graph. This ability may be attributed to the fact that, contrary to translational models which only model the local neighborhood of each triple, GNNs update entity embeddings (and relation embeddings in the case of CompGCN) based on the information found in the extended, h-hop neighborhood of the central node. While translational and semantic matching models treat each triple independently, GNNs model interactions between entities on a large range of semantic relations. It is likely that this extended neighborhood comprises signals or patterns that help the model infer the classes of entities, thus providing very promising semantic capabilities in all experimental conditions.

### 6.2 Dynamic appraisal of KGEM semantic awareness

For certain models, rank-based metrics performance and semantic capabilities improve jointly. For others, the enhancement of their performance in terms of rank-based metrics comes at the expense of their semantic awareness. Interestingly, trends emerge relatively to families of models. First, we observe that a trade-off exists for semantic matching models. Results are particularly striking on FB15K237-ET (see Fig. 8), where it is obvious that after reaching the best Sem@\(K\) values after a few epochs, Sem@\(K\) values of DistMult and ComplEx quickly drop while MRR continues rising. Conversely, translational models are more robust to Sem@\(K\) degradation throughout the epochs. Even though the best achieved Sem@\(K\) values are also reported in the very first epochs, once these values are reached they remain stable for the remaining epochs of training. This might be due to the geometric nature of such KGEMs, which will organize the representation space so as to \(h + r\) falls in a region of the space where neighboring entities of the ground-truth tail \(t\) all are entities of the expected type. This is highly related to the block structure property, which is a common statistical pattern found in KGs Nickel et al. [2016a]. It refers to the fact that entities can naturally belong to different groups (blocks), such that all the entities of a given group are linked to entities of another group through the same relationship. In this case, each group comprises entities of the same class. Translational models will naturally group entities of the same class in the same region of the representation space, as this is determined by the translation vector in that space.
Plots of the joint evolution of MRR and \( \text{Sem}@K \) values show that most of the KGEMs reaches their best \( \text{Sem}@K \) values after a few number of epochs. This means that predictions get semantically valid in the early stages of training. As previously mentioned, \( \text{Sem}@K \) then usually start to decrease, as it has been noted for semantic matching models in particular. To this respect, an excerpt of the head and tail predictions of DistMult on YAGO3-37K is depicted in Fig. 9. Even though the ground-truth entity does not show up in neither the head nor the tail top-K list, we clearly see that after only 30 epochs of training, predictions made by DistMult are more meaningful than after 400 epochs of training. This relative trade-off between making semantically valid predictions and predictions that comprise the ground-truth entity higher in the top-K list calls for finding a compromise in terms of training. The LP task is usually addressed in terms of rank-based metrics only, hence the choice of performing more and more training epochs so as to find the optimal KGEM in terms of MRR and Hits@K. However, as discussed in the present work, adding training steps may improve KGEM performance at the expense of its semantic awareness. In many cases, rank-based metrics values only slightly increase, whereas \( \text{Sem}@K \) values drastically drop. For instance, comparing MRR vs. \( \text{Sem}@K \) evolution of DistMult on FB15K237-ET (Fig. 10c), we clearly see that after a moderate number of epochs, any additional epoch of training only provides a very slight improvement in terms of MRR, while it is very detrimental to \( \text{Sem}@K \) values. Depending on the use case, such a decline in the semantic capabilities of the model is not desirable, and a compromise is to be found between training more to increase KGEM predictive performance and stopping training early enough so as not to deteriorate its semantic awareness too much.

6.3 On the use of \( \text{Sem}@K \) for different kinds of KGs

As reported in Table 1, KGs based upon a schema and a class hierarchy are candidates for the computation of all the versions of \( \text{Sem}@K \). For the schema-defined KGs used in our experiments, we choose to report values regarding all these metrics so as to enable multi-view comparisons across models. From Table 8 it can be clearly seen that the relative superiority of models is consistent throughout the different \( \text{Sem}@K \) definitions. From a higher perspective, this means that even for schema-defined KGs with a hierarchy class, \( \text{Sem}@K[^{ext}] \) is already a good proxy. This may be a good option to only rely on the \( \text{Sem}@K[^{ext}] \) whenever the computation of \( \text{Sem}@K[^{base}] \) is too expensive, due to the entity type checking part. This is even more true for \( \text{Sem}@K[^{wup}] \), which requires an additional step of semantic relatedness computation.
Sem@K: Is my KGEM semantic-aware?

Figure 9: Top-ten ranked entities for head and tail predictions at epochs 30 and 400 for a sample triple from YAGO3-37K. Green, blue and white cells respectively denote the ground-truth entity, entities other than the ground-truth and semantically valid, and entities other than the ground-truth and semantically invalid. In this case, semantic validity is based on the domain and range of the relation <hasCapital>.

7 Discussion

Three major research questions have been formulated in Section 1. Based on the analysis presented in Section 6, we discuss each research question individually. We ultimately discuss the potential for further considerations of semantics into KGEMs.

7.1 RQ1: how semantic-aware agnostic KGEMs are?

From a coarse-grained viewpoint, we noted that KGEMs trained in an agnostic way prove capable of giving higher scores to semantically valid triples. However, disparities exist between models. Interestingly, these disparities seem to derive from the family of such models. Globally, translational models and GNNs – represented by R-GCN and CompGCN in this work – provide promising results. It appears that the two aforementioned families of KGEMs are better able than semantic matching models (DistMult, ComplEx, SimplE) at recovering the semantics of entities and relations to give higher score to semantically valid triples. In fact, semantic matching models are almost systematically the worst performing models in terms of semantic awareness. From a dynamic standpoint, it is worth noting the high semantic capabilities of KGEMs reached during the first epochs of training. In most cases, this is even during the first epochs that the optimal semantic awareness is attained.

7.2 RQ2: how KGEM semantic awareness’ evaluation should adapt to the typology of KGs?

Drawing on the initial version of Sem@K as presented in Hubert et al. [2022b] – referred to herein as Sem@K[base] – an issue is quickly encountered when it comes to schemaless KGs, which do not contain any rdfs:domain (resp. rdfs:range) clause to indicate the class that candidate heads (resp. tails) should belong to. Our work introduces Sem@K[ext] – a new version of Sem@K that overcomes the aforementioned limitation. In addition, even with schema-defined KGs, Sem@K[base] is not necessarily sufficient in itself. This metric can be further enriched whenever a KG comes with a class hierarchy. In Section 4.4, we integrate class hierarchy into Sem@K by means of a similarity measure between concepts. We subsequently provide an example using the Wu-Palmer similarity score. The resulting
Sem@\(K\) is used in the experiments in Section 6 and provide a finer-grained measure of KGEMs semantic awareness.

### 7.3 RQ3: does the evaluation of KGEM semantic awareness offers different conclusions on the relative superiority of some KGEMs?

A major finding is that models performing well with respect to rank-based metrics are not necessarily the most competitive regarding their semantic capabilities. We previously noted that translational models globally showcase better Sem@\(K\) values compared to semantic matching models. Considering MRR and Hits@\(K\), the opposite conclusion is often drawn. Hence, the performance of KGEMs in terms of rank-based metrics is not indicative of their semantic capabilities. The only exception that might exist is for GNNs that perform well both in terms of rank-based metrics and semantic-oriented measures.

The answers provided to the research questions also lead to consider new matters. As evidenced in Table 8 and Table 9, some KGs are more challenging with regard to Sem@\(K\) results. Due to its tailored extraction strategy that purposely favored difficult relations to feature in the validation and test sets, YAGO4-19K is the schema-defined KG with the lowest achieved Sem@\(K\). This observation raises a deeper question: what characteristics of a KG make it inherently challenging for KGEMs to recover the semantics of entities and relations? An extensive study of the influence of KGs characteristics on the semantic capabilities of KGEMs would require to benchmark them on a broad set of KGs with varying dimensions, so as to determine those that are the most prevalent. Such characteristics can be the total number of relations, the average number of instances per class, or a combination of different factors. We leave this experimental study for future work.

Recall that this work is motivated by the possibility of going beyond a mere assessment of KGEM performance regarding rank-based metrics. We showed that these metrics only evaluate one aspect of such models, somehow providing a partial view on the quality of KGEMs. Our proposal for further assessing KGEM semantic capabilities aims at diving deeper into their predictive expressiveness and measuring to what extent their predictions are semantically valid. However, this second evaluation component does not shed full light on the respective KGEM peculiarities. Other evaluation components may be added, such as the storage and computational requirements of KGEMs Wang et al. [2021], Portisch et al. [2020] and the environmental impact of their training and testing procedure Peng et al. [2021]. Furthermore, the explainability of KGEMs is another dimension that deserves great attention Zhang et al. [2020b], Rossi et al. [2022].

### 7.4 Towards further considerations of semantics in knowledge graph embeddings models

The Sem@\(K\) metric presented in this work allows for a more comprehensive evaluation of KGEMs. Based on domains and ranges of relations, Sem@\(K\) assesses to what extent the predictions of a model are semantically valid. The present work constitutes one of several stepping stones toward the further consideration of ontological and semantic information in KGEM design and evaluation.

It should be noted that due to the only consideration of domains and ranges, Sem@\(K\) cannot indicate whether predictions are logically consistent with other constraints posed by the ontology. This is in contrast with Inc@\(K\) presented in Jain et al. [2021a] that takes a broader set of ontological axioms into account. However, Inc@\(K\) and Sem@\(K\) intrinsically assess distinct dimensions of predictions. While the former is concerned with the logical consistency of predictions, the latter focuses on whether these predictions are semantically valid. For instance, an ontology can specify that City is the range of livesIn, that Seattle is a City but not a Capital, and that entities of type President should be linked to a Capital through the relation livesIn. Hence, it would still be meaningful and semantically correct to predict that (BarackObama,livesIn,Seattle). However, this prediction is not logically consistent w.r.t. ontology specifications. Inc@\(K\) and Sem@\(K\) thus consider triples at different levels: while a given triple can be meaningful and semantically correct on its own, its combination with other triples may not be semantically valid or consistent with the ontology. In future work, we will consider more expressive ontologies and see how the broad collection of axioms that constitute them can be incorporated into Sem@\(K\).

KGEMs evaluated in this work are all agnostic to ontological information in their design. However, some models that consider or ensure specific ontological or logical properties exist. For example, HAKE Zhang et al. [2020a] is constructed with the purpose of preserving hierarchies, Logic Tensor Networks Badreddine et al. [2022] are designed to ensure logical reasoning, and the training of TransOWL and TransROWL d’Amato et al. [2021] is enriched with additional triples deduced from, e.g., inverse predicates or equivalent classes. Because of the integration of semantic information in their design or training, one could wonder if they present improved semantic awareness compared to agnostic models. Additionally, KGEMs can also be used to predict triples that represent class instantiations. A possible extension of the present work thus consists in studying whether predicted links and class instantiations are
consistent and lead to increased Sem@\(K\) values. This would further qualify and highlight the semantic awareness and the consistency of predictions of KGEMs. We leave these questions for future work.

8 Conclusion

In this work, we consider the link prediction task and extend our previously introduced Sem@\(K\) metric to measure the ability of KGEMs to assign higher scores to triples that are semantically valid. In particular, to adapt to different types of KGs (e.g., schemaless, class hierarchy), we introduce Sem@\(K\)\[base\], Sem@\(K\)\[ext\], or Sem@\(K\)\[wup\]. Compared with the traditional evaluation approach that solely relies on rank-based metrics, we show that the evaluation procedure is enhanced with the addition of semantic-oriented metrics that bring an additional perspective on KGEM quality. Our experiments with different types of KGs highlight that there is no clear correlation between the performance of KGEMs in terms of traditional rank-based metrics versus their performance regarding semantic-oriented ones. In some cases, however, a trade-off does exist. Consequently, this calls for monitoring KGEM training under more scrutiny. Our experiments also point out that most of the conclusions that have been drawn actually hold at the level of families of models.

In future work, we will conduct experiments considering a broader array of KGEM families (e.g., KGEMs that include semantics) and propose evaluation metrics that consider additional and more expressive ontological constraints.

Acknowledgements

This work is supported by the AILES PIA3 project (see https://www.projetailes.com/). Experiments presented in this paper were carried out using the Grid’5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000.fr).

A Hyperparameters

For datasets with no reported optimal hyperparameters, grid-search based on curated hyperparameters were performed. The full hyperparameter space is provided in Table 6. Chosen hyperparameters for each pair of KGEM and dataset are provided in Table 7.

Table 6: Hyperparameter search space

<table>
<thead>
<tr>
<th>Hyperparameters</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch Size</td>
<td>{128, 256, 512, 1024, 2048}</td>
</tr>
<tr>
<td>Embedding Dimension</td>
<td>{50, 100, 150, 200}</td>
</tr>
<tr>
<td>Regularizer Type</td>
<td>{None, L1, L2}</td>
</tr>
<tr>
<td>Regularizer Weight ((\lambda))</td>
<td>{1e(^{-1}), 1e(^{-2}), 1e(^{-3}), 1e(^{-4}), 1e(^{-5})}</td>
</tr>
<tr>
<td>Learning Rate ((lr))</td>
<td>{0.005, 0.003, 0.001, 0.0005, 0.0003, 0.0001}</td>
</tr>
</tbody>
</table>

B Results achieved with the best reported hyperparameters

Results achieved with the best reported hyperparameters are presented in Tables 8 and 9.

C Evolution of MRR and Sem@\(K\) values with respect to the number of epochs

The evolution of MRR and Sem@\(K\) values with respect to the number of epochs is presented in Fig. 10, 11, 12, 13, 14, 15, and 16. For equity and clarity sakes, we choose to present 2 KGEMs for each family of model (translational models, semantic matching models, CNNs, and GNNs). Regarding semantic matching models, DistMult and ComplEx are chosen, as the evolution of their MRR and Sem@\(K\) values is less erratic than SimplE. The evolution of MRR and Sem@\(K\) values for SimplE are made available on the GitHub repository of the datasets\(^{11}\).

\(^{11}\)https://github.com/nicolas-hbt/benchmark-sematk
Table 7: Chosen hyperparameters for schema-defined and schemaless KGs used in the experiments. $|\mathcal{B}|$, $d$, $lr$, and $\lambda$ denote the batch size, embedding dimension, learning rate, and regularization weight, respectively. We experimentally found that $L2$ regularizer systematically worked the best. We therefore decide not to refer to it in the table. For ConvKB, the number of filters $\mathcal{T}$ was set to 128. Parameters specific to R-GCN and CompGCN were left as defined in the original implementations.

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>FB15K237-ET</th>
<th>DB93K</th>
<th>YAGO3-37K</th>
<th>YAGO4-19K</th>
<th>Codex-S</th>
<th>Codex-M</th>
<th>WN18RR</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TransE</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$</td>
<td>\mathcal{B}</td>
<td>$</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>$d$</td>
<td>100</td>
<td>200</td>
<td>150</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>$lr$</td>
<td>$10^{-4}$</td>
<td>$10^{-3}$</td>
<td>$5 \times 10^{-3}$</td>
<td>$10^{-3}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
</tr>
<tr>
<td><strong>TransH</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$</td>
<td>\mathcal{B}</td>
<td>$</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>$d$</td>
<td>100</td>
<td>200</td>
<td>150</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>$lr$</td>
<td>$10^{-4}$</td>
<td>$10^{-3}$</td>
<td>$5 \times 10^{-3}$</td>
<td>$10^{-3}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
</tr>
<tr>
<td><strong>DistMult</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$</td>
<td>\mathcal{B}</td>
<td>$</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>$d$</td>
<td>100</td>
<td>200</td>
<td>150</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>$lr$</td>
<td>$10^{-4}$</td>
<td>$10^{-3}$</td>
<td>$5 \times 10^{-3}$</td>
<td>$10^{-3}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
</tr>
<tr>
<td><strong>ComplEx</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$</td>
<td>\mathcal{B}</td>
<td>$</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>$d$</td>
<td>100</td>
<td>200</td>
<td>150</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>$lr$</td>
<td>$10^{-4}$</td>
<td>$10^{-3}$</td>
<td>$5 \times 10^{-3}$</td>
<td>$10^{-3}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
</tr>
<tr>
<td><strong>SimplE</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$</td>
<td>\mathcal{B}</td>
<td>$</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>$d$</td>
<td>100</td>
<td>200</td>
<td>150</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>$lr$</td>
<td>$10^{-4}$</td>
<td>$10^{-3}$</td>
<td>$5 \times 10^{-3}$</td>
<td>$10^{-3}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
</tr>
<tr>
<td><strong>ConvE</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$</td>
<td>\mathcal{B}</td>
<td>$</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>$d$</td>
<td>100</td>
<td>200</td>
<td>150</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>$lr$</td>
<td>$10^{-4}$</td>
<td>$10^{-3}$</td>
<td>$5 \times 10^{-3}$</td>
<td>$10^{-3}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
</tr>
<tr>
<td><strong>ConvKB</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$</td>
<td>\mathcal{B}</td>
<td>$</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>$d$</td>
<td>100</td>
<td>200</td>
<td>150</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>$lr$</td>
<td>$10^{-4}$</td>
<td>$10^{-3}$</td>
<td>$5 \times 10^{-3}$</td>
<td>$10^{-3}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
</tr>
<tr>
<td><strong>R-GCN</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$</td>
<td>\mathcal{B}</td>
<td>$</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>$d$</td>
<td>100</td>
<td>200</td>
<td>150</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>$lr$</td>
<td>$10^{-4}$</td>
<td>$10^{-3}$</td>
<td>$5 \times 10^{-3}$</td>
<td>$10^{-3}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
<td>$10^{-5}$</td>
</tr>
<tr>
<td><strong>CompGCN</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$</td>
<td>\mathcal{B}</td>
<td>$</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>$d$</td>
<td>100</td>
<td>200</td>
<td>150</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>$lr$</td>
<td>$10^{-4}$</td>
<td>$10^{-3}$</td>
<td>$5 \times 10^{-3}$</td>
<td>$10^{-3}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
<td>$10^{-2}$</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>$0$</td>
<td>$0$</td>
<td>$0$</td>
<td>$0$</td>
<td>$0$</td>
<td>$0$</td>
<td>$0$</td>
</tr>
</tbody>
</table>
Table 8: Rank-based and semantic-based results on the schema-defined knowledge graphs. Bold fonts indicate which model performs best with respect to a given metric.

(a) FB15K237-ET

<table>
<thead>
<tr>
<th>Model Family</th>
<th>Model</th>
<th>MRR</th>
<th>Rank-based H@1</th>
<th>H@3</th>
<th>H@10</th>
<th>Sem@K [base]</th>
<th>Sem@K [wup]</th>
<th>Sem@K [ext]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geometric</td>
<td>TransE</td>
<td>.273</td>
<td>.182</td>
<td>.302</td>
<td>.453</td>
<td>.978</td>
<td>.964</td>
<td>.949</td>
</tr>
<tr>
<td></td>
<td>TransH</td>
<td>.281</td>
<td>.187</td>
<td>.306</td>
<td>.458</td>
<td>.981</td>
<td>.968</td>
<td>.955</td>
</tr>
<tr>
<td>Semantic Matching</td>
<td>DistMult</td>
<td>.289</td>
<td>.206</td>
<td>.313</td>
<td>.456</td>
<td>.907</td>
<td>.919</td>
<td>.922</td>
</tr>
<tr>
<td></td>
<td>ComplEx</td>
<td>.267</td>
<td>.185</td>
<td>.292</td>
<td>.431</td>
<td>.874</td>
<td>.823</td>
<td>.761</td>
</tr>
<tr>
<td></td>
<td>SimpE</td>
<td>.257</td>
<td>.180</td>
<td>.275</td>
<td>.416</td>
<td>.895</td>
<td>.878</td>
<td>.848</td>
</tr>
<tr>
<td>Convolutional</td>
<td>ConvE</td>
<td>.310</td>
<td>.211</td>
<td>.337</td>
<td>.490</td>
<td>.972</td>
<td>.970</td>
<td>.961</td>
</tr>
<tr>
<td></td>
<td>ConvKB</td>
<td>.251</td>
<td>.159</td>
<td>.261</td>
<td>.411</td>
<td>.918</td>
<td>.897</td>
<td>.872</td>
</tr>
<tr>
<td></td>
<td>R-GCN</td>
<td>.238</td>
<td>.153</td>
<td>.257</td>
<td>.412</td>
<td>.980</td>
<td>.970</td>
<td>.953</td>
</tr>
<tr>
<td></td>
<td>CompGCN</td>
<td>.337</td>
<td>.227</td>
<td>.365</td>
<td>.530</td>
<td>.998</td>
<td>.996</td>
<td>.992</td>
</tr>
</tbody>
</table>

(b) DB93K

<table>
<thead>
<tr>
<th>Model Family</th>
<th>Model</th>
<th>MRR</th>
<th>Rank-based H@1</th>
<th>H@3</th>
<th>H@10</th>
<th>Sem@K [base]</th>
<th>Sem@K [wup]</th>
<th>Sem@K [ext]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geometric</td>
<td>TransE</td>
<td>.233</td>
<td>.145</td>
<td>.275</td>
<td>.397</td>
<td>.985</td>
<td>.975</td>
<td>.961</td>
</tr>
<tr>
<td></td>
<td>TransH</td>
<td>.236</td>
<td>.147</td>
<td>.278</td>
<td>.399</td>
<td>.988</td>
<td>.979</td>
<td>.968</td>
</tr>
<tr>
<td></td>
<td>ComplEx</td>
<td>.287</td>
<td>.213</td>
<td>.325</td>
<td>.417</td>
<td>.941</td>
<td>.917</td>
<td>.877</td>
</tr>
<tr>
<td>Convolutional</td>
<td>ConvE</td>
<td>.256</td>
<td>.183</td>
<td>.289</td>
<td>.392</td>
<td>.871</td>
<td>.862</td>
<td>.862</td>
</tr>
<tr>
<td></td>
<td>ConvKB</td>
<td>.178</td>
<td>.121</td>
<td>.199</td>
<td>.283</td>
<td>.883</td>
<td>.895</td>
<td>.881</td>
</tr>
<tr>
<td></td>
<td>R-GCN</td>
<td>.208</td>
<td>.144</td>
<td>.233</td>
<td>.319</td>
<td>.969</td>
<td>.959</td>
<td>.945</td>
</tr>
<tr>
<td></td>
<td>CompGCN</td>
<td>.319</td>
<td>.249</td>
<td>.351</td>
<td>.446</td>
<td>.988</td>
<td>.976</td>
<td>.967</td>
</tr>
</tbody>
</table>

(c) YAGO3-37K

<table>
<thead>
<tr>
<th>Model Family</th>
<th>Model</th>
<th>MRR</th>
<th>Rank-based H@1</th>
<th>H@3</th>
<th>H@10</th>
<th>Sem@K [base]</th>
<th>Sem@K [wup]</th>
<th>Sem@K [ext]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geometric</td>
<td>TransE</td>
<td>.184</td>
<td>.080</td>
<td>.198</td>
<td>.408</td>
<td>.989</td>
<td>.988</td>
<td>.988</td>
</tr>
<tr>
<td></td>
<td>TransH</td>
<td>.187</td>
<td>.091</td>
<td>.199</td>
<td>.415</td>
<td>.995</td>
<td>.993</td>
<td>.993</td>
</tr>
<tr>
<td>Semantic Matching</td>
<td>DistMult</td>
<td>.225</td>
<td>.112</td>
<td>.251</td>
<td>.465</td>
<td>.885</td>
<td>.940</td>
<td>.959</td>
</tr>
<tr>
<td></td>
<td>ComplEx</td>
<td>.430</td>
<td>.250</td>
<td>.551</td>
<td>.780</td>
<td>.740</td>
<td>.820</td>
<td>.859</td>
</tr>
<tr>
<td></td>
<td>SimpE</td>
<td>.311</td>
<td>.081</td>
<td>.468</td>
<td>.784</td>
<td>.377</td>
<td>.733</td>
<td>.866</td>
</tr>
<tr>
<td>Convolutional</td>
<td>ConvE</td>
<td>.493</td>
<td>.350</td>
<td>.578</td>
<td>.775</td>
<td>.933</td>
<td>.923</td>
<td>.910</td>
</tr>
<tr>
<td></td>
<td>ConvKB</td>
<td>.305</td>
<td>.162</td>
<td>.352</td>
<td>.631</td>
<td>.979</td>
<td>.979</td>
<td>.978</td>
</tr>
<tr>
<td></td>
<td>R-GCN</td>
<td>.115</td>
<td>.046</td>
<td>.110</td>
<td>.254</td>
<td>.993</td>
<td>.994</td>
<td>.993</td>
</tr>
<tr>
<td></td>
<td>CompGCN</td>
<td>.399</td>
<td>.269</td>
<td>.646</td>
<td>.663</td>
<td>.998</td>
<td>.997</td>
<td>.996</td>
</tr>
</tbody>
</table>

(d) YAGO4-19K

<table>
<thead>
<tr>
<th>Model Family</th>
<th>Model</th>
<th>MRR</th>
<th>Rank-based H@1</th>
<th>H@3</th>
<th>H@10</th>
<th>Sem@K [base]</th>
<th>Sem@K [wup]</th>
<th>Sem@K [ext]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geometric</td>
<td>TransE</td>
<td>.749</td>
<td>.651</td>
<td>.833</td>
<td>.901</td>
<td>.975</td>
<td>.890</td>
<td>.843</td>
</tr>
<tr>
<td></td>
<td>TransH</td>
<td>.752</td>
<td>.656</td>
<td>.829</td>
<td>.898</td>
<td>.979</td>
<td>.909</td>
<td>.868</td>
</tr>
<tr>
<td>Semantic Matching</td>
<td>DistMult</td>
<td>.863</td>
<td>.839</td>
<td>.881</td>
<td>.898</td>
<td>.931</td>
<td>.687</td>
<td>.534</td>
</tr>
<tr>
<td></td>
<td>ComplEx</td>
<td>.890</td>
<td>.881</td>
<td>.897</td>
<td>.903</td>
<td>.923</td>
<td>.627</td>
<td>.479</td>
</tr>
<tr>
<td></td>
<td>SimpE</td>
<td>.808</td>
<td>.757</td>
<td>.848</td>
<td>.883</td>
<td>.891</td>
<td>.646</td>
<td>.500</td>
</tr>
<tr>
<td>Convolutional</td>
<td>ConvE</td>
<td>.901</td>
<td>.891</td>
<td>.908</td>
<td>.913</td>
<td>.977</td>
<td>.858</td>
<td>.844</td>
</tr>
<tr>
<td></td>
<td>ConvKB</td>
<td>.828</td>
<td>.772</td>
<td>.883</td>
<td>.908</td>
<td>.865</td>
<td>.614</td>
<td>.564</td>
</tr>
<tr>
<td></td>
<td>R-GCN</td>
<td>.893</td>
<td>.885</td>
<td>.896</td>
<td>.903</td>
<td>.969</td>
<td>.832</td>
<td>.727</td>
</tr>
<tr>
<td></td>
<td>CompGCN</td>
<td>.907</td>
<td>.903</td>
<td>.908</td>
<td>.918</td>
<td>.980</td>
<td>.906</td>
<td>.862</td>
</tr>
</tbody>
</table>

23
Table 9: Rank-based and semantic-based results on the schemaless knowledge graphs. Bold fonts indicate which model performs best with respect to a given metric.

(a) Codex-S

<table>
<thead>
<tr>
<th>Model Family</th>
<th>Model</th>
<th>MRR</th>
<th>H@1</th>
<th>H@3</th>
<th>H@10</th>
<th>S@1</th>
<th>S@3</th>
<th>S@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geometric</td>
<td>TransE</td>
<td>.354</td>
<td>.223</td>
<td>.409</td>
<td>.620</td>
<td>.927</td>
<td>.900</td>
<td>.873</td>
</tr>
<tr>
<td></td>
<td>TransH</td>
<td>.355</td>
<td>.225</td>
<td>.410</td>
<td>.621</td>
<td>.928</td>
<td>.901</td>
<td>.875</td>
</tr>
<tr>
<td>Semantic Matching</td>
<td>DistMult</td>
<td>.426</td>
<td>.552</td>
<td>.668</td>
<td>.872</td>
<td>.786</td>
<td>.694</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ComplEx</td>
<td>.511</td>
<td>.223</td>
<td>.384</td>
<td>.556</td>
<td>.936</td>
<td>.921</td>
<td>.889</td>
</tr>
<tr>
<td></td>
<td>SimplE</td>
<td>.464</td>
<td>.268</td>
<td>.427</td>
<td>.597</td>
<td>.993</td>
<td>.986</td>
<td>.974</td>
</tr>
<tr>
<td>Convolutional</td>
<td>ConvE</td>
<td>.379</td>
<td>.259</td>
<td>.439</td>
<td>.610</td>
<td>.912</td>
<td>.913</td>
<td>.886</td>
</tr>
<tr>
<td></td>
<td>ConvKB</td>
<td>.453</td>
<td>.361</td>
<td>.493</td>
<td>.632</td>
<td>.858</td>
<td>.806</td>
<td>.769</td>
</tr>
<tr>
<td></td>
<td>R-GCN</td>
<td>.335</td>
<td>.225</td>
<td>.381</td>
<td>.556</td>
<td>.936</td>
<td>.921</td>
<td>.889</td>
</tr>
<tr>
<td></td>
<td>CompGCN</td>
<td>.380</td>
<td>.268</td>
<td>.427</td>
<td>.597</td>
<td>.993</td>
<td>.986</td>
<td>.974</td>
</tr>
</tbody>
</table>

(b) Codex-M

<table>
<thead>
<tr>
<th>Model Family</th>
<th>Model</th>
<th>MRR</th>
<th>H@1</th>
<th>H@3</th>
<th>H@10</th>
<th>S@1</th>
<th>S@3</th>
<th>S@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geometric</td>
<td>TransE</td>
<td>.262</td>
<td>.188</td>
<td>.284</td>
<td>.405</td>
<td>.889</td>
<td>.855</td>
<td>.839</td>
</tr>
<tr>
<td></td>
<td>TransH</td>
<td>.266</td>
<td>.190</td>
<td>.291</td>
<td>.415</td>
<td>.894</td>
<td>.862</td>
<td>.842</td>
</tr>
<tr>
<td>Semantic Matching</td>
<td>DistMult</td>
<td>.274</td>
<td>.213</td>
<td>.298</td>
<td>.386</td>
<td>.768</td>
<td>.687</td>
<td>.608</td>
</tr>
<tr>
<td></td>
<td>ComplEx</td>
<td>.269</td>
<td>.212</td>
<td>.289</td>
<td>.378</td>
<td>.799</td>
<td>.713</td>
<td>.629</td>
</tr>
<tr>
<td></td>
<td>SimplE</td>
<td>.269</td>
<td>.212</td>
<td>.289</td>
<td>.378</td>
<td>.799</td>
<td>.713</td>
<td>.629</td>
</tr>
<tr>
<td>Convolutional</td>
<td>ConvE</td>
<td>.255</td>
<td>.184</td>
<td>.280</td>
<td>.395</td>
<td>.863</td>
<td>.839</td>
<td>.814</td>
</tr>
<tr>
<td></td>
<td>ConvKB</td>
<td>.230</td>
<td>.156</td>
<td>.250</td>
<td>.376</td>
<td>.797</td>
<td>.771</td>
<td>.770</td>
</tr>
<tr>
<td></td>
<td>R-GCN</td>
<td>.185</td>
<td>.110</td>
<td>.201</td>
<td>.340</td>
<td>.944</td>
<td>.920</td>
<td>.887</td>
</tr>
<tr>
<td></td>
<td>CompGCN</td>
<td>.314</td>
<td>.237</td>
<td>.346</td>
<td>.460</td>
<td>.996</td>
<td>.992</td>
<td>.983</td>
</tr>
</tbody>
</table>

(c) WN18RR

<table>
<thead>
<tr>
<th>Model Family</th>
<th>Model</th>
<th>MRR</th>
<th>H@1</th>
<th>H@3</th>
<th>H@10</th>
<th>S@1</th>
<th>S@3</th>
<th>S@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geometric</td>
<td>TransE</td>
<td>.186</td>
<td>.032</td>
<td>.303</td>
<td>.455</td>
<td>.770</td>
<td>.756</td>
<td>.715</td>
</tr>
<tr>
<td></td>
<td>TransH</td>
<td>.191</td>
<td>.034</td>
<td>.306</td>
<td>.458</td>
<td>.773</td>
<td>.760</td>
<td>.717</td>
</tr>
<tr>
<td>Semantic Matching</td>
<td>DistMult</td>
<td>.399</td>
<td>.372</td>
<td>.413</td>
<td>.444</td>
<td>.795</td>
<td>.732</td>
<td>.688</td>
</tr>
<tr>
<td></td>
<td>ComplEx</td>
<td>.430</td>
<td>.400</td>
<td>.446</td>
<td>.487</td>
<td>.715</td>
<td>.669</td>
<td>.673</td>
</tr>
<tr>
<td></td>
<td>SimplE</td>
<td>.397</td>
<td>.375</td>
<td>.406</td>
<td>.438</td>
<td>.690</td>
<td>.598</td>
<td>.673</td>
</tr>
<tr>
<td>Convolutional</td>
<td>ConvE</td>
<td>.406</td>
<td>.375</td>
<td>.422</td>
<td>.459</td>
<td>.865</td>
<td>.854</td>
<td>.871</td>
</tr>
<tr>
<td></td>
<td>ConvKB</td>
<td>.356</td>
<td>.302</td>
<td>.391</td>
<td>.444</td>
<td>.712</td>
<td>.713</td>
<td>.732</td>
</tr>
<tr>
<td></td>
<td>R-GCN</td>
<td>.382</td>
<td>.345</td>
<td>.402</td>
<td>.441</td>
<td>.836</td>
<td>.797</td>
<td>.752</td>
</tr>
<tr>
<td></td>
<td>CompGCN</td>
<td>.471</td>
<td>.437</td>
<td>.483</td>
<td>.536</td>
<td>.970</td>
<td>.946</td>
<td>.927</td>
</tr>
</tbody>
</table>
Figure 10: Evolution of MRR (---), Sem@1 (--), Sem@3 (- - -), and Sem@10 (-----) on FB15K237-ET
Figure 11: Evolution of MRR (—), Sem@1 (•••), Sem@3 (••••), and Sem@10 (•••••) on DB93K
Figure 12: Evolution of MRR (—), Sem@1 (••), Sem@3 (•••), and Sem@10 (⋯) on YAGO3-37K
Figure 13: Evolution of MRR (—), Sem@1 (••), Sem@3 (•••), and Sem@10 (••••) on YAGO4-19K
Figure 14: Evolution of MRR (---), Sem@1 (---), Sem@3 (----), and Sem@10 (-----) on Codex-S
Figure 15: Evolution of MRR (-----), Sem@1 (---), Sem@3 (--), and Sem@10 (-----) on Codex-M
Figure 16: Evolution of MRR (---), Sem@1 (--), Sem@3 (----), and Sem@10 (-----) on WN18RR
Sem@K: Is my KGEM semantic-aware?

References


Nicolas Hubert, Pierre Monnin, Armelle Brun, and Davy Monticolo. Knowledge Graph Embeddings for Link Prediction: Beware of Semantics! In DL4KG@ISWC 2022: Workshop on Deep Learning for Knowledge Graphs, held as part of ISWC 2022: the 21st International Semantic Web Conference, Virtual, China, October 2022b.


Sem@K: Is my KGEM semantic-aware?


Ilaria Ferrari, Giacomo Frisoni, Paolo Italiani, Gianluca Moro, and Claudio Sartori. Comprehensive analysis of knowledge graph embedding techniques benchmarked on link prediction. Electronics, 11(23), 2022. ISSN 2079-9292.


35
Sem@K: Is my KGEM semantic-aware?


