An Ontology-Based Deep Learning Approach for Triple Classification with Out-of-Knowledge-Base Entities

Elvira Amador-Domínguez[a], Emilio Serrano[a,*], Daniel Manrique[b], Patrick Hohenecker[c], Thomas Lukasiewicz[c]

[a] *Ontology Engineering Group, Department of Artificial Intelligence, ETSI Informáticos, Universidad Politécnica de Madrid, 28660 Madrid, Spain*
[b] *Artificial Intelligence Lab, Department of Artificial Intelligence, ETSI Informáticos, Universidad Politécnica de Madrid, 28660 Madrid, Spain*
[c] *Department of Computer Science, University of Oxford, Oxford, OX1 3QD, UK*

Abstract

Knowledge graphs (KGs) are one of the most common frameworks for knowledge representation. However, they suffer from a severe scalability problem that hinders their usage. KG embedding aims to provide a solution to this issue. Nonetheless, general approaches are incapable of representing and reasoning about information not previously contained in the graph. This paper proposes to leverage semantic and ontological information for a significant benefit of knowledge graph completion, focusing on triple classification. The goal of this task is to determine whether a given fact holds. Furthermore, this paper also considers the classification of facts that include entities that have not been seen during training, denoted out-of-knowledge-base or *OOKB* entities. An incremental method is presented, composed of six stages. Although the proposal can be applied to any KG embedding model, this work focuses on its application for semantic matching models, such as ComplEx and DistMult. Compared to other approaches, our proposal is model-agnostic, computationally inexpensive, and does not require retraining. The results show that triple classification accuracy scales up to 15% with the proposed approach, as well as accelerating the convergence of the model to its optimal solution. Furthermore, facts containing OOKB entities can be classified with a reasonable accuracy.

Key words: Knowledge Graph Embeddings, Entity Initialization, Knowledge Graph Completion, Word Embeddings, Ontological Information

1. Introduction

Knowledge graphs (KGs) are currently one of the primary forms of knowledge representation. In these structures, knowledge is stored in the form of facts, or

*Corresponding author
Email address: emilioserra@fi.upm.es (Emilio Serrano)

triples, that follow the schema $(head, relation, tail)$, modelling the interactions between entities. KGs are proliferating over time, requiring methods capable of synthesizing the present information to perform certain operations, such as question answering or recommendation in an efficient and accurate way. *KG embeddings (KGEs)* provide a solution to this problem, converting the explicit information contained in the KG into vector (or matrix) representations.

Ontologies are explicit specifications of conceptualizations, which are shared and understood between humans and machines [20]. Among others, ontologies present a formal definition of classes (or types), properties, and relationships between entities in a domain. Therefore, unlike the facts encoded in KGs, ontologies provide general and static knowledge.

KGs possess a high degree of incompleteness, which severely hinders their usage. Furthermore, adding knowledge to KGs is usually performed manually, which noticeably limits their growth. *KG completion* provides an automatic solution to this problem. In KG completion, the goal is to find potential new feasible facts from the knowledge in the KG. KG embeddings are at the base of this task, as they provide expressive arithmetical representations of the KG that enable prediction and decision making over the graph. Given an incomplete fact $(h, r, ?)$, a KG embedding can be used to automatically detect the missing element that maximizes its feasibility according to a given scoring function, thus generating new facts previously not existing in the graph.

However, current KG embedding techniques for KG completion present two main shortcomings: (1) they do not leverage ontological information, and (2) they cannot reason over facts that include entities that were not previously in the graph during training. Although implicit pseudo-ontological information can usually be obtained from the existing facts, it is generally treated the same as the rest of the information contained within the KG. Consequently, it is not shared across similar entities. As for unseen or *out-of-knowledge-base* (*OOKB*) entities, no operation can be performed, as most of the current KG embedding techniques do not support the representation of entities that are not seen during training.

This work presents an initialization method to enhance KG embeddings introducing semantic and ontological information, enabling the representation of OOKB entities. Besides, the proposed approach is model-agnostic, meaning that it can be easily implemented alongside multiple KGE models, while still being computationally inexpensive. Triple classification is used to assess the performance of the proposed initialization method, as well as measuring the extent of the reasoning capabilities of the generated OOKB entity embeddings. Two different scenarios are considered for evaluation: facts containing existing entities and facts combining OOKB and existing entities.

The rest of this paper is organized as follows. Section 2 discusses related works. The proposed initialization method is presented in Section 3. Section 4 reports on the conducted experiments, while Section 5 provides a general discussion on the obtained results. Section 6 draws the conclusions.

2. Related work

A KG consists of three main parts: a set of entities E, a set of relations R, and the interactions between entities and relations, represented by facts (or triples). Triples follow the structure (h, r, t), where h is the *head* entity, t is the *tail* entity, and r is the relation between them. The complete set of known facts composes the KG, which serves as the scaffold for the construction of a KG embedding.

2.1. Knowledge graph embeddings

The purpose of KG embeddings is to subsymbolically represent the information encoded by the facts, associating every entity and relation with a unique representation. The resulting depiction aims to embed the knowledge stored in the facts so that several constraints are met. One of the most important restrictions is that those entities representing similar concepts should have similar embeddings, meaning that the distance between their representations should be fairly small. An appropriate scoring function must be selected to ensure that this property holds.

According to the scoring function, two different types of approaches can be defined: translation-based models and semantic matching models. Translation-based models aim to minimize distance-based scoring functions. In this context, the relation between two entities is perceived as a translation operation, namely:

$$h + r \approx t\,.$$

In such models, both entities and relations are represented as vectors of the same space $\mathbb{R}^k$, where k is the embedding dimension. Relation representations r are interpreted as translation embeddings, so a fact (h, r, t) holds when the distance between the operation $h + r$ and t is minimal. Figure 1 shows a simple representation of this operation. Therefore, the following scoring function is used to ensure higher scores when the aforementioned constraint is met:

$$f_r(h, t) = \|h + r - t\|_p \quad (p \in \{1, 2\})\,.$$

Out of the existing translation-based models, *TransE* [4] is the most commonly used. Although it achieves accurate results despite its simplicity, it also presents several flaws, including its incapability of dealing with 1-1 relations. Several variations of TransE have been developed to deal with this shortcoming, such as *TransD* [10] or *TransR* [13]. TransE is also the cornerstone of several state-of-the-art approaches, such as RotatE [30] and CrossE [39].

The second approach to generate knowledge graph embeddings are semantic matching models, which measure the plausibility of a fact by matching the latent semantics of entities and relations. *RESCAL* [19] is considered to be the first model to exploit latent semantics to represent KGs. This model represents each entity as a vector and each relation with a matrix that models pairwise interactions between each entity on the graph. Therefore, this representation is formalized as a tensor of dimension $M \times M \times N$, where M is the number of

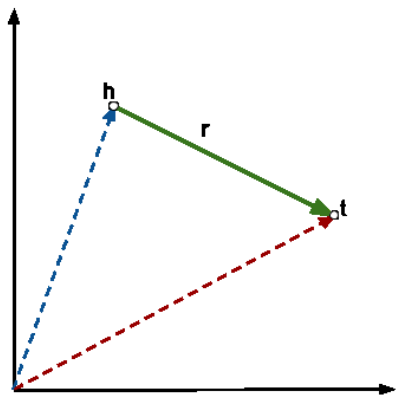


Figure 1: Representation of the TransE operation, adapted from [4].

entities, and N is the number of relations. Every slice of the tensor models each relation. The following scoring function is employed to measure the plausibility of a fact, where h and t are two entity vectors, and M_r is the relation matrix:

$$f_r(h,t) = h^T M_r t\,.$$

Although this model provides an excellent performance, while being fairly intuitive, it lacks scalability, as its complexity increases proportionally with the size of the KG. *DistMult* [37] provides a solution to this flaw by forcing relation matrices to be diagonal, simplifying the base model considerably. *ComplEx*[33] provides a solution to RESCAL's incapability of modelling asymmetric relations, by representing entities and relations in a complex space instead of a real one.ANALOGY [14] unifies the strategies proposed by ComplEx and DistMult, exploiting the analogical properties embedded in the KG, while still having a low complexity.

Finally, neural-network-based approaches combine the entity and relation representations of semantic matching models with the processing capability of neural networks. *Neural tensor networks* [29] are one example of such models. A relation is represented not only by a single matrix but by a unique set of neural network parameters, as shown in Figure 2. Therefore, to determine whether a fact holds or not, the two entities are operated with the set of parameters associated with their joining relation, obtaining a score that represents its plausibility. SimplE [11] provides an enhanced version of canonical polyadic tensor factorization, providing fully-expressive representations while reducing the computational resources required. More recently, the emergence of graph neural networks [1, 42] has initiated a new stream of KG embeddings, leading to models such as those by Nathani et al. [18] and Zhang et al. [40].

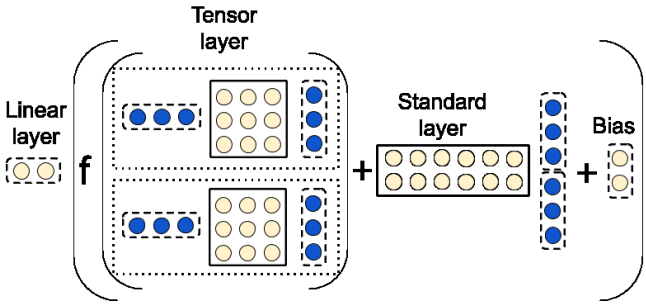


Figure 2: Adaptation of the neural tensor layer presented in [29]. In blue, entity representations. In yellow, the values associated with relation parameters.

2.2. Knowledge graph completion

KG completion is one of the main applications of KG embeddings. Its goal is to extract new facts from the current information, and it can subsequently be divided into two tasks: triple classification and prediction. In triple classification, the goal is to determine whether a fact (h, r, t) holds. Therefore, the fact is evaluated by the scoring function of the model, assigning a confidence value of f to it. If the provided score is higher than a certain threshold, the fact is considered to be true and false, otherwise.

On triple prediction, given an incomplete fact $(?, r, t)$ or $(h, r, ?)$, the goal is to fill in the missing value in the triple. All the potential triples meeting the given constraints are constructed and evaluated to predict the missing value. Generated triples are ranked by its score in decreasing order. Those triples with higher feasibility are expected to feature higher in the ranking. Link prediction can also be considered a variation of triple prediction, where the objective is to determine the relation that best links the entities of an incomplete fact $(h, ?, t)$. In the example presented in Figure 3, having the head entity "Michelle Obama" and the tail entity "United States", the expected predicted relation between them would be "nationality".

KG completion approaches typically follow a local search strategy instead of an explorative one, limiting their scope to entities and relations existing during training. Due to the ever-growing nature of KGs, scalable approaches are needed to cover reasoning over the so-called *out-of-knowledge-base* (*OOKB*) entities. This term refers to entities that are unseen during the training process and, therefore, have no embedding associated.

A potential solution for this issue lies in the inclusion of ontological information, as previously explored by Hohenecker and Lukasiewicz [9]. Ontologies are an inherent part of KGs, serving as a base for their generation and providing explicit restrictions about relations. However, as noted by Paulheim [21], their usage is limited to the generation and introduction phases of the KG, while their usage in subsequent steps is usually dismissed.

Recently, several proposals capable of dealing with reasoning over OOKB

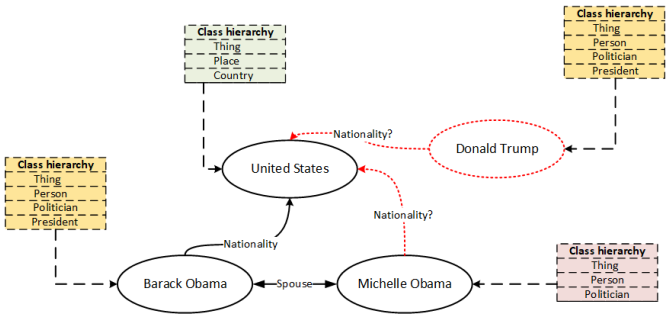


Figure 3: An example of KG completion. Boxes limited by dashed lines represent the class hierarchy of the entity. Dotted lines are used to represent unknown information.

facts have emerged. Tay et al. [32] proposed *puTransE*, an online extension of TransE capable of embedding OOKB entities without retraining the model from scratch. *DKGE*, proposed by Wu et al. [36], relies on graph convolutional networks, alongside with a gating strategy and translation operations inspired by TransE, to model new facts. This approach considers the context of the information to be introduced, updating only those elements that are related to the new fact. Hamaguchi et al. [8] employ graph neural networks to deal with OOKB entity representation, focusing on the triple classification task. Finally, Shah et al. [28] present an alignment model between word and KG embeddings, capable of generating approximate representations for a given entity from its word embedding. Even though these approaches remove the necessity to retrain the model from scratch when introducing an OOKB entity, they still present some flaws that need to be addressed. First, they still require retraining to a certain degree. Secondly, some are based on fairly complex models such as graph neural networks, which require a high computational power. In this work, a simpler alternative to the previously mentioned works is presented, which not only offers a solution to the aforementioned shortcomings but can be easily integrated within any KGE model.

3. Semantic-based initialization to enhance KG embedding

Since most existing KG embedding methods can only perform on a fixed set of known facts, retraining the model from scratch is usually required if new information is introduced to the KG. However, aside from the particular nature of KG embeddings, where every entity has a unique representation, certain generalizations can be stated from the existing facts. This general information is usually encoded in ontologies, and its inclusion can enable KG embeddings to achieve a certain degree of generalization, while still maintaining a high specificity. Furthermore, ontological information is shared across similar entities, while being static over time, which complements the volatile nature of KGs. Ontologies also play a key role in the introduction of new entities in the graph,

and this information is usually explicitly declared for each new and existing entity . Thus, even for those entities that are not included in the KG, their ontological information may still be known.

Figure 3 illustrates the presented idea. As shown, every entity has its class hierarchy, which remains static throughout time. Class hierarchies are similar or even equal in those entities representing closely related concepts. Such is the case of the entities "Barack Obama" (called e_1) and "Donald Trump" (called e_2), where their class hierarchies are equal, meaning that they represent the same type of concept. In the case of the entity "Michelle Obama", its class hierarchy is also shared with the aforementioned entities, stating that the entity also represents a "Person" and a "Politician". Considering the equality of the class hierarchies of entities e_1 and e_2, even though e_2 was not involved during training, an analogical inference can be performed. Encoding ontological information in conjunction with specific entity information can make it possible to leverage information about similar entities, leading the model to infer non-explicit restrictions that can even enable reasoning over OOKB entities. Even though no further information is known about the entity "Donald Trump", it can be determined whether the fact $(DonaldTrump, nationality, UnitedStates)$ is feasible or not based on its high similarity with the known fact $(BarackObama, nationality, United\ States)$. However, even with the inclusion of ontological information, OOKB entity initialization remains an issue, as none of the general KGE approaches is capable of providing a representation for an unseen entity. An initialization method capable of accurately represent a single entity without retraining the complete KG embedding model is needed.

Figure 4 depicts the proposed incremental six-step procedure conducted to study the possible degree of reasoning over OOKB entities by using a proper initialization in combination with ontological information.

3.1. Entity initialization

The goal of the first phase of the proposed methodology is to determine whether employing a semantic-based initialization improves the training procedure, as well as the results already achieved by several existing KG embedding methods. Even though the models presented in Section 2 can lead to highly representative embeddings, where all the known information about an entity is self-contained in its representation, they lack any semantic information. This kind of information is highly useful and enriches the obtained embeddings with an additional level of knowledge that cannot be inferred otherwise. Furthermore, it can ease the learning procedure, accelerating the model's convergence, while achieving similar or better results. While some works have explored this possibility, as in the case of Socher et al. [29], most of the state-of-the-art methods rely on random initialization for training.

For this purpose, different word embedding approaches can be considered. One-hot vectors are an easy and straightforward approach to the subject. In this encoding, each entity is embedded as a vector of dimension N, where N is the total number of entities to represent. For every entity, a single vector is produced composed entirely of zeros except for a single positive value in a given

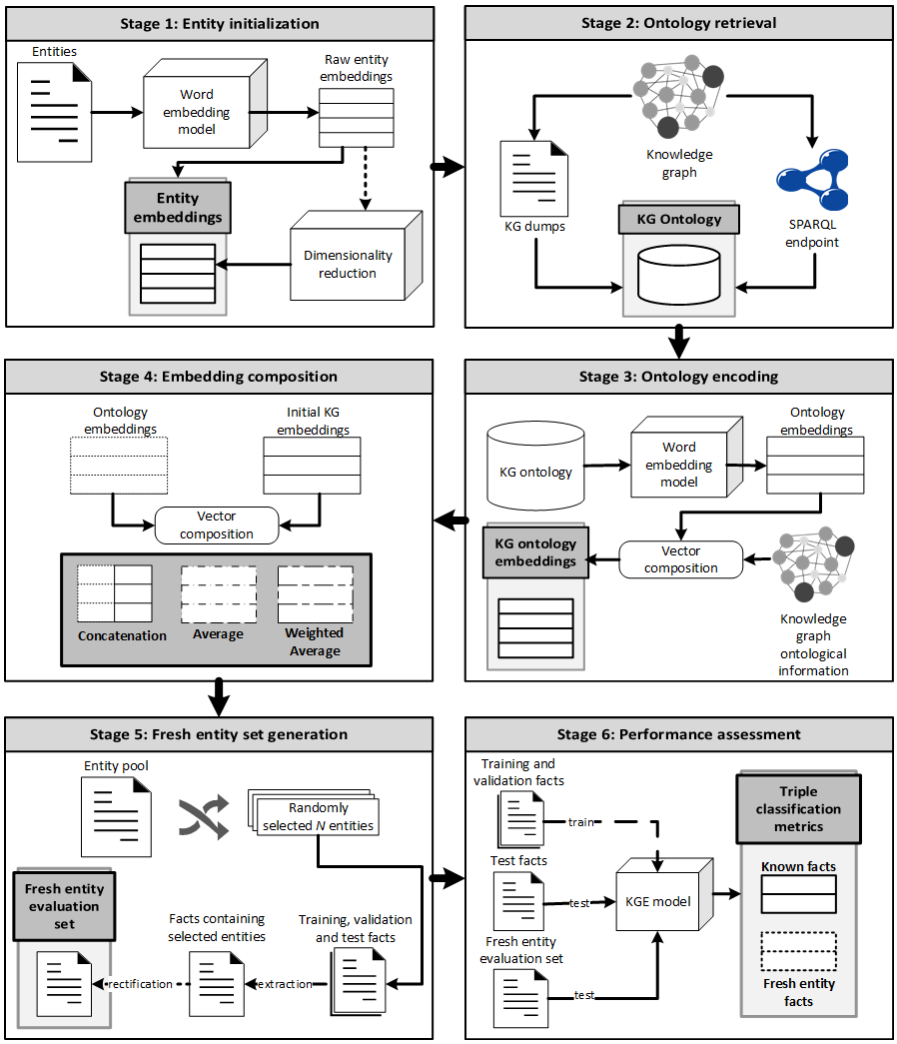


Figure 4: Overview of the six stages of the proposed procedure.

position, which is unique to each entity. While this encoding may be suitable for small datasets, it is inappropriate for large KGs, as the dimension of the embedding grows linearly with the size of the KG. At the same time, zero-valued initializations can hinder the convergence of the model to its optimal solution. Furthermore, this type of representation does not encode specific semantic and syntactic information about the entity.

Word embedding provides a solution to this issue, generating expressive representations for the words in a vocabulary. Word2Vec [15] was one of the pioneer proposals of this kind. A neural network is trained over large corpora of texts, creating a vector representation for each word represented in the training corpus. The resulting embeddings preserve the semantic properties of the vocabulary, enabling analogical inference across words. Although Word2Vec is, to date, one of the most used word embedding models, it lacks scalability, as it only generates representations for the words seen during training time. Therefore, if new words are to be represented, the model has to be retrained from scratch.

FastText [2, 16] introduces an n-gram representation as a solution to the issue mentioned above. In this model, each word is considered a composition of fixed-dimensional parts, called n-grams, which enables the sharing of lexical information between similar words. The representation of a word is the arithmetic mean of the representation of all its n-grams. This composition procedure enables the representation of words that were not previously in the training corpora, while still enabling analogical inference over them.

Recently, word embedding models have been gradually substituted by language models like ElmO [22] or BERT [5]. While word embeddings only consider a fixed window of a given size as context, language models consider a broader scope, such as full sentences. Hence, these approaches lead to more expressive representations.

Similarly to FastText, ElmO [22] also employs an n-gram representation but incorporates an additional dimension by considering the polysemy of words. Therefore, a word with N definitions is associated with N different representations, each of them comprising one of its unique meanings. The context in which the word is placed determines which of its representations is used.

Even though ElmO achieves an outstanding performance in most of the central natural language processing tasks, it bases context modelling on the concatenation of several *long short-term memories* aligned from left to right. This design hinders the model from considering the left and right contexts of the word simultaneously. BERT [5] follows the idea of context-dependent representations presented in ElmO while reducing its complexity. Furthermore, it considers both the left and right context simultaneously while training, achieving even more expressive representations and, subsequently, better results.

Considering the complexity of the information modelled by word embeddings, high-dimensional representations are usually required to be expressive enough, ranging from length 300 in the case of Word2Vec or FastText, up to 2048 on BERT. However, the dimensionality of the representations produced by KG embeddings rarely exceeds 200, as larger embeddings are associated with higher computational times. Dimensionality reduction based on *principal*

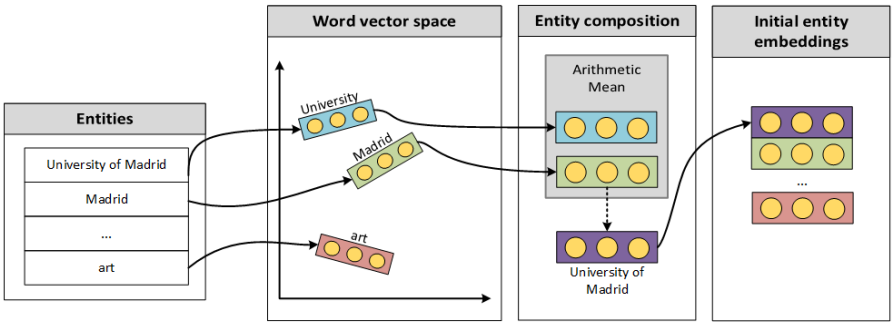


Figure 5: Entity composition by averaging token representations.

component analysis [23] is a potential solution to the mismatch between the recommendable KG embedding and the provided word embedding dimensionalities. However, it is noticeable that reducing the dimension of word embeddings is directly reflected in a reduction of their expressivity.

A significant dissimilarity between word and KG embeddings is that whereas word embeddings follow a token-based approach, KG embeddings are oblivious to the number of words that name an entity. Therefore, a composition method for entity initialization is needed that guarantees that all the tokens that compose an entity are represented in its initial embedding. One possible approach would be to concatenate all the representations of each entity token. This approach, however, is not feasible for KG embeddings, as entity embeddings have to be equally dimensional. Therefore, averaging word representations to generate the initial entity embeddings is a more suitable solution for this issue. Figure 5 shows the procedure of entity composition by averaging its singular token representations.

3.2. Ontology retrieval

Differently from the volatility and specificity of KGs, ontologies provide stable general information about a given domain. Ontologies are involved in the creation procedure of KGs, but their scope is mostly limited to the creation and introduction of new information in the graph. However, this information is particularly useful in the context of KG completion, as ontologies also establish explicit restrictions about relations and provide information that can significantly enhance inference.

Typically, ontologies comprise the following parts: individuals, classes, properties, relations, function terms, restrictions, rules, axioms, and events. Out of all these parts, only individuals (or entities in the context of KGs), relations, and properties are considered by KG embeddings to generate the graph's representations. Therefore, a considerable amount of relevant information is ignored which could lead to more expressive representations.

Class hierarchies are at the core of both ontologies and KGs. During the introduction procedure, entities are classified under the class that better represents

them conceptually. Furthermore, classes are not atomic but form a hierarchical structure, ranging from more specific (or *leaf*) classes to the *root* class, which is shared by every entity represented in the graph. This taxonomical structure not only enables fine-grained distinctions between conceptually similar classes, as illustrated in Figure 3, where both entities "Barack Obama" and "Michelle Obama" are *Politicians*, but one of them is also a *President*. Thus, it provides a tradeoff between the generality provided by the hierarchical taxonomy with the specificity provided by the multiple subclasses that can be represented. Some works, such as HAKE [41] and the one by Tang et al. [31], have recently explored this possibility.

Differently from the generalist view of classes, properties provide distinct and unique information about entities. Although some of this information is translated into facts, the majority of it is usually discarded. This is particularly observed in properties with numerical values, such as years, zip codes, or population density. Nonetheless, this explicit information can also be encoded among the knowledge provided by facts to further enrich the resulting entity representations.

Rules and axioms can also be considered for enhancing KG embeddings. In Wang et al. [35], an approach for rule introduction for KG completion is presented, where rules are modelled independently from the KG embeddings and joined afterwards to perform inference. Guo et al. [7] represent both rules and KG embeddings under the same framework, efficiently incorporating rule information during training, which translates into better predictions. More recently, Wang et al. [34] provide a method for jointly embedding rules and KGs in the same semantic space for translation-based models, severely improving triple prediction results. Zhang et al. [38] present a novel probability-based model that combines soft rules with an EM inference model, achieving an outstanding performance in knowledge graph completion.

Restrictions are also closely related to KG embeddings, as the generated relation representations encode implicit constraints about the expected head and tail entities. However, the restrictions inferred during training have a local scope, as they are based on the particular features about the entities contained in the training facts. Therefore, these constraints are not valid for every fact featuring the relation. To tackle this issue, explicit and general restrictions about relations can be encoded jointly with their representations, which can potentially enhance the model's results on triple classification.

As shown in the second stage of Figure 4, two different paths for KG ontology retrieval can be considered: using KG data dumps or via SPARQL. Currently, most of the publicly available KGs provide their data dumps, which usually contain both the KG facts and the corresponding ontology file. Furthermore, every entity and relation is explicitly related to its corresponding ontological information, which reduces the time needed for the retrieval procedure. However, these dumps tend to come in a single compressed file that can scale up to several terabytes, which can be undesirable in terms of computational resources. KGs usually comprise a SPARQL endpoint, which also enables information retrieval via queries. This is a more efficient approach, as fewer resources are required,

and the target information can be directly obtained with a single query. The time required for information retrieval may increase in this scenario due to network latency. Thus, this approach can be undesirable if the size of the KG becomes large, or if several SPARQL queries are needed.

3.3. Ontology encoding

Once the required ontological information is retrieved, an encoding procedure is needed to generate suitable representations for KG embeddings. Depending on the selected ontological information, either entity or relation embeddings will be affected by its inclusion. Therefore, the chosen encoding procedure must be lined up with the restrictions imposed by the KG embedding.

Class hierarchies only encompass entities and, considering that entity embeddings are mostly vectorial, the same representation type should be used for their encoding. The most simplistic encoding approach uses one-hot vectors of dimension O, where O is the number of classes that comprise the ontology. Despite its simplicity, this representation may be adequate, as it maintains the similarity between the embeddings of the class hierarchies of similar entities. Although suitable, it presents two main flaws: (i) it lacks semantical information about the concept represented by the class, and (ii) its dimension grows linearly with the number of classes of the ontology.

Another approach for class hierarchy encoding is the usage of word embeddings. In this case, a representation is assigned to each ontology class. Considering that an entity can belong to multiple classes, all of its singular class representations must be combined to produce a single embedding. Different vector composition techniques can be employed to merge the singular class representations into a unique one. Averaging all the entity class embeddings is one potential solution, although this approach does not consider the existing hierarchy between the ontology classes. Weighting the class embeddings before averaging their representations could potentially introduce this hierarchy.

Matrices can be used to encode explicit restrictions about relations. Each relation can be represented by a matrix of dimension $C \times C$, where C is the total amount of leaf classes and restrictions that can be encoded using binary vectors. Therefore, if the head class i is restricted to the tail class j through a relation, the cell (i, j) should have a positive value, and zero, otherwise.

3.4. Embedding composition

As stated in the previous stage, the employed encoding must be aligned with its target on the KG. If the ontological information is embedded alongside the KG embedding, then the selected encoding must be restricted to the dimensionality imposed by the KG. The same restriction applies to relations. Depending on the composition procedure, three different approaches can be considered:

1. *Concatenation*: A basic approach is to append both representations to generate a higher-dimensional embedding. This procedure ensures that a section of the embedding remains fixed throughout training and that it is equal for every entity comprising the same ontological information,

enforcing a similarity between their representations. Therefore, if a particular embedding dimension d_T is set for the KG embedding, the following constraint must be met:

$$d_O + d_M = d_T \,,$$

where d_M is the dimension of the target in the KG embedding, and d_O is the dimension of the ontological embedding. In this scenario, d_O and d_M are not required to have the same dimension. While this approach is suitable for the problem, the model's training time increases due to the increment in the size of the embeddings. Furthermore, the optimal sizes for ontological and entity representations must be determined beforehand to ensure a fair tradeoff between specific and general information.

2. *Average*: This approach consists of averaging both representations. In this case, both entity and ontological embeddings must have the same initial dimension, following the constraint:

$$d_T = d_O = d_M .$$

The simplest way to ensure that this restriction is met is to employ the same embedding model for both entity and ontology embeddings. Otherwise, either reduction or zero-padding may be necessary to ensure that both representations have the same dimension before averaging them. This scenario provides equal importance to general and specific information. As all embedding dimensions are identical in this scenario, performing dimensionality reduction is less penalizing than in the previous case.

3. *Weighted average*: Even though averaging solves the dimensionality issue existing in the concatenation, this technique is still oblivious to the balance between general and specific information. Weighted average can be employed to establish a fair tradeoff between the relevance of the general and the specific component of the entity embedding. In the simple averaging approach, both the ontology and entity elements of the embedding shared the same weight. While this may appear to be the fairest approach, it may not be optimal in performance. The dimensionality constraint presented in the previous case has to be met, along with the composition constraint:

$$T = \frac{\lambda * O + (1 - \lambda) * M}{2} \,,$$

where T is the composed representation, O the ontology embedding, M the initial value of the target KG embedding, and λ the employed relevance factor. Although this approach provides the balancing benefits of the concatenation while maintaining a fixed dimension, it poses a new challenge: determining the optimal value of λ. This value can be determined based on external criteria or considered as an additional training parameter.

This ontological information introduction approach can be applied to any KG embedding. However, on a conceptual level, it may not be suitable for any

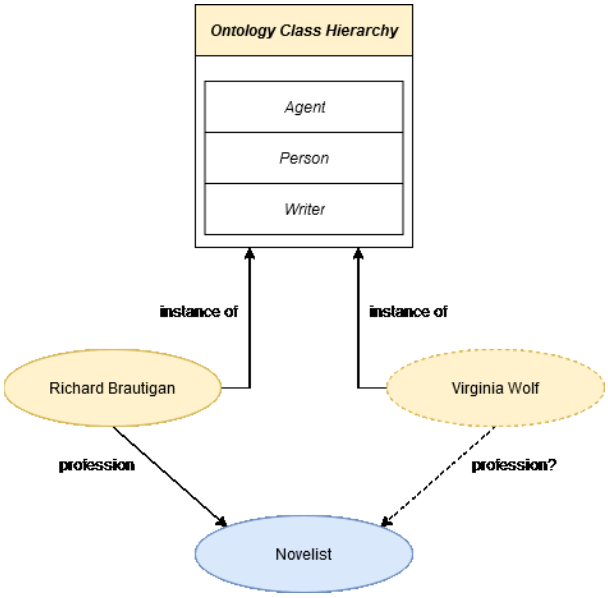


Figure 6: An example of reasoning over facts including OOKB entities. Existing facts are depicted with a continuous line, while a dashed line represents OOKB entities and predictions.

proposal. In translation-based models, the optimization relies on diminishing the distance between the result of a translation operation and the target, which follows a purely arithmetical approach. As neither latent information nor correlations are considered, the introduction of ontological information could be detrimental for the model.

Semantic matching models base their predictions on the exploitation of latent semantics and implicit correlations. Therefore, an enrichment of the semantic information available in the model can potentially be translated into an improvement in its performance. Moreover, the inclusion of ontological knowledge can add an exploratory dimension to KG embeddings, in addition to their exploitative nature.

3.5. OOKB entity set generation

The problem of reasoning over facts, including unseen entities, remains a challenge to most state-of-the-art models. Nonetheless, with the inclusion of semantic and explicit ontological information, unseen entities can be characterized in a reliable (but superficial) manner. Furthermore, by forcing the KG embedding to use word embeddings for initialization and entity class information for training, certain implicit restrictions are inferred about the types of the entities appearing as the head and the tail of a relation. OOKB entities must be initialized following the same procedure as existing entities, so that their ontological information is correctly recognized by the model.

Figure 6 exemplifies the inferential process with OOKB entities. Let the known entity "Richard Brautigan" be h_1, and the OOKB entity "Virginia Wolf" be h_2. The KG embedding encodes the fact *(Richard Brautigan, profession, novelist)* and the class hierarchy of both entities. A parallelism can be established between h_1 and h_2, which can lead the model to correctly determine if the new fact *(Virginia Wolf, profession, novelist)* is feasible, even if no further information about h_2 is provided.

Triple classification of facts composed by a known and an OOKB entity is conducted to assess the extent of this generalization. Triple prediction cannot be performed with facts containing OOKB entities, as the model can only complete the input facts with entities seen during training. As presented in the fifth stage of Figure 4, a set of N entities is randomly extracted from the total entity pool. Subsequently, all the training, validation, and test facts containing one of these entities are extracted, generating a new set comprising entirely facts composed by one known and one unseen entity. The selected KG embedding model is then trained and adjusted with the remaining training and validation datasets, respectively. Finally, it is evaluated over the fully known test set to set a reference mark for the posterior evaluation with the OOKB entity test set.

Considering the constant growth of KGs, developing approaches capable of effortlessly integrating new information is crucial. Most of the current KG embeddings mine information from within the existing data in the graph, but cannot perform any inference over OOKB entities, as they have no associated embedding. Thus, the model would have to be retrained from scratch to include the new entities and the information associated with them, which entails a disproportionate computational effort for a single entity. While predictions over facts containing OOKB entities may not be reliable enough to automatically introduce this new knowledge in the graph, they can serve as guidance to detect new potential introductions. Those facts achieving high scores have a higher probability of being introduced to the KG, whereas those with low scores can be discarded.

3.6. Performance assessment

The final stage of the proposed method is performance assessment, as depicted in Figure 4. Once the model has been evaluated over the known test set and the OOKB entity set for the triple classification task, a comparison between the metrics achieved is established to assess the performance of the model after following the proposed procedure. Based on the accuracy achieved using the fully known set, Acc_T, and the OOKB entity set, Acc_O, there are three possible scenarios:

1. $Acc_T \gg Acc_O$: A considerable disparity between the results in favour of the fully-known set can be an indicator that the ontological information is not being appropriately introduced in the model, inducing an exploitative training approach where the predictions are mostly based on the particularities of the entities. This result can be due to the type of ontological information employed and its level of complexity. If the KG coverage of

Model name	Model type	Embedding Dimensions	Scoring Function	Complexity Order
TransE [4]	Translation-based	$h, t \in \mathbb{R}^d$ $r \in \mathbb{R}^d$	$-\lVert h + r - t \rVert_{\frac{1}{2}}$	$\mathcal{O}(d)$
DistMult [37]	Semantic Matching	$h, t \in \mathbb{R}^d$ $r \in \mathbb{R}^d$	$h^T diag(r) t$	$\mathcal{O}(d)$
ComplEx [33]	Semantic Matching	$h, t \in \mathbb{C}^d$ $r \in \mathbb{C}^d$	$Re(h^T diag(r) \bar{t})$	$\mathcal{O}(d)$
ANALOGY [14]	Semantic Matching	$h, t \in \mathbb{R}^d$ $r \in \mathbb{R}^d$	$h^T M_r t$	$\mathcal{O}(d)$
SimplE [11]	Semantic Matching	$h, t \in \mathbb{R}^d$ $r, r^{-1} \in \mathbb{R}^d$	$\frac{1}{2}[(hrt^T) + (hr^{-1}t)]$	$\mathcal{O}(d)$

Table 1: Summary of the models considered for evaluation. Head and tail entity embeddings are shown as h and t, respectively. Relations whose embedding is vectorial are represented by an r, while relation matrices are defined as M_r. The embedding dimension is denoted by d.

the selected ontological information is minimal (e.g., too many rules have been selected, or the class hierarchy is too complex), then this may be a potential cause for this flaw. Therefore, more straightforward ontological information, or a different encoding approach, could enhance the generalization capability of the model. This would result in an improvement of its coverage, as well as enabling information sharing between existing and new entities.

2. $Acc_T \approx Acc_O$: If the results achieved in both sets are similar, then this indicates that the model is appropriately introducing the ontological information provided, therefore extending its generalization capability to cover unseen entities. A slight difference in the metrics in favour of the fully known set is expected.
3. $Acc_T << Acc_O$: If the results greatly favour the OOKB entity set, there may be a non-optimal parameter selection. Insufficient learning of specific entity knowledge can be one of the causes of this mismatch, indicating that the model has not converged to its optimal solution, but diverged.

4. Experimental results

Out of all the models studied in Section 2, five have been considered for experimental evaluation. Regarding the semantic nature of the proposed method, the experiments focus on its application on the following semantic matching models: SimplE [11], ComplEx [33], DistMult [37], and ANALOGY [14]. Entity initialization is also evaluated in TransE [4]. Table 1 details the properties of each considered model.

Section 2 introduced some existing approaches for reasoning with facts containing OOKB entities. However, they are not directly comparable with our proposal, due to a remarkable difference. These proposals present KGE models, which are specifically designed for OOKB entity reasoning. On the contrary, our proposal is an initialization that can be adapted for different KGE paradigms,

instead of being a model itself. In our approach, OOKB entity reasoning is an induced benefit of the proposal but not the goal itself.

The base implementation of ComplEx and DistMult provided by Trouillon et al. [33] was extended to include SimplE and ANALOGY, as well as the necessary parameters to include entity initialization and ontological information. The implementation provided for the models remained unchanged to ensure the correctness of the obtained results. The parameters employed during the experimentation procedure are those indicated as optimal by their authors. Similarly, the optimization algorithms recommended by the original authors are used: Adam [12] for FB13, and ADAGRAD [6] for WN11.

Two datasets were considered for evaluating the proposed method: FB13 and WN11, as presented by Socher et al. [29]. These datasets contain facts about 13 and 11 relations extracted from Freebase [3] and WordNet [17], respectively, and are specifically designed for triple classification. The relations selected from Freebase are related to the "Person" domain.

Other widely-used datasets (such as FB15k or WN18RR) were not considered for this task, as they are designed specifically for triple prediction, hence being composed entirely of positive facts. If they were to be used for triple classification, negative facts would have to be generated artificially. This would change the original data, subsequently hindering its comparison with other existing approaches.

Three different triple classification metrics are reported to assess model performance:

- *Recall.* This metric determines the proportion of true facts that have actually been denoted as true by the model. Being TP and FN the number of true facts correctly and incorrectly classified respectively, the recall can be obtained as:
$$R = \frac{TP}{TP + FN}. \tag{1}$$

- *Average Precision (AP).* This metric weights the precision and recall increment of the model at different threshold values n. It provides an overall measurement of the model's classification performance while penalizing biased predictions. It is calculated as follows:
$$AP = \sum_n (R_n - R_{n-1}) P_n, \tag{2}$$
where R refers to the recall value, and P represents the precision, computed as:
$$P = \frac{TP}{TP + FP}, \tag{3}$$
where FP denotes the number of unfeasible facts predicted as true.

- *F1 Score.* This metric is the harmonic mean between precision and recall and serves as an indicator of the model's accuracy. It can be calculated

	FB13			WN11		
	TCAP	Recall	F1 Score	TCAP	Recall	F1 Score
ComplEx	0,78	0,65	0,68	0,71	0,58	0,62
CompEx + entity initialization	0,76	0,64	0,67	0,76	0,6	0,64
DistMult	0,71	0,61	0,64	0,68	0,58	0,61
DistMult + entity initialization	0,71	0,61	0,64	0,68	0,58	0,61
TransE	0,72	0,58	0,62	0,82	0,65	0,69
TransE + entity initialization	0,72	0,58	0,62	0,83	0,65	0,69
ANALOGY	0,65	0,52	0,58	0,67	0,58	0,59
ANALOGY + entity initialization	0,68	0,58	0,61	0,68	0,56	0,6
SimplE	0,77	0,65	0,67	0,48	0,49	0,49
SimplE + entity initialization	0,74	0,64	0,67	0,6	0,53	0,55

Table 2: Triple classification average precision, recall, and F1 score with and without entity initialization

using the following equation:

$$F1 = \frac{2(P * R)}{(P + R)}. \tag{4}$$

4.1. Entity initialization

A pre-trained Word2Vec model [27] is employed to obtain the initial embeddings for each entity. As exposed in Section 3.1, entities are compound, whereas the vocabulary represented in the selected Word2Vec model is tokenized. Therefore, the procedure presented in Figure 5 is followed to generate the initial entity embeddings. Firstly, the entity vocabulary V is created, containing all the unique entity tokens. For each token $t \in V$, its representation is extracted from the Word2Vec pre-trained model, and stored in a dictionary of tuples of the form $(token, embedding)$.

The representations provided by the word embedding model have the dimension 300. In contrast, the usual KG embedding dimension ranges from 100 to 200. Therefore, dimensionality reduction based on *principal component analysis* is performed to decrease the size of the token embeddings to 200. Then, for each entity e, its tokens are extracted from the previously constructed dictionary, and averaged afterwards, to generate a single representation per entity. In the example presented in Figure 5, the entity *University of Madrid* is composed of two distinct words: *University* and *Madrid*. Thus, its initial embedding is the arithmetic mean between the representations of both words. Conjunctions and prepositions are not considered. In those entities where none of its tokens is contained in the Word2Vec model's vocabulary, random initialization following a normal distribution is employed to ensure data integrity.

For each of the selected KG embeddings, two different instances are trained: one employing randomly initialized values as usual and another using the proposed entity initialization, marked as *+ entity initialization*. TCAP, F1, and recall values per model instance are reported in Table 2.

```
PREFIX dbr: <http://dbpedia.org/resource/>
SELECT DISTINCT ?class
WHERE {
 ?item rdfs:label "Albert Einstein"@en .
 ?item rdf:type ?class.
 ?class a owl:Class.
}
```

Listing 1: Example of a SPARQL query used to retrieve the class hierarchy of each entity.

4.2. Ontological information introduction

For this stage, the class hierarchy is the selected ontological information, out of all the proposals provided in Section 3.2. This decision is due to its easier integration amongst the entity embeddings, enabling the comparison of the results obtained in this stage with the ones achieved previously.

SPARQL endpoints are employed to retrieve the class hierarchy of each entity. For WordNet entities, the associated homonym ontology is applied, which splits the entities into four different classes, according to the main word types: nouns, adjectives, verbs, and adverbs. In the case of Freebase, two different ontologies are considered. As Freebase is currently part of the Google Knowledge Graph (GKG) [26], it inherits this ontology. The GKG API was employed to retrieve the classes of each Freebase entity, identifying a total of 64 classes. The root class "Thing" was assigned to those entities where no class was specified.

An additional ontology is considered for Freebase to study the dependency between the complexity of the class hierarchy and the performance of the model. The DBpedia ontology [25] is selected, having a taxonomy of a total of 685 different classes. The DBpedia SPARQL endpoint is used to retrieve the class hierarchy of each entity. As this ontology is external to the KG, each entity's class hierarchy cannot be directly extracted from the KG. Thus, the SPARQL query shown in Listing 1 is executed for each entity, exchanging the label content by every entity in the set. The class hierarchy of each entity per ontology is saved up in a dictionary $Dict\_Entity\_Classes$ of tuples of the form $(entity, [classes])$. Out of the total 685 classes of the DBpedia ontology, only 197 appear on the class hierarchies of the entity set. As in the previous case, the root class "Thing" is assigned to those entities where no class information is available.

Once each entity's class hierarchy has been retrieved, an encoding procedure is conducted, as reflected in the third stage of Figure 4. The same Word2Vec pre-trained model employed for entity initialization is used to encode the classes of each ontology. Similarly, for those classes composed of more than one word, average composition is employed to generate a singular representation for each class, subsequently stored in a dictionary $Dict\_Class\_Embeddings$ composed by tuples $(class, embedding)$. Dimensionality reduction to 200 is performed to the class embeddings to maintain equality between the dimension of the entity

	FB13			WN11		
	TCAP	Recall	F1 Score	TCAP	Recall	F1 Score
ComplEx + DBOnt	0,65	0,6	0,62	-	-	-
ComplEx + GKGOnt	0,74	0,62	0,65	-	-	-
ComplEx + WNOnt	-	-	-	0,75	0,58	0,64
CompEx + DBOnt + entity initialization	0,73	0,62	0,65	-	-	-
ComplEx + GKGOnt + entity initialization	0,73	0,61	0,63	-	-	-
ComplEx + WNOnt + entity initialization	-	-	-	0,76	0,6	0,65
DistMult + DBOnt	0,71	0,62	0,64	-	-	-
DistMult + GKGOnt	0,74	0,62	0,65	-	-	-
DistMult + WNOnt	-	-	-	0,65	0,57	0,6
DistMult + DBOnt + entity initialization	0,71	0,62	0,64	-	-	-
DistMult + GKGOnt + entity initialization	0,75	0,61	0,62	-	-	-
DistMult + WNOnt + entity initialization	-	-	-	0,75	0,6	0,64
ANALOGY + DBOnt	0,67	0,6	0,62	-	-	-
ANALOGY + GKGOnt	0,68	0,6	0,63	-	-	-
ANALOGY + WNOnt	-	-	-	0,7	0,58	0,6
ANALOGY + DBOnt + entity initialization	0,67	0,6	0,62	-	-	-
ANALOGY + GKGOnt + entity initialization	0,68	0,6	0,62	-	-	-
ANALOGY + WNOnt + entity initialization	-	-	-	0,64	0,55	0,59
SimplE + DBOnt	0,63	0,6	0,61	-	-	-
SimplE + GKGOnt	0,76	0,63	0,66	-	-	-
SimplE + WNOnt	-	-	-	0,6	0,55	0,57
SimplE + DBOnt + entity initialization	0,66	0,6	0,62	-	-	-
SimplE + GKGOnt + entity initialization	0,74	0,61	0,65	-	-	-
SimplE + WNOnt + entity initialization	-	-	-	0,62	0,54	0,58

Table 3: Triple classification average precision per instance model with and without entity initialization and ontology introduction. For Freebase, two different ontologies are used: DBOnt refers to the DBpedia Ontology, and GKGOnt to the Google Knowledge Graph Ontology.

and ontology embeddings.

For each entity e in the entity set, its class list C is extracted from the dictionary $Dict\_Entity\_Classes$ to generate the initial ontological information matrix. For each $c \in C$, its corresponding class embedding is retrieved from the dictionary $Dict\_Class\_Embeddings$, then averaged to produce a single representation per entity. As a result, a matrix of dimension $N \times 200$ is obtained, where N is the number of entities. Each entity embedding is then averaged with its corresponding ontology embedding on each iteration to encode this information in the KG embedding, generating a single representation of dimension 200 for each entity. Whereas the entity embeddings are model parameters, therefore updated throughout training, the ontology embeddings remain static throughout the training procedure. This constraint ensures that the ontological information remains constant and that the equality between the class hierarchies of similar entities is always met.

This approach is evaluated over the four semantic matching models considered. The metrics achieved by each model with each ontology are presented in Table 3.

4.3. OOKB entity fact evaluation

As shown in Figure 4, the final two stages of the method comprise the evaluation of facts with OOKB entities and an assessment of the model performance based on the model's results in this task. 1500 entities are randomly selected

	FB13	WN11
training facts	300.134	103.789
validation facts	11.366	4.803
test facts	45.793	19.364
OOKB facts	34.319	19.722

Table 4: Number of examples per set and dataset. All training facts are positive, whereas validation, test, and OOKB facts contain an equal proportion of positive and negative examples.

from the total entity pool, both in Freebase and WordNet, to generate the OOKB entity set. To ensure that these entities remain entirely anonymous to the model, all facts containing exactly one OOKB entity are extracted from the training, validation, and test sets, generating a new set of facts composed of one known and one OOKB entity. Facts composed of two OOKB entities are discarded.

As the training set contains only positive examples, the resulting OOKB entity set is severely imbalanced in favour of the positive (or feasible) examples. Therefore, to maintain the proportion of the fully known test set, which has the same amount of positive and negative examples, a corrupted sample is generated for each fact retrieved from the training set. The known entity of each fact is randomly switched by another entity from the known set to create negative (or unfeasible) examples, thus following the local closed-world assumption [24]. Table 4 provides the number of examples of each set per studied dataset.

In this evaluation, random initialization is not considered, as the introduction of facts containing OOKB entities requires entity initialization to exploit the semantics contained in word embeddings to extend this knowledge to unseen entities.

Furthermore, ontological information is also required, as the generalization capability of the model is directly linked to its introduction. As in the previous stage, two different ontologies are considered for Freebase, but only one for WordNet. Each model is trained with the remaining training facts and evaluated over the fully known and the OOKB set for the triple classification task.

5. Discussion

5.1. Entity initialization

According to the results reported in Table 2, entity initialization induces a slight improvement in several of the studied models. The introduction of explicit semantic information about the entities, combined with the information provided by the KG training facts, leads to more expressive representations. This quality increment is reflected in the achieved results, as shown by the increase in the triple classification average precision reached by ComplEx and SimplE in WordNet (WN11). This improvement is particularly remarkable in the case of SimplE, where the triple classification average precision rises from 48% to 62%. In ComplEx and DistMult, the improvements, though existent,

	FB13					
	Known set			OOKB set		
	TCAP	Recall	F1 Score	TCAP	Recall	F1 Score
ComplEx + DBOnt	0,75	0,63	0,66	0,63	0,5	0,55
ComplEx + GKGOnt	0,72	0,61	0,65	0,75	0,58	0,65
DistMult + DBOnt	0,76	0,62	0,65	0,6	0,6	0,59
DistMult + GKGOnt	0,62	0,6	0,6	0,71	0,5	0,53
ANALOGY + DBOnt	0,74	0,63	0,65	0,6	0,48	0,62
ANALOGY + GKGOnt	0,73	0,62	0,66	0,74	0,55	0,53
SimplE + DBOnt	0,69	0,6	0,63	0,61	0,47	0,53
SimplE + GKGOnt	0,77	0,63	0,67	0,71	0,54	0,6

Table 5: TCAP, Recall, and F1 Score for the Freebase dataset.

	WN11					
	Known set			OOKB set		
	TCAP	Recall	F1 Score	TCAP	Recall	F1 Score
ComplEx + WNOnt	0,75	0,61	0,64	0,51	0,5	0,64
DistMult + WNOnt	0,72	0,59	0,63	0,54	0,5	0,65
ANALOGY + WNOnt	0,65	0,58	0,59	0,52	0,51	0,51
SimplE + WNOnt	0,62	0,55	0,57	0,52	0,51	0,52

Table 6: TCAP, Recall, and F1 Score for the WordNet dataset.

Table 7: Top: (a) metrics obtained for the Freebase dataset. DBOnt refers to the DBpedia ontology, and GKGOnt to the Google Knowledge Graph Ontology. Bottom: (b) metrics obtained for the WordNet dataset. Entity initialization is used in all instances

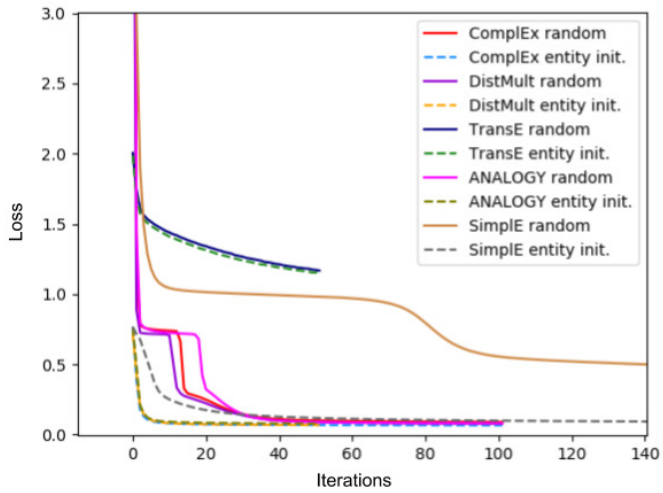


Figure 7: Loss convergence per iteration for each studied model according to the employed initialization. Dashed lines relate to entity initialization, whereas solid lines represent random initialization.

are less pronounced: they are close to 4.5%. In Freebase (FB13), the results on triple classification remain fairly similar with and without initialization. This same statement holds for the recall and F1 scores obtained, where the values are almost identical in all model instances.

In TransE, no improvement is observed in neither one of the studied datasets. This is directly related to the model's approach to generating the embeddings, as it does not rely on latent semantics to achieve optimal representations. Therefore, the additional semantic information provided by the initialization is not exploited by the model and is treated as randomly-valued initialization.

Aside from the improvement in the obtained results, the proposed initialization directly benefits from the training procedure of the models, easing their convergence and reducing the number of iterations needed. Figure 7 shows that models employing entity initialization converge to a stable solution considerably sooner than those using randomly initialized values. However, this affirmation does not hold for TransE, since the convergence rate of both initialization options remains relatively similar throughout the training.

5.2. Ontological information introduction

For ontology introduction, the results in Table 3 evidence an additional improvement over the results achieved with entity initialization, reported in Table 2. In WN11, the results of ComplEx, DistMult, and SimplE employing ontological information with entity initialization noticeably outperform their

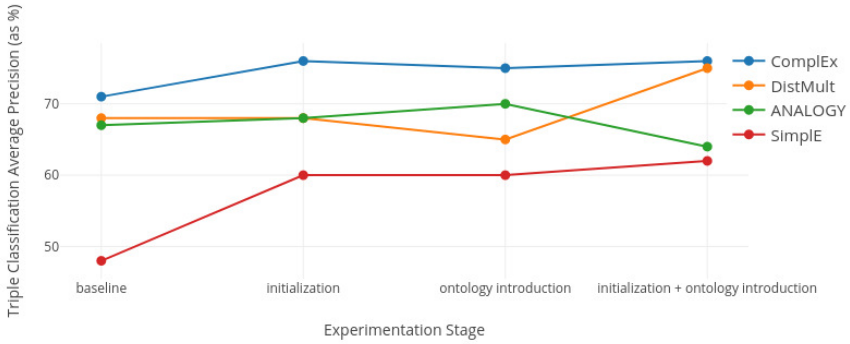


Figure 8: WN11 triple classification average precision (in percentage) per model and experimentation stage.

corresponding baseline models. Figure 8 depicts each model's gradual improvement in the WN11 dataset at successive experimentation stages. This increment is particularly remarkable in DistMult, where the baseline model reached a 68% average precision. In comparison, the introduction of ontological information combined with non-random initialization induces a noticeable 7% improvement, reaching a final 75%. This improvement is even more noticeable in SimplE, where the baseline model achieved an accuracy of under 50%. With the introduction of ontological information and entity initialization, this value steadily scales up to 62%, about 15% higher than the baseline value. This increment, although less pronounced, is also reflected in ComplEx results.

Furthermore, the introduction of ontological information, even without entity initialization, remarkably improves the models' performance. The simplicity of the employed ontology, comprised of only four disjoint classes, plays a crucial role in this improvement.

In FB13, while the improvement is less noticeable than in WN11, there is still a slight improvement in several of the studied models. In this scenario, entity initialization does not further enhance the results introduced by adding ontological information. This may be directly related to the nature of each dataset, as WN11 models semantic relations between words, and subsequently benefits from the inclusion of additional semantic information about entities. Regarding ontology comparison, it can not be asserted whether the complexity of the ontology affects the results obtained for triple classification, as both ontologies lead to similar results. Nonetheless, the complexity of the ontology could be a key factor for triple prediction, as it would significantly reduce the dimension of the candidate pool in tasks such as head or tail prediction.

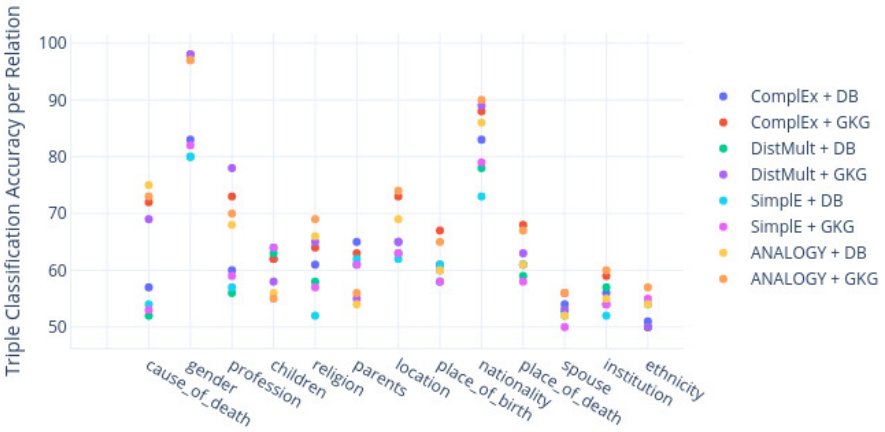
5.3. OOKB entity fact evaluation

With entity initialization and ontological information demonstrating a performance improvement for KG embeddings, the extent of this approach is evaluated over facts including OOKB entities. As evidenced by the results reported in Table 2, entity initialization using word embeddings increases the specific knowledge about the entities, encompassing semantic information that could not be inferred by the model. On the other hand, the introduction of ontological knowledge (Table 3) improves the generalization aspect of the model, enabling the inference of general restrictions about the relations, which directly translates into an improvement in the obtained accuracy. Therefore, considering these two aspects, an instance of each model employing entity initialization and ontological information is trained over the training facts remaining after the fifth stage of the procedure, as shown in Figure 4.

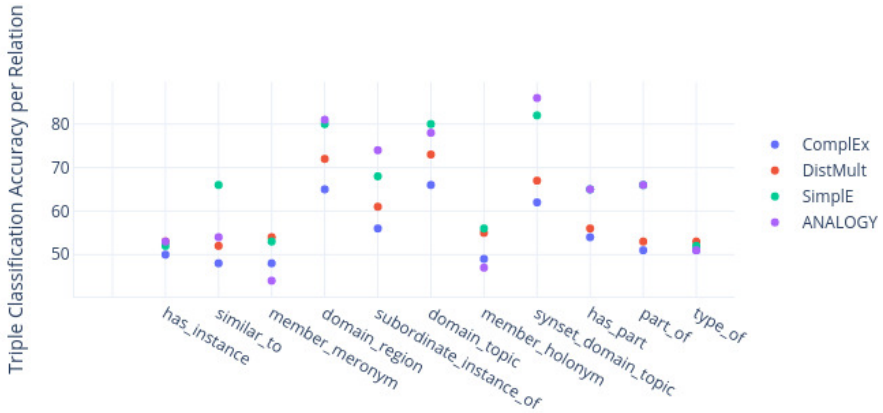
In WN11, even though ontological information considerably enhances the models' performance, no strict class restrictions can be inferred due to the simplicity of the ontology. Therefore, even though the models achieve a high precision over the test set, this performance is not extended to the OOKB entity set, where this value is slightly above 50%, as shown in Table 6. Figure 9b shows an in-depth vision of the results obtained by each model over the OOKB entity set per relation. As showcased in the figure, although the TCAP is between 50% and 60% in most relations, it may rise to around 70% like in "Domain Region" and "Domain Topic".

Table 5 shows that the results achieved in FB13 for the OOKB entity set are similar to those obtained for the fully known test set, except for DistMult, where the results obtained in the OOKB set are slightly higher. This is directly related to the nature of the information modelled in the KG, as the restrictions associated with its relations are closely related to the entities' classes. This statement is further validated by the results in Figure 9a, where the average precision in half of the studied relations is at least 70%. The results show a direct correlation between the homogeneity of the head and tail classes of the relation and the average precision obtained. Therefore, in relations such as "Children", "Parents", or "Spouse" that always relate entities that refer to people, the precision obtained is still close to 60%, as restrictions are easily inferred. Similarly, the relations "Location" and "Nationality" also achieve a remarkable performance in terms of precision, though the results fluctuate across the different models. It is worth noting that, in both relations, models employing the GKG ontology achieve better results than those using the DBpedia ontology. This is a consequence of the fine-graining level provided by the DBpedia ontology for the general class "Place", where more than 170 specific sub-classes are distinguished, which is more than twice the size of the complete GKG ontology.

As stated in Section 2, there are existing approaches that directly tackle the problem of reasoning over OOKB entities. However, the following flaws can be identified on a general overview: (i) they cannot be applied to any KG embedding model, (ii) they rely on complex representations, such as graph neural networks, and (iii) they require, at least partially, a retraining of the model to introduce new entities.



(a) FB13 TCAP per relation



(b) WN11 TCAP per relation

Figure 9: Triple classification average precision (TCAP) in percentage per relation on the OOKB datasets. Figure 9a provides the results on FB13, while Figure 9b illustrates the results obtained in WN11.

The presented approach can be easily implemented alongside every KG embedding model, does not require retraining, and does not add complexity to the model itself. As reported in Table 1, all the studied models have a complexity of $\mathcal{O}(d)$, where d is the embedding dimension. The proposed initialization is performed offline, and its complexity is the same independently from the embedding composition approach. Therefore, regarding d, the proposal's overall complexity is equal to that of the selected KGE model. Besides, the results obtained in FB13 show that the approach is capable of reasoning with facts containing OOKB entities as well as with those composed by two known entities, exhibiting the robustness of the proposal.

6. Conclusion

This work has presented a novel and general KG embedding initialization method. The method proposed is based on the use of word embedding techniques and ontological information, which is directly encoded in the KG embedding initialization. This contribution allows KG embeddings to improve in the KG triple classification task. The experimental results show that the method enables classical KG embeddings (ComplEx, DistMult, ANALOGY, and SimplE) to progress in these tasks considering the standard benchmarks (FB13 and WN11). The proposed approach is capable of improving the accuracy in triple classification by up to 15%.

Beyond the improvement in the obtained results, the method presented in this paper also allows KG embedding models to converge faster to their optimal solution, reducing the number of iterations needed in half.

This contribution also allows KG embeddings to predict over *OOKB entities*, i.e., completing triples where one of the entities has not been included in the training or validation datasets. This advance is essential, because KGs are continually expanding, and obtaining KG embeddings for them is computationally expensive in terms of resources. While some proposals focus on this issue achieve good results, the current approach is straightforward, can be implemented alongside any KG embedding model, and obtains competitive results at a much lower computational cost.

Our main future work is to evaluate this method to conduct *transfer learning* with KG embeddings. We hypothesize that, since several datasets typically share an ontology, a KG embedding trained for one of these datasets could predict and complete triples belonging to the others, leveraging the information encoded in the shared ontology. Future research topics include introducing different knowledge models, such as rules or textual descriptions, within the proposed initialization and assessing their impact on explainability. An in-depth study on how the complexity and features of these models affect the final performance of the KGE may be beneficial to optimize the proposed initialization model.

Acknowledgements

This research work is supported by the Spanish Ministry of Science, Innovation and Universities under the program "Estancias de movilidad en el extranjero José Castillejo para jóvenes doctores" (CAS18/00229), by the Autonomous Region of Madrid through the program CABAHLA-CM (GA No. P2018/TCS-4423), and by the "Universidad Politécnica de Madrid" under the programs "Ayudas al Personal Docente e Investigador para Estancias Breves en el Extranjero", and "Ayudas para Contratos Predoctorales para la Realización del Doctorado". The authors also thank the reviewers and editors for their valuable comments and suggestions, which have improved this paper.

References

[1] Arora, S. (2020). A survey on graph neural networks for knowledge graph completion. *CoRR*, *abs/2007.12374*.

[2] Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, *5*, 135–146.

[3] Bollacker, K., Evans, C., Paritosh, P., Sturge, T., & Taylor, J. (2008). Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data* SIGMOD (pp. 1247–1250).

[4] Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26* (pp. 2787–2795).

[5] Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. *CoRR*, *abs/1810.04805*.

[6] Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, *12*, 2121–2159.

[7] Guo, S., Wang, Q., Wang, L., Wang, B., & Guo, L. (2016). Jointly embedding knowledge graphs and logical rules. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing* (pp. 192–202).

[8] Hamaguchi, T., Oiwa, H., Shimbo, M., & Matsumoto, Y. (2017). Knowledge transfer for out-of-knowledge-base entities: A graph neural network approach. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence* (pp. 1802–1808).

[9] Hohenecker, P., & Lukasiewicz, T. (2020). Ontology reasoning with deep neural networks. *Journal of Artificial Intelligence Research*, *68*, 503–540.

[10] Ji, G., He, S., Xu, L., Liu, K., & Zhao, J. (2015). Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing* (pp. 687–696).

[11] Kazemi, S. M., & Poole, D. (2018). Simple embedding for link prediction in knowledge graphs. In *Advances in Neural Information Processing Systems 31* (pp. 4284–4295).

[12] Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, Conference Track Proceedings*.

[13] Lin, Y., Liu, Z., Sun, M., Liu, Y., & Zhu, X. (2015). Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*.

[14] Liu, H., Wu, Y., & Yang, Y. (2017). Analogical inference for multi-relational embeddings. In D. Precup, & Y. W. Teh (Eds.), *Proceedings of Machine Learning Research* (pp. 2168–2178). volume 70.

[15] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, Workshop Track Proceedings*.

[16] Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., & Joulin, A. (2018). Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation*.

[17] Miller, G. A. (1995). WordNet: A lexical database for English. *Communications of the ACM*, *38*, 39–41.

[18] Nathani, D., Chauhan, J., Sharma, C., & Kaul, M. (2019). Learning attention-based embeddings for relation prediction in knowledge graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (pp. 4710–4723).

[19] Nickel, M., Tresp, V., & Kriegel, H.-P. (2011). A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on Machine Learning* (pp. 809–816).

[20] Noy, N. F., & McGuinness, D. L. (2001). *Ontology development 101: A guide to creating your first ontology*. Technical Report Stanford Knowledge Systems Laboratory KSL-01-05 and Stanford Medical Informatics SMI-2001-0880, Stanford, CA.

[21] Paulheim, H. (2017). Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web*, *8*, 489–508.

[22] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.

[23] Raunak, V., Gupta, V., & Metze, F. (2019). Effective dimensionality reduction for word embeddings. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)* (pp. 235–243).

[24] Reiter, R. (1978). On closed world data bases. In H. Gallaire, & J. Minker (Eds.), *Logic and Data Bases* (pp. 55–76). Boston, MA: Springer US.

[25] Resource (). Dbpedia ontology. `https://wiki.dbpedia.org/services-resources/ontology`. Last Accessed: 2019-05-21.

[26] Resource (). Google knowledge graph search. `https://developers.google.com/knowledge-graph/`. Last Accessed: 2019-05-21.

[27] Resource (). Word2vec model trained over the googlenews dataset. `https://code.google.com/archive/p/word2vec/`. Last Accessed: 2019-05-21.

[28] Shah, H., Villmow, J., Ulges, A., Schwanecke, U., & Shafait, F. (2019). An open-world extension to knowledge graph completion models. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence* (pp. 3044–3051).

[29] Socher, R., Chen, D., Manning, C. D., & Ng, A. (2013). Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems* (pp. 926–934).

[30] Sun, Z., Deng, Z., Nie, J., & Tang, J. (2019). Rotate: Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations*.

[31] Tang, X., Chen, L., Cui, J., & Wei, B. (2019). Knowledge representation learning with entity descriptions, hierarchical types, and textual relations. *Information Processing & Management*, *56*, 809 – 822.

[32] Tay, Y., Luu, A. T., & Hui, S. C. (2017). Non-parametric estimation of multiple embeddings for link prediction on dynamic knowledge graphs. In S. P. Singh, & S. Markovitch (Eds.), *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence* (pp. 1243–1249).

[33] Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., & Bouchard, G. (2016). Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on Machine Learning* (pp. 2071–2080).

[34] Wang, P., Dou, D., Wu, F., de Silva, N., & Jin, L. (2019). Logic rules powered knowledge graph embedding. *CoRR*, *abs/1903.03772*.

[35] Wang, Q., Wang, B., & Guo, L. (2015). Knowledge base completion using embeddings and rules. In *Proceedings of the 24th International Conference on Artificial Intelligence* (pp. 1859–1865).

[36] Wu, T., Khan, A., Gao, H., & Li, C. (2019). Efficiently embedding dynamic knowledge graphs. *CoRR*, *abs/1910.06708*.

[37] Yang, B., Yih, S. W.-t., He, X., Gao, J., & Deng, L. (2015). Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the International Conference on Learning Representations 2015*.

[38] Zhang, R., Mao, Y., & Zhao, W. (2020). Knowledge graphs completion via probabilistic reasoning. *Information Sciences*, *521*, 144–159.

[39] Zhang, W., Paudel, B., Zhang, W., Bernstein, A., & Chen, H. (2019). Interaction embeddings for prediction and explanation in knowledge graphs. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining* (p. 96–104).

[40] Zhang, Y., Chen, X., Yang, Y., Ramamurthy, A., Li, B., Qi, Y., & Song, L. (2020). Efficient probabilistic logic reasoning with graph neural networks. *CoRR*, *abs/2001.11850*.

[41] Zhang, Z., Cai, J., Zhang, Y., & Wang, J. (2020). Learning hierarchy-aware knowledge graph embeddings for link prediction. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence* (pp. 3065–3072).

[42] Zhou, J., Cui, G., Zhang, Z., Yang, C., Liu, Z., & Sun, M. (2019). Graph neural networks: A review of methods and applications. *CoRR*, *abs/1812.08434*.