Towards Interpretable Embeddings: Aligning Representations with Semantic Aspects

Nitisha Jain [a,*], Antoine Domingues [b], Adwait Baokar [a], Albert Meroño Peñuela [a] and Elena Simperl [a]
[a] *Department of Informatics, King's College London, London, UK*
E-mails: {nitisha.jain, adwait.baokar, albert.merono, elena.simperl}@kcl.ac.uk
[b] *ENSTA Paris, Institut Polytechnique de Paris, Palaiseau, France*
E-mail: antoine.domingues@ensta-paris.fr

Abstract. Knowledge Graph Embedding Models (KGEMs) project entities and relations from Knowledge Graphs (KGs) into dense vector spaces, enabling tasks such as link prediction and recommendation systems. However, these embeddings typically suffer from a lack of interpretability and struggle to represent entity similarities in a way that is meaningful to humans. To address these challenges, we introduce *InterpretE*, a neuro-symbolic approach that generates interpretable vector spaces aligned with human-understandable entity aspects. By explicitly linking entity representations to their desired semantic aspects, *InterpretE* not only improves interpretability but also enhances the clustering of similar entities based on these aspects. Our experiments demonstrate that *InterpretE* effectively produces embeddings that are interpretable and improve the evaluation of semantic similarities, making it a valuable tool in explainable AI research by supporting transparent decision-making. By offering insights into how embeddings represent entities, *InterpretE* enables KGEMs to be used for semantic tasks in a more trustworthy and reliable manner.

Keywords: knowledge graph embeddings, semantic similarity, interpretable vectors, explainable AI.

1. Introduction

Knowledge Graphs (KGs) are structured representations of real-world entities and their relationships, organized in the form of nodes and edges, where nodes represent entities while edges illustrate the relationships between them. KGs have gained significant attention for their applications in tasks like question-answering, information retrieval, and recommender systems [3, 17, 27, 70]. Despite the availability of large amounts of source data and the inclusion of millions of facts, knowledge graphs (KGs) remain incomplete, with missing entities or facts about entities. Knowledge Graph Embedding Models (KGEMs) have been proposed to address this limitation. Since the early 2010s, significant advancements have been made in developing KGEMs, which aim to project entities and relations in KGs into a low-dimensional latent vector space. This representation enables machine readability and manipulation of KG data while preserving the relationships between entities. In doing so, KGEMs offer a sub-symbolic way of representing both entities and their connections within the original graph [5]. Several types of KGEMs exist, such as translation-based models (e.g., TransE [4], TransH [65]) and semantic matching models (e.g., RESCAL [45], ComplEx [63]). These models have proven useful in various tasks, including link prediction [50], entity alignment [59], recommendation systems [49] and so on (see [20, 64] for an overview).

Although KGEMs were primarily designed and trained for the task of link prediction or triple completion in knowledge graphs, there is a widespread belief that these models can also effectively capture similarities between

---

[*] Corresponding author.

2949-8732/$35.00 © 0 – IOS Press. All rights reserved.

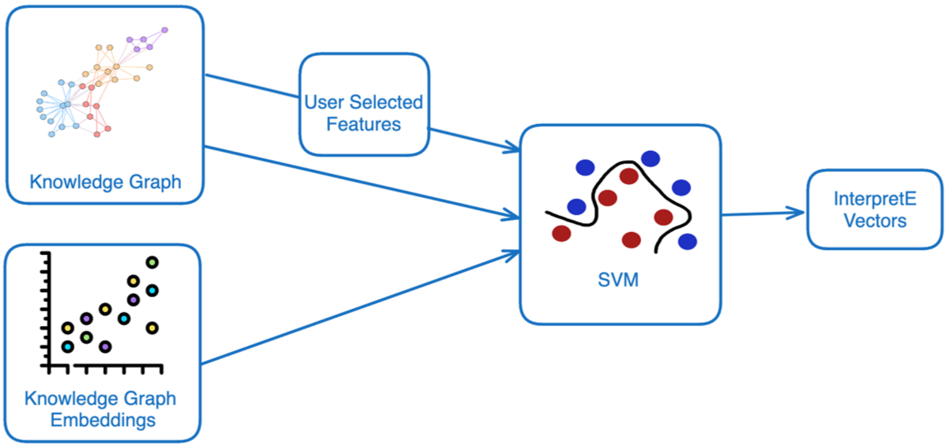


Fig. 1. Overview of the proposed *InterpretE* method

entities, suggesting that similar entities will naturally cluster together in the vector space. As a result, KGEMs have been widely adopted for semantic tasks, including entity or relation similarity and conceptual clustering [18, 38, 60]. The assumption that KGEMs possess strong semantic capabilities was first called into question by Jain et al. [32]. In their study, the authors conducted simple yet systematic experiments, revealing that entities belonging to the same type or ontological class do not consistently cluster together in the vector space, except for the most basic entity types such as *person* and *organization*. Subsequently, other recent studies have delved into this further, arriving at similar conclusions [1, 29]. These findings cast doubt on the generalizability and utility of KGEMs for tasks that rely on capturing semantic relationships effectively.

A fundamental challenge for KGEMs in capturing semantic properties arises from the complexity of the underlying data. Entities in a knowledge graph possess diverse *'aspects'* in terms of their attributes as well their relationships with other entities, all of which significantly impact their vector representations. This complexity makes it exceedingly difficult to identify the *specific factors* that shape the distribution of vectors within the embedding space. Given that entities have different *types* and *numbers* of connections in the KG, and the learned vectors span hundreds of dimensions, there is *no clear correspondence between entity aspects and the dimensions of the resulting vectors*. This absence of a direct mapping leads to a lack of semantic interpretability, making it difficult to understand why certain vectors in the embedding space are similar or to determine which entity aspects influence their representations.

Although a formal definition of interpretability has been elusive in machine learning [44], in this work, we align with a model-based interpretability presented by Murdoch et al. [44] as *'models that readily provide insight into the relationships they have learned*' drawing on a more traditional definition that puts emphasis on *human-understandability* of the functionality of a model [48].

While the ability to represent complex data in low-dimensional spaces allows for large-scale vector manipulations, and is certainly a desirable trait in KGEMs for enabling generalization and their effective application in tasks such as link prediction, this same factor contributes to the poor semantic interpretability of these models. Nevertheless, KGEMs are still widely used in different semantic tasks, making the ability to capture and interpret the semantic features of underlying entities highly desirable. This work aims to bridge this gap by mapping the semantics of the entities with the dimensions in the vector representations of these entities, enhancing the interpretability of these embeddings and improving their utility for semantic tasks.

In this paper, we propose a novel neuro-symbolic approach *InterpretE* that explicitly connects the embedding vectors to the desired task-driven or user selected aspects of the KG entities. Taking inspiration from previous works on *conceptual spaces* [22], we accomplish this by deriving new vector spaces (from the vector of a given KGEM) having *interpretable dimensions* that can be understood in terms of the human-understandable aspects of the entities. This understandability can help in enabling informed decisions in downstream semantic tasks (e.g. recommendation systems and entity clustering), debugging and comparing the models as well as understanding hidden biases [56].

An overview of the proposed approach is shown in Figure 1. While several previous works have proposed KG embedding models that attempt to capture the semantics of entities in terms of ontological information [25, 57], these approaches are limited to encapsulating only the ontological classes or *types* of entities (e.g., whether an entity is a *person* or an *organization*). They are not designed to account for other relevant or application-specific aspects of the entities, for instance, the location where a person was born or the awards received by a scientist. In contrast, our approach allows for the incorporation of a broader range of existing and interesting aspects from KG data, especially for entities. Through various experiments, we demonstrate that the vector spaces generated by *InterpretE* can effectively encapsulate any desired semantic aspects from the KG. Moreover, our method is highly flexible, accommodating a diverse array of entity aspects in terms of both quantity and type.

It is to be noted here that *InterpretE* serves as a way to derive alternative embeddings from existing methods, to specifically improve their interpretability for the applications where such a feature is a desirable and necessary. As such, *InterpretE* embeddings do not intend to compete with or outperform existing KGEMs on tasks such as link prediction, rather they serve as complementary embeddings for semantic tasks. The proposed *InterpretE* method serves as an effective and highly customizable way to obtain these alternative embeddings that can be tailored to fit any downstream semantic tasks. In view of this, the evaluation of the approach is presented in terms of the quality of the resulting clusters in the derived vector space, as well in terms of the semantic similarity of the corresponding entities. We also make the code publicly available[1] to promote further research in this important direction.

Our work is situated within the broader context of explainable AI (XAI) research, where, with the popularity of large language models (LLMs) and their increasing integration across various applications, the importance of transparency and interpretability in these models has garnered significant attention. As large models become more widespread in fields such as healthcare, finance, and autonomous systems, understanding how these models make decisions has become crucial. The importance of XAI stems from concerns related to trust, fairness, and accountability, especially given that deep learning models and KGEMs are often regarded as 'black boxes'. To the best of our knowledge, the *InterpretE* framework introduced in this work represents the first effort to address this issue for KGEMs in terms of restoring semantic interpretability to entity vectors by explicitly mapping these vectors to underlying, human-understandable aspects of the entities.

The salient contributions of our work can be summarized as follows.

- Presentation of a novel neuro-symbolic approach called *InterpretE* that can derive interpretable embeddings (from any KG embedding model) for the KG entities.
- Description of the data-driven process of identifying and selecting desired user-selected or task-oriented entity aspects from KG datasets.
- Demonstration of the proposed method in that the embeddings generated by *InterpretE* encapsulate the desired semantic aspects of the underlying entities and that *InterpretE* is highly flexible in terms of the number and types of aspects that it can work with, making it scalable for different datasets and requirements of downstream applications.
- The evaluation of the approach and the resulting embeddings in terms of the properties in the vector space as well as with the measurement of semantic similarity of the entities illustrates that *InterpretE* indeed leads to improved interpretability for KG embeddings.

The rest of the paper is organized as follows: Section 2 introduces key concepts and background that is essential for understanding the proposed method in detail. Section 3 provides a comprehensive review and comparison with related work, highlighting the ongoing challenges addressed by our approach. In Section 4, we describe the selection process for entity *aspects* or *features* from the KG datasets, followed by a formal description of the *InterpretE* method in Section 5. Section 6 presents the method's evaluation through various experiments, demonstrating its effectiveness and assessing the interpretability of the derived vectors, supported by illustrative plots and a discussion of results. Finally, Section 7 concludes the paper and suggests directions for future work.

---

[1] https://github.com/toniodo/InterpretE

2. Preliminaries

2.1. Knowledge Graphs

A knowledge graph (KG) is a directed graph that represents knowledge in a structured format. It consists of nodes that correspond to real-world entities, such as people or cities, and edges that represent the relationships between these entities. The edges are labeled to indicate the nature of these relationships. More formally, a knowledge graph can be represented as $G = (E, R, T)$, where E is the set of all entities, R is the set of relations and T is the set of triples (h, r, t) such that $h \in E$, $t \in E$, and $r \in R$. Each triplet (h, r, t) indicates a relationship r between the head h and the tail t. KGs play a crucial role in modeling networks of interconnected objects, such as citations, relationships among individuals, and more. Their structured representation facilitates semantic understanding, enabling both machines and humans to interpret complex relationships and contexts within the data. This capability has led to their increasing adoption in diverse fields, such as Computer Vision, where they have been shown to enhance performance through techniques like Graph Convolutional Networks [39]. KGs are particularly useful in various applications in Natural Language Processing. They significantly enhance question-answering systems by allowing the pruning of irrelevant information, which reduces the search space and accelerates the retrieval of accurate answers. For example, the QA-GNN framework [70] showcases how KGs can improve the efficiency and effectiveness of question-answering tasks.

2.2. Knowledge Graph Embeddings

Knowledge graph embedding models aim to represent entities and relations from knowledge graphs as continuous vectors or matrices, known as embeddings (see [64] for an overview). The main purpose of learning these embeddings is to simplify downstream tasks, while preserving the underlying structure of the knowledge graph. A scoring function is used to evaluate how likely a predicted entity is to accurately complete a triple, ensuring that the embeddings maintain the integrity of the original graph's relationships.

Notable types of KGE models are as follows:

Translation Distance Models. These models operate under the assumption that adding the vectors of the head and relation will result in a vector close to that of the tail. One of the earliest examples of this type of KGEMs is TransE [4]. Formally, if h, r and t denote the vectors of the head, relation, and tail respectively, then it holds that: $h + r \approx t$. To ensure the accuracy of the triple, the following scoring function must be minimized:

$$f(h, r, t) = ||\vec{X}_h + \vec{X}_r - \vec{X}_t||_{L_{1,2}} \tag{1}$$

where $\vec{X}_h$, $\vec{X}_r$ and $\vec{X}_t$ are the vectors of the head, relation, and tail, respectively, all residing in the same shared embedding space.

However, TransE struggles to capture complex relationships such as one-to-many, many-to-one, and many-to-many. TransH [65] addresses this limitation by introducing a relation-specific hyperplane for each relationship, allowing entities connected through that relationship to be distinguished based on their unique semantics within that context. TransR [42] builds on a similar concept but defines relation-specific spaces instead of hyperplanes. TransR is further refined by TransD [36], which uses two embedding vectors for each entity and relation and introduces a mapping matrix that generates two distinct mapping matrices for the head and tail entities.

Semantic Matching Models. These models employ a scoring mechanism based on vector similarity, where entities are represented as vectors and relations as matrices. The core assumption is that the transformation of the head embedding will closely approximate the tail embedding, which is formalized as: $\vec{X}_h \vec{Y}_r \approx \vec{X}_t$, where $\vec{X}_h$ and $\vec{X}_t$ are the vectors of the head and tail, respectively, and $\vec{Y}_r$ is the matrix representing the relation used for mapping. RESCAL [45] utilizes a bilinear scoring function, where each relation is represented as a matrix, and the mapping between the head and tail vectors is computed using this matrix. DistMult [68] simplifies RESCAL by constraining the relation matrix to be diagonal, which reduces the number of trainable parameters. ComplEx [63] extends this approach by introducing complex-valued embeddings, enabling the model to capture asymmetrical relations effectively.

Among other types of models, ConvE [16] was the first to predict missing links in knowledge graphs using Convolutional Neural Networks (CNNs). Unlike fully connected dense layers, CNNs can train with fewer parameters, allowing them to capture complex non-linear relationships. ConvE establishes local interactions across multiple dimensions between different entities, enabling it to model intricate patterns more effectively.

2.3. Conceptual Spaces and Interpretable Dimensions

According to Gärdenfors [22], a *conceptual space* is a multidimensional framework where each dimension represents a different quality or property of a concept. These dimensions serve to describe various aspects of a concept in a structured and meaningful way. For example, when considering animals, dimensions such as height, weight, and color could represent specific *quality dimensions* that collectively define the concept of ‘animal’. These dimensions are fundamental in understanding how concepts are represented and compared within the space. Each dimension in a conceptual space is assumed to have its own inherent structure. For instance, some dimensions, like time or weight, are one-dimensional, represented by real, non-negative values. For more complex attributes, such as color, Gärdenfors explains that the mental model can be represented by three dimensions: *hue* (circular), *saturation*, and *brightness* (linear), creating a cognitive conceptual space where different points correspond to specific colors.

In this context, *interpretable dimensions* [15] refer to the axes or directions in the conceptual space that correspond to human-understandable properties of entities. For example, in a conceptual space representing animals, the interpretable dimensions could be height, weight, and speed. Each of these dimensions has a clear and intuitive meaning, making it easier to relate the points in the space to real-world attributes. Interpretable dimensions are critical because they allow us to map abstract vectors or mathematical representations back to meaningful, semantic concepts. To understand the semantics of conceptual spaces, consider that a language L can be interpreted as a projection onto a conceptual space. In this projection, distinct elements of the language are represented as vectors, and predicates within the language correspond to regions or areas in the conceptual space. These regions can be *primary*, representing fundamental concepts, or *secondary*, derived from other regions. In a conceptual space, every point represents a possible individual, with each point consistently displaying well-defined properties based on its position along the interpretable dimensions. This structure allows for clear comparisons and distinctions between concepts, helping to identify similarities and differences based on their positions within the space (see [15] for further details).

3. Related work

3.1. Explainability in Large Models

Recently, the majority of embedding spaces have emerged from the training of large language models (LLMs). However, Simhi et al. [56] highlight a significant limitation of such representations: they often exceed human comprehension. To address this issue, they propose a new method for generating a conceptual space with dynamic granularity based on demand. Their work also introduces a novel assessment technique that demonstrates that the conceptualized vectors indeed reflect the semantics of the original latent representations, validated through either human raters or LLM-based raters. In relation with large models, Cunningham et al. [28] discuss the concept of polysemanticity, which poses a challenge to the interpretation of neural networks. They attempt to reconstruct the internal activations of the language model to tackle this issue arising from neural networks having fewer neurons compared to the features they represent. This line of research is important within the framework of explainable AI [2], our work focuses on Knowledge Graph Embedding Models (KGEMs). While [28] essentially proposes a way to reverse-engineer the monosemantic features from a given network, our intention with *InterpretE* is instead to derive new embedding vectors for the KG entities while aligning them to a set of customizable, pre-defined and desirable aspects of these entities that may be user-defined or task-driven. By striving to make representations more understandable and interpretable, we aim to address the challenges faced in downstream applications where semantics are critical, such as entity similarity and recommendation systems.

3.2. Semantics in KG Embeddings

KG embedding models provide sub-symbolic representations of entities and relations in a KG, and enable the vector manipulations of the data for tasks such as KG completion and triple classification. In recent literature, several critical works have questioned the widely-held assumption that KGEMs produce semantically meaningful representations of underlying entities [29, 32]. In a popular previous work, Jain et al. [32] investigated the degree to which similar entities correspond to similar vectors and concluded that this does not hold true universally. They demonstrated that entity embeddings derived from KGEMs often struggle to effectively discern entity types within a Knowledge Graph (KG), with simpler statistical methods offering comparable performance. Additionally, Ilievski et al. [30] observed consistent under-performance of KGEMs compared to simpler heuristics in tasks reliant on similarity, particularly within word embeddings. The authors argue that many properties that heavily relied upon by KGEMs are not conducive to determining similarity, thereby introducing noise that ultimately undermines performance. In [29], Hubert et al. challenge the widely held belief that entity similarity within a graph is adequately represented in the embedding space. Their comprehensive tests assess the capacity of KGEMs to effectively group related entities and investigate the underlying characteristics of this phenomenon. However, these previous studies primarily focus on questioning the validity of the aforementioned assumption without offering concrete solutions to address the identified shortcomings, which is the focus of our work.

3.3. KG Embeddings and Ontologies

There has been considerable work on embedding ontologies in the literature [11, 21, 23, 25, 57, 58]. Recent techniques have aimed to develop robust and efficient methods for embedding OWL (Web Ontology Language) and OWL2 ontologies that effectively express their semantics. Holter et al. [25] computed embeddings for OWL2 ontologies by projecting ontology axioms into a graph and creating a corpus of phrases through random walks over this graph. A neural language model generates concept embeddings from this corpus. This work addresses limitations in earlier approaches [57, 58] that treated each axiom as a sentence, leading to issues such as insufficient corpus size for small to medium ontologies, noise introduced by OWL constructs, and Word2Vec's inability to differentiate between logically similar sentences. To overcome these challenges, the authors developed a system that (*i*) creates a graph from the ontology, (*ii*) navigates the ontology graph using various techniques, (*iii*) constructs a corpus of phrases based on these walks, and (*iv*) derives concept embeddings from this corpus.

Following this, Chen et al. [11] introduced OWL2Vec^*, an ontology embedding technique based on random walks and word embedding that captures the semantics of an OWL ontology by considering its semantic information, logical constructors, and graph structure. They expanded OWL2Vec to create OWL2Vec^*, a more robust embedding system. OWL2Vec^* navigates the graph forms of an OWL (or OWL2) ontology to generate a corpus of three documents that encapsulate various aspects of the ontology's semantics, including (i) graph topology and logical constructors, (*ii*) syntactic information, and (*iii*) a combination of (*i*) and (*ii*). Ultimately, OWL2Vec^* employs a word embedding model to produce word and entity embeddings from the generated corpus. While these works primarily focus on embedding the semantics represented in ontologies, their goals differ significantly from ours. They do not aim to establish clear connections between the embedding space and the underlying concepts in the ontology. Another line of work concerns with creating embeddings specifically for Ontologies with the goal to enable ontology specific tasks such as ontology learning, reasoning and ontology-mediated question answering [31, 67, 69]. Ontology embedding methods also have been used for vision tasks such as few shot learning and image classification [34, 35].

There are yet other works that are concerned with the integration of ontological knowledge directly into embedding models (e.g., [11, 14, 19, 24, 40, 66, 73]), typically through modifications to the loss function during training. Indeed, while these works have the same motivation of improving the semantics in KG embedding models by leveraging the information in the ontology concepts and roles, contrary to our work, these works do not focus on the interpretability of the embedding spaces that they generate. While adding ontological information during the training of embeddings has been shown to enhance the semantic capabilities of the embeddings in some cases [33], this does not automatically entail interpretability in terms of human-understandable aspects of the entities for the generated embedding space. Moreover, the *InterpretE* approach is not limited to the ontological classes of KG entities.

It can derive interpretable dimensions corresponding to various relevant aspects, including entity attributes (e.g., *gender* for *person* entities, *genre* for *movie* entities) and relationships with other entities (e.g., *bornIn* [location] for *person* entities, locatedIn [location] for *organization* entities), or any combination thereof (which may be user-defined or task-driven). In fact, *InterpretE* can be applied to any of the aforementioned KG embedding techniques, generating interpretable embedding spaces with dimensions reflecting desired semantic aspects.

3.4. Interpretable Dimensions

Various approaches have focused on constructing interpretable spaces using multiple data sources, primarily text but also images [6, 7, 15, 56, 72]. As discussed in Section 2.3, conceptual spaces [22] represent concepts through cognitively meaningful features known as quality dimensions. These dimensions are typically derived from human judgments and serve as an intermediary representation layer between neural and symbolic representations. Bouraoui et al. [15] discuss techniques that facilitate a looser integration between embeddings and symbolic knowledge, deriving similarity and other forms of conceptual relatedness from vector space embeddings to support adaptable reasoning using ontologies. In another work, Bouraoui et al. [7] demonstrate that incorporating conceptual neighbors leads to more accurate region-based representations through a straightforward technique for identifying them. Derrac et al. [6] illustrate how a large corpus of text documents can be leveraged to learn essential semantic relations. While these approaches show promise for advancing explainable AI, they have not been extended to more complex datasets like knowledge graphs and their representations using KGEMs. In contrast, our proposed approach represents a first step toward identifying interpretable dimensions for such models, focusing on the underlying aspects of knowledge graph entities and thereby deriving vector spaces that are more human-understandable.

4. Data Analysis and Selecting Entity Aspects

In Section 5, the *InterpretE* method will be explained as a generalized and scalable process for obtaining entity *aspects* or entity *features*[2] from a given KG dataset, as well as deriving interpretable entity vectors from it. In this section, we focus on dataset acquisition, specifically providing a detailed explanation of the data-driven analysis conducted for two KG benchmark datasets. This analysis aims to illustrate the nuances of entity feature extraction for real-world entities. To derive and categorize aspects for different entities in the KG, their type (or ontological class) information was essential. As such, we leveraged KG datasets with associated ontologies, focusing on subsets of Yago (Yago3-10 [43]) and Freebase (FB15k-237) [61]. Additionally, we reused *Wordnet*-based entity type mappings from Jain et al. [32] to obtain the ontological classes for the entities. As a first step, the entities in the KGs were categorized by their ontological classes using *WordNet* types such as persons, organizations, and locations. Next, for each entity type, the most representative relations were selected and their values were categorized based on their distribution in KG triples.

4.1. YAGO

An overview of the dataset analysis in terms of the most representative entity types for the YAGO3-10 dataset is shown in Figure 2. The YAGO3-10 dataset is dominated by entities of the class *person*. In Figure 2, it can be seen that while *person* is the most frequent class, various subclasses of *person* (at different levels of hierarchy in the ontology structure) are also frequent. For instance, *player* is a subclass of *person*, while *football_player* is a subclass of *player*. This illustrates that the *person* type is extensively represented throughout the dataset, ensuring sufficient data availability for this type in subsequent experiments, as the number of triples associated with it is substantial.

When analyzing a given entity class, emphasis was placed on identifying the most represented relations. High-frequency relations are expected to be effectively captured by the embedding model, encapsulating relevant relational information within the final entity embeddings. The most significant relations for the *person* entities are

---

[2] The terms *aspects* and *features* of the entities are used interchangeably throughout the paper.

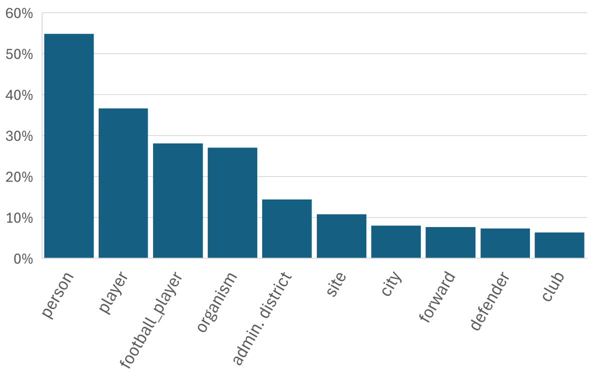


Fig. 2. Top 10 most represented entity classes in YAGO3-10

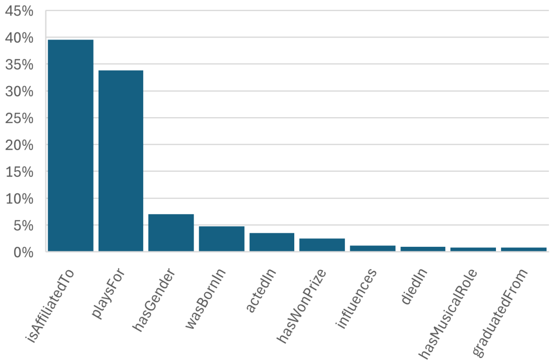


Fig. 3. Top 10 most represented relations for *person* entities in YAGO3-10

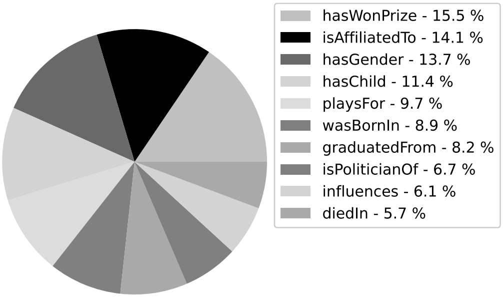


Fig. 4. Most represented relations for class *politician* in YAGO3-10

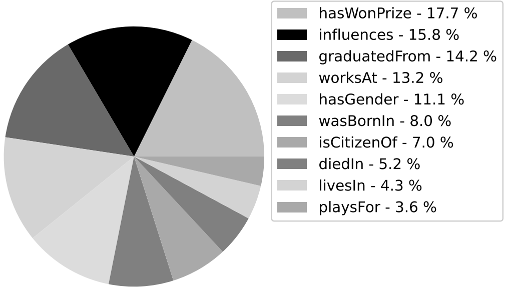


Fig. 5. Most represented relations for class *scientist* in YAGO3-10

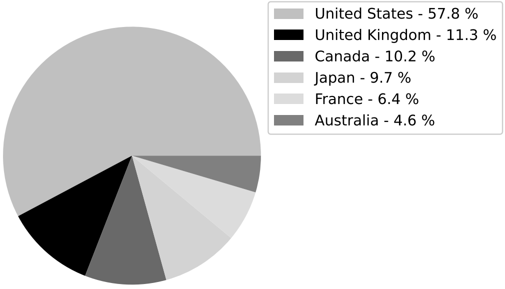


Fig. 6. Most represented values for class *organization* with relation *isLocatedIn* in YAGO3-10

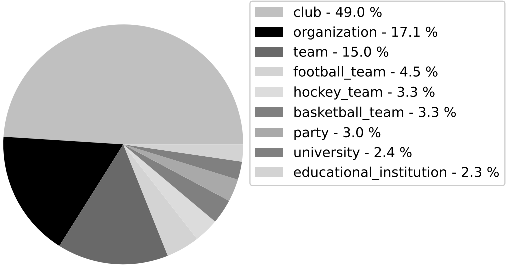


Fig. 7. Most represented values for class *person* in with the relation *isAffiliatedTo* in YAGO3-10

shown in Figure 3. In this context, the relations *isAffiliatedTo* and *playsFor* emerge as the most represented for *person* class. It is interesting to note that an analysis of these relations in the YAGO3-10 dataset revealed that 87.65% of the triples associated with *playsFor* were identical to those linked with *isAffiliatedTo*. Due to this redundancy,

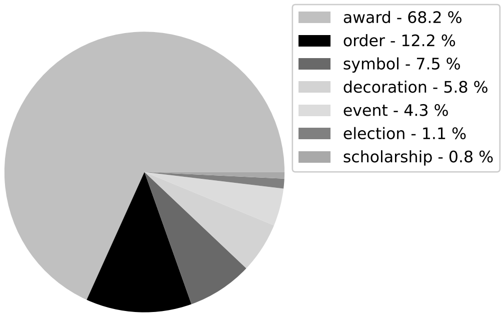


Fig. 8. Most represented values for class *scientist* with the relation *hasWonPrize* in YAGO3-10

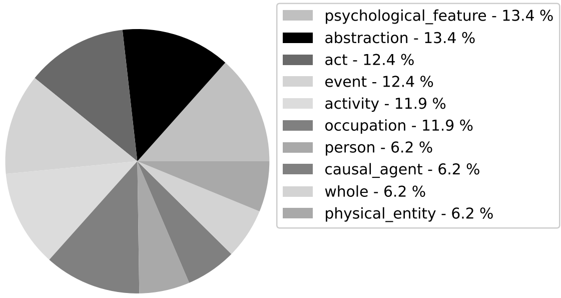


Fig. 9. Most represented values type for *person* in FB15K-237 with relation *profession*

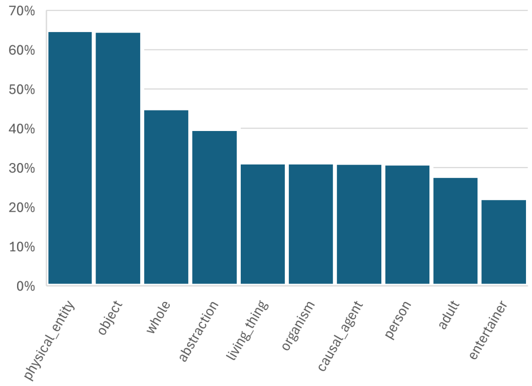


Fig. 10. Top 10 most represented entity types in FB15K-237

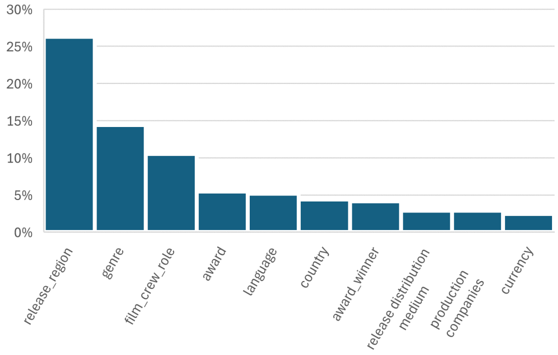


Fig. 11. Top 10 most represented relations for *film* entities in FB15K-237

only one of these relations was retained in the experiments to reduce overlap.

This process was repeated for other classes. To perform an in-depth analysis of various relations, the most represented values for a given relation (i.e. entities or values serving as the tail in the (h, r, t) triplets) were examined, with the intention of finding out the values that were prominent for specific relations. For the experiments, we considered these values, coupled with the associated relation, to serve as the entity aspects (as described in Section 5). As shown in Figure 6, for entities of type *organization* and the relation *isLocatedIn*, certain countries appeared frequently; for example, the United States accounted for 57.8% of all triples that pertained to *organization* entities with the relation *isLocatedIn*.

The different types of values associated with each entity-rela/;tion pair were also examined, as illustrated in Figures 7 and 8. This analysis was aimed at informing the design of potential processes for transforming these values. It was found to be particularly valuable in instances where the distribution of values was nearly uniform, comprising a wide range of distinct entries. By understanding the type of each value, appropriate transformation strategies could be implemented. For instance, for the relation *isAffiliated*, it was found that the most frequently represented value type was *club*. With this insight, methods to categorize the clubs based on various criteria, such as their geographical locations (e.g., *country*, *continent*...) or the specific sports they are associated with, could be conceptualized. Different experiments could be designed to capture such features as desired.

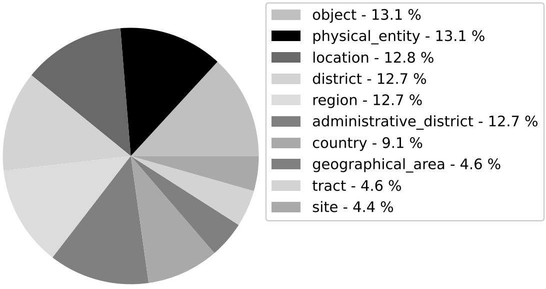


Fig. 12. Most represented values for *film* class with relation *release_region* in FB15K-237

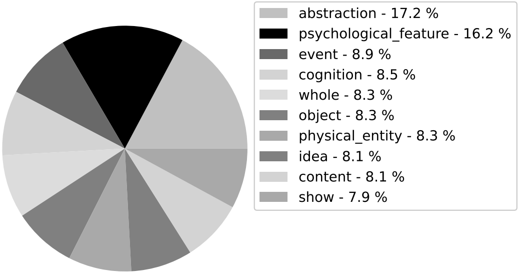


Fig. 13. Most represented values for *film* class with relation *genre* in FB15K-237

4.2. *Freebase*

Similar to Yago3-10, we conducted a statistical analysis to select features for the FB15K-237 benchmark dataset. First, the most represented classes in the dataset were identified, as shown in Figure 10 (without any distinction as per their hierarchical levels in the ontology). For each type considered, we identified the most represented relations, this is detailed in Figure 11 for *film* entities as an example. Being the most represented relations, *release_region* and *genre* were focused upon for the *film* class entities as shown in Figure 12 and Figure 13. In a different example, Figure 9 shows the most frequent types of *professions* for *person* class entities in this dataset. As with Yago3-10, this dataset study serves as a guideline for the experimental design, and similar figures were generated across various classes, relations, and values to extract the most pertinent and representative information from the dataset.

It is interesting to note here that different levels of abstraction were considered for the features of the entities while designing the experiments,. For example, for *person* entities, the relation *wasBornIn* (e.g., *wasBornIn* Paris) was found to be significant. One experiment mapped locations from specific cities to their respective countries (e.g., France), while another grouped cities by continent (e.g., Europe), allowing for evaluations across varying abstraction levels. (These experiments are presented and discussed in Section 6). This adaptable process was primarily driven by the availability of sufficient data points for the entity features within the KG. Once features were defined, entities were labeled with binary values indicating the presence or absence of each feature. This labeled data was subsequently used for SVM training in the next phase.

4.3. *LLMs for automated extraction of entity aspects*

An alternative method for deriving entity aspects from the KGs was explored using large language models due to their promise of capturing complex relationships in data. In this subsection, we give an overview of how LLMs were used for feature selection via a retrieval-augmented generation (RAG) pipeline [41].

The process began by converting the knowledge graph into plain text, where each triple was treated as a document chunk. These chunks were embedded into a vector space using the LlamaIndex framework (v0.10.28)[3], enabling the construction of a vectorized database. Relevant chunks were then retrieved to enrich the prompt provided to the LLM (Mistral-7B [37]), with prompt design playing a crucial role in guiding the model. Figure 14 shows an example of the prompt template used to steer the LLM toward extracting salient quality dimensions from the KG.

The outputs, as illustrated in Figure 15 include detailed lists of features and 2D vector projections capturing key relationships within the data. However, the LLM-generated answers were neither consistent nor explainable, which undermines the goal of obtaining human-interpretable features. This lack of consistency and clarity does not align with our objective of deriving features that are both transparent and statistically grounded. Although our experiments

---

[3] https://www.llamaindex.ai/

with LLMs and RAG were promising, they ultimately fell short of providing the reliable, understandable features needed for our analysis. We plan to revisit and expand on this automated approach in future work.

Prompt.
Context: {retrieved context}

Question: You are a useful information retrieval agent, A conceptual space is a geometric structure that represents a number of quality dimensions, which denote basic features by which concepts and objects can be compared, From the context, extract 20 quality dimensions, present the result as a list.

Fig. 14. Prompt example used for the RAG pipeline

Question. You are a useful information retrieval agent, A conceptual space is a geometric structure that represents a number of quality dimensions, which denote basic features by which concepts and objects can be compared, From the context, extract 20 quality dimensions, present the result as a list.

1. Affiliation to a football club or basketball team
2. Gender (male or female)
3. Is influenced by another person or concept
4. Influences another person or concept
5. Acts in a film or play
6. Marital status
7. Citizenship of a country or region
8. Plays for a football club or basketball team
9. Has won a prize or award
10. Is connected to an airport or location
11. Graduated from a specific university
12. Is affiliated to a specific organization, be it sports-related or not
13. Has musical role (in this case, plays the guitar)
14. Plays for a national football team
15. Is located in a country or city
16. Died in a specific location
17. Belongs to a specific political party
18. Created an album or song
19. Has a specific role in a sports team (goalkeeper, midfielder, forward)
20. Is affiliated to a football club from a different country

Question You are a useful information retrieval agent, answer the question: From the context, can you make a list of 20 concepts that represent the knowledge graph?

1. Country (mentioned multiple times)
2. Western Cape (a specific region)
3. South Africa (a country)
4. Algeria (a country)
5. African Union
6. United Nations
7. World Trade Organization
8. Arab League
9. Organisation of Islamic Cooperation
10. International Bank for Reconstruction and Development
11. International Development Association
12. International Finance Corporation
13. Multilateral Investment Guarantee Agency
14. International Centre for Settlement of Investment Disputes
15. African Development Bank
16. Arab Bank for Economic Development in Africa
17. Asian Development Bank
18. Organisation internationale de la Francophonie
19. UNESCO
20. Universal Postal Union

Fig. 15. Example results using LlamaIndex with Mistral-7B for Yago3-10

5. *InterpretE*

In this section, we present the proposed *InterpretE* approach, which aligns vector representations with entity features by manipulating vector spaces to enhance interpretability. Figure 16 illustrates the components of this ap-

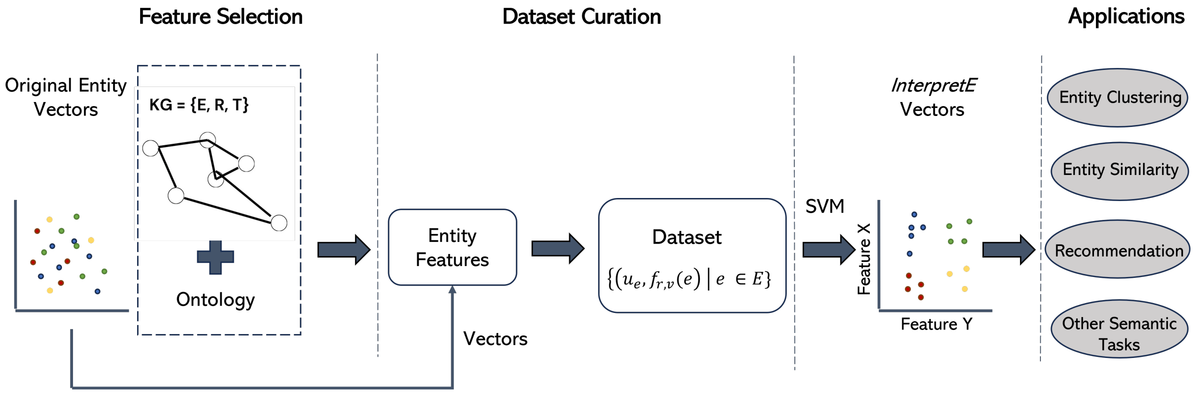


Fig. 16. Different components of *InterpretE*

proach. The method begins with feature selection, leveraging data from the KG and associated ontology to select the desired entity features that are intended to be represented in the vector space. These features can be task-specific and context-driven (e.g., distinguishing *players* from *politicians* or grouping similar professions). The main idea is to guide the entity representation based on these features, ensuring that entities with shared features are positioned close together in the final derived space. The selected entity features (suppose d), along with the original pre-trained entity vectors from a KGEM (typically having n>=100 dimensions), form the dataset. To generate interpretable embeddings, Support Vector Machine (SVM) classifiers are trained on this dataset, with the entity features as guiding labels. This process transforms the n-dimensional vectors into d-dimensional *InterpretE* vectors, where each dimension explicitly corresponds to one of the entity features (as illustrated in the figure with a 2-dimensional space featuring *Feature X* and *Feature Y*). A formal representation of the approach, including the feature selection and SVM training process, is detailed below.

5.1. Feature Selection

The *InterpretE* approach is centered around the representation of the desired aspects or features of the entities in the vector space. The purpose of the feature selection step is to extract one or more entity feature or combinations of multiple entity features present in the KG and transform the latent vectors for these entities (from a KGEM) to a human-understandable and interpretable vector space representing these feature(s). We designed several experiments with different features to test the approach, as detailed previously in Section 4. Feature selection was crucial as it guided experiment design [4].

Intuitively, the feature selection process focuses on choosing the most relevant relations and their values for each entity within a class. The most common relations per class are selected, with relations having statistically insignificant occurrences being excluded. For each selected relation, the values it takes are identified, such as specific locations for a "*isLocatedIn*" relation. Then, binary features are created for the entities in a class, indicating whether a particular value for a given relation is present or not. These binary features are concatenated in different combinations to form the feature vector, ensuring that it represents the key characteristics of the entities while remaining compact and interpretable.

Formally, given a Knowledge Graph $G = (E, R, T)$, where E is the set of entities, R is the set of relations and T is the set of triples (h, r, t) such that $h \in E$, $t \in E$, $r \in R$ and head entity h has relation r with tail entity t.[5] Also, let V_r denote the set of values that are associated with a given relation r in the set of triples T.

---

[4] Note that the attributes of the KG entities could not be considered as features since most KGEMs are not trained on them, hence such features cannot be derived from the original vectors.

[5] In some cases, there might be values instead of tail entities as t, e.g. *male*, *female* for relation *hasGender*

Let $C_{\text{set}} = \{C_1, C_2, \ldots, C_k\}$ be the set of ontological classes (e.g., persons, organizations, locations) defined by the KG ontology (as previously shown in Figures 2 and 10 for Yago and Freebase resp.). For each class $C \in C_{\text{set}}$, the entities of class C are denoted as:

$$E_C = \{e \in E \mid \text{class}(e) = C\}$$

Next, among all the relations associated with the entities of each class, a subset of the associated relations is chosen that is most representative of these entities (examples shown in Figures 3 and 11). To derive this, for each relation $r \in R$, the number of occurrences, denoted as $N(r \mid class(h) = C)$, is computed based on how frequently the relation r appears in triples where the head entity h belongs to E_C. A threshold τ is used to select significant relations for each class C, meaning that only relations with a number (of occurrences) above the threshold are considered relevant. The set of selected relations for class C is denoted as R_C, and the condition for selecting a relation r is:

$$r \in R_C \text{ if } P(r \mid class(h) = C) \geqslant \tau$$

It is important to note that the value of the threshold τ is highly dependent on the characteristics of the dataset and may vary for different relations within the dataset. Generally, the threshold was established such that values falling below this threshold were deemed irrelevant in comparison to the most frequent values. Thus, only those values exceeding the threshold were included in the analysis.

For each selected relation $r \in R_C$, the corresponding values are typically given by:

$$V_r = \{t \mid (h, r, t) \in T, h \in E_C\}$$

These values are then categorized into features, e.g., for relation *isLocatedIn* the feature categories could be specific locations such as ‘Paris’ or ‘New York’.[6]

For each entity $e \in E_C$, a binary feature is defined to indicate whether the entity has a certain value for a given relation. The feature vector is constructed by including all selected relations R_C and their associated values V_r. This is defined as:

$$f_{r,v}(e) = \begin{cases} 1, & \text{if entity } e \text{ has value } v \in V_r \text{ for relation } r \\ 0, & \text{otherwise} \end{cases}$$

This binary feature $f_{r,v}(e)$ is defined for each selected relation r and its corresponding selected values V_r. Since different relations may have different numbers of relevant values, the total number of features for an entity depends on how many categories were obtained for each relation.

Some relations may contribute multiple binary features if multiple values are important (e.g., an *isLocatedIn* relation might have multiple locations, ‘Paris’, ‘London’ and so on, as relevant value categories). Other relations might contribute fewer binary features. This variability is reflected in the construction of the feature vector, ensuring that it captures all meaningful aspects of the entity while remaining interpretable. In this way, for each class C, a set of features F_C is defined, and the feature vector for each entity $e \in E_C$ is given by:

$$f_e = \left[\bigoplus_{r \in R_C} (f_{r,v_1}(e), f_{r,v_2}(e), \ldots, f_{r,v_k}(e)) \right]$$

In this representation, the feature vector f_e is the concatenation of the binary features $f_{r,v}(e)$, where each feature corresponds to a relation r and its respective value $v \in V_r$.

---

[6] The number of selected categories of values was determined through a similar analysis to that of the threshold τ; values that were not statistically significant were omitted.

Algorithm 1 Feature Selection and Vector Derivation in *InterpretE*

```
Input: Knowledge Graph $G = (E, R, T)$, Ontological classes $C_{\text{set}}$, threshold $\tau$, Pre-trained embeddings $U$
Output: Interpretable embeddings $U'$
Initialize $W \leftarrow \emptyset$
Initialize $U' \leftarrow \emptyset$
for each class $C \in C_{\text{set}}$ do
    Extract entities $E_C = \{e \in E \mid \text{class}(e) = C\}$
    Select relations and values $R_C = \{r \in R, v \in V_r \mid P(r, v \mid \text{class}(h) = C) \geqslant \tau\}, (where\ h \in (h, r, t))$
    for each $r, v \in R_C$ do
        Derive feature vectors $f_r(e)$ for entities $e \in E_C$
        Construct dataset $D_C = \{(u_e, f_{r,v}(e)) \mid e \in E_C, u_e \in U\}$
    end for
    for each feature $f_r \in F_C$ do
        Train $\text{SVM}_{r,v}$ on $D_C$ to estimate hyperplane weight vector $w_{r,v}$
        $W \leftarrow W \bigoplus \{w_{r,v}\}$
        $u'_e(r, v) \leftarrow g(w_{r,v}, u_e)$
    end for
    $u'_e \leftarrow \bigoplus_{r,v} u'_{e(r,v)}$
    $U' \leftarrow U' \bigoplus \{u'_e\}$
end for
return $U'$
```

5.2. Dataset Curation

After selecting features, we pair the entities with their corresponding pre-trained KG embedding vectors u_e (from the KGEM). The labeled feature vector f_e forms the training dataset:

$$D_C = \{(u_e, f_e) \mid e \in E_C\}$$

The dataset D_C is then used in the subsequent phases for deriving interpretable vector spaces.

5.3. Derivation of Interpretable Vectors

After curating the dataset with the extracted features for different types of entities, the next step is to derive interpretable vectors using Support Vector Machine (SVM) classifiers. For each feature identified during the feature extraction phase, a separate SVM classifier is trained to map the pre-trained KG embedding vectors to a new interpretable vector space. This approach builds on the methodology used by Derrac et al. [15], with the goal of learning dimensions in the vector space that correspond to human-understandable features of the entities.

Although the ground truth feature vectors f_e are available for each entity, directly converting these into binary vectors would result in a significant loss of the detailed information encapsulated in the original KG embeddings u_e. Instead, we employ SVM classifiers, which allow us to leverage the continuous information from the original embeddings while learning to separate entities based on the selected features.

For each feature $f_{r,v} \in F_C$ (where F_C is the set of features defined for entities in class C), we define a binary classification problem. The binary label $y_{r,v}(e)$ for each entity e is derived from the feature function:

$$y_{r,v}(e) = f_{r,v}(e)$$

A separate SVM classifier $\text{SVM}_{r,v}$ is trained for each feature $f_{r,v}$, using the KG embedding vectors u_e as input. The objective of the SVM is to find a hyperplane that best separates the entities possessing the feature $f_{r,v}$ from those that do not. Formally, the SVM optimization problem is defined as follows:

$$(w_{r,v}, b_{r,v}) = \arg\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N} \max(0, 1 - y_{r,v}(e_i)(w \cdot u_{e_i} + b))$$

Here, $w_{r,v} \in \mathbb{R}^n$ is the weight vector corresponding to feature $f_{r,v}$, $b_{r,v}$ is the bias term, C is the regularization parameter controlling the trade-off between margin maximization and classification error, and N is the number of training examples.

The weight vector $w_{r,v}$ can be perceived as the direction in the embedding space that corresponds to feature $f_{r,v}$. These weights are used to define the hyperplane that separates entities having the feature from those that do not. The decision function associated with each hyperplane provides the signed distance between the estimated hyperplane and a given entity. This value represents the new coordinate for the corresponding feature. Specifically, the decision function for feature $f_{r,v}$ is given by:

$$u'_e(r, v) = g(w_{r,v}, b_{r,v}, u_e) \text{ where } g \text{ is the decision function and } u_e \text{ is a pretrained embedding}$$

The sign of this decision function determines whether the entity is classified as having the feature (above the hyperplane, class 1) or not (below the hyperplane, class 0). The scalar value itself is used as the new coordinate for this feature in the derived vector space, thus encoding both the presence of the feature and its relative strength in the embedding space.

The resulting weight vector $w_{r,v}$ characterizes the estimated hyperplane for feature $f_{r,v}$, and the decision function provides the corresponding coordinate for each entity.

InterpretE Vector Space. The collection of weight vectors $w_{r,v}$ associated with their biases for all features $f_{r,v} \in F_C$ defines the set of estimated hyperplanes which help to transform the embeddings in the new vector space (via the decision function):

$$W = \{(w_{r,v}, b_{r,v}) \mid f_{r,v} \in F\}$$

For each estimated hyperplane (represented by the weight vector) the new coordinates (one for each hyperplane) are computed for each entity. $u'_{e(r,v)}$ represents the coordinate linked to the value v of the relation r for the entity e. The concatenation of all coordinates forms the new vector u'_e associated to entity e:

$$u'_e \leftarrow \bigoplus_{r,v} u'_{e(r,v)}$$

Each new coordinate $u'_{e(r,v)}$ refers to a human-understandable feature, such that entities sharing common features are positioned close together in the transformed space. This makes the new vectors, referred to as *InterpretE* embeddings u'_e, more interpretable and transparent than the original KG embeddings.

The above process is described in Algorithm 1.

6. Experiments

In this section, we present experiments evaluating the efficacy of the proposed approach. We first specify the KG embedding models used in our experiments, then the implementation details for the SVM classifiers, followed by the assessment of the performance of the derived *InterpretE* embeddings in two distinct ways. We introduce metrics that capture the accuracy of the method and the consistency of the generated embedding space, providing scores and visualizations of the resulting embeddings to illustrate the results.

Table 1

κ scores on the test set and *simtop10* scores on the original and *InterpretE* vectors for different number of features (mean values) with Yago3-10 for the different KGEMs.

Number of features		ConvE	TransE	DistMult	Rescal	Complex
1	κ score	.93	.88	.92	.95	.91
	original	.182	.195	.199	.21	.206
	InterpretE	**.233**	**.223**	**.233**	**.234**	**.232**
2	κ score	.87	.83	.86	.88	.84
	original	.177	.231	.230	.240	.229
	InterpretE	**.270**	**.268**	**.270**	**.270**	**.270**
3	κ score	.62	.49	.61	.63	.6
	original	.577	.607	.640	.666	.648
	InterpretE	**.928**	**.914**	**.918**	**.924**	**.893**
4	κ score	.89	.88	.90	.91	.90
	original	.665	.695	.691	.696	.707
	InterpretE	**.814**	**.726**	**.787**	**.810**	**.824**
6	κ score	.75	.71	.75	.74	.75
	original	.635	.659	.679	.678	.648
	InterpretE	**.938**	**.888**	**.945**	**.923**	**.936**
9	κ score	.87	.83	.86	.88	.84
	original	.343	.353	.337	.347	.345
	InterpretE	**.624**	**.556**	**.624**	**.622**	**.621**

6.1. KG Embedding Models

Following previous works [29, 32], several popular and benchmark KGEMs were considered for the experiments to analyse the flexibility of the *InterpretE* approach across vector spaces generated with different methods, including ConvE [16], TransE [4], DistMult [68], Rescal [45] and Complex [63]. Although more recent embedding models have been introduced in the literature, as demonstrated by Ruffinelli et al. [52], classical embedding models remain highly competitive when paired with effective parameter optimization. Therefore, we have chosen the most widely used and popular embedding models as representative methods, obtaining the pretrained embeddings from previous work[7], where the best parameters were found using the LibKGE library [52]. It is important to note that our approach is entirely independent of the specific KGEM used and can be applied in conjunction with any pretrained model, as long as the embedding vectors can be extracted from it.

6.2. Classifier Training and Optimization

To streamline the training of the SVM classifiers, a grid search with cross-validation was performed using the Scikit-learn [46] library, which is based on LibSVM [9]. This process allowed us to automatically select the optimal hyperparameters (e.g., the regularization parameter and prevent overfitting, thereby ensuring a more generalized solution). Class imbalance, which is common in large scale KGs as well as popular benchmark datasets, was addressed by assigning weights to entities based on the distribution of positive and negative examples for each feature. This

---

[7] https://github.com/nitishajain/KGESemanticAnalysis

Table 2

κ scores on the test set and *simtop10* scores on the original and *InterpretE* vectors for representative experiments with Yago3-10 for the different KGEMs.

Experiments and features		ConvE	TransE	DistMult	Rescal	Complex
person: hasGender	κ score	1	1	1	1	.99
	original	.068	.059	.054	.061	.068
	InterpretE	**.079**	**.079**	**.079**	**.079**	**.079**
person : hasGender - wasBornIn Europe	κ score	.96	.93	.95	.96	.94
	original	.456	.496	.492	.507	.504
	InterpretE	**.540**	**.529**	**.538**	**.543**	**.539**
person : wasBornIn (Europe - Asia - North America)	κ score	.92	.84	.90	.94	.90
	original	.687	.8	.814	.871	.831
	InterpretE	**.987**	**.959**	**.983**	**.987**	**.979**
city : isLocatedIn (Europe - Asia - (North - South) America)	κ score	.94	.96	.96	.98	.98
	original	.899	.959	.949	.966	.972
	InterpretE	**.989**	**.993**	**.991**	**.996**	**.996**
scientist: hasWonPrize 6 top prizes	κ score	.96	.84	.97	.85	.98
	original	.539	.510	.575	.538	.578
	InterpretE	**.958**	**.934**	**.966**	**.926**	**.972**
person: types, player - artist - politician - scientist - officeholder - writer	κ score	.77	.75	.78	.78	.74
	original	.745	.772	.805	.794	.662
	InterpretE	**.953**	**.945**	**.958**	**.944**	**.938**
person: hasGender, wasBornIn (Europe - Asia - North America), types (player - artist - politician - scientist - officeholder)	κ score	.87	.83	.86	.88	.84
	original	.343	.353	.337	.347	.345
	InterpretE	**.624**	**.556**	**.624**	**.622**	**.621**

weighting scheme helped balance the influence of underrepresented classes in the training process. A held-out test set comprising 20% of the entities (with no overlap with the training set) was used to evaluate the performance of each SVM classifier.

6.3. Evaluation of InterpretE Vector Space

The derived *InterpretE* vector spaces are expected to yield entity vectors that are organized into clusters that align with the selected features. To assess the effectiveness of these clusters and ensure a consistent representation across different entity types, we calculated the Cohen's kappa coefficient (κ score) for the test set (following [15]). This metric evaluates the level of agreement between two sets of categorical labels, in this case, the predictions made by the trained SVM and the ground truth labels for the test entities. The κ score ranges from -1 to 1, with values closer to 1 indicating a stronger alignment between the model's classifications and the expected feature-based grouping of entities in the vector space.

The mean κ scores across various experiments on the Yago3-10 dataset are shown in Table 1, with results for FB15k-237 provided in Table 3. As discussed in Section 4, entity features were selected in a range of combinations to explore diverse configurations and capture a variety of aspects for the entities, leading to a large number of experimental configurations. To streamline presentation, these results represent the aggregated mean values of the metrics across experiments, organized by the number of selected features. For each feature count, a representative example experiment and its corresponding scores are provided in Table 2 for Yago3-10 and Table 4 for FB15k-237.

Table 3

κ scores on the test set and *simtop10* scores on the original and *InterpretE* vectors for different number of features (mean values) with FB15K-237 for the different KGEMs.

Number of features		ConvE	TransE	DistMult	Rescal	Complex
1	κ score	.90	.80	.90	.90	.85
	original	.211	.210	.214	.215	.210
	InterpretE	**.322**	**.298**	**.313**	**.322**	**.319**
2	κ score	.89	.8	.9	.9	.89
	original	.336	.329	.342	.343	.335
	InterpretE	**.484**	**.480**	**.493**	**.514**	**.509**
5	κ score	.72	.68	.72	.65	.73
	original	.561	.538	.545	.523	.547
	InterpretE	**.853**	**.844**	**.889**	**.882**	**.868**
6	κ score	.84	.73	.83	.88	.84
	original	.587	.524	.575	.563	.563
	InterpretE	**.952**	**.918**	**.936**	**.956**	**.932**

Table 4

κ scores on the test set and *simtop10* scores on the original and *InterpretE* vectors for representative experiments (mean values) with FB15K-237 for the different KGEMs.

Experiments and features		ConvE	TransE	DistMult	Rescal	Complex
person : gender - place_of_birth United States	κ score	.91	.78	.92	.92	.90
	original	.676	.689	.689	.693	.675
	InterpretE	**.909**	**.909**	**.932**	**.99**	**.977**
organizations: locations (USA - UK - Japan - Canada - Germany	κ score	.78	.70	.75	.58	.79
	original	.766	.738	.758	.731	.768
	InterpretE	**.951**	**.947**	**.958**	**.959**	**.96**
film: film_release_region (USA - Sweden - France - Spain - Finland)	κ score	.71	.69	.71	.66	.71
	original	.705	.66	.661	.621	.661
	InterpretE	**.876**	**.866**	**.903**	**.907**	**.892**
film: film genre (drama - comedy - romance - thriller - action)	κ score	.68	.65	.71	.72	.70
	original	.212	.217	.215	.217	.213
	InterpretE	**.732**	**.719**	**.805**	**.78**	**.753**

The κ values, which are close to 1 in most cases, underscore the approach's strong potential in effectively clustering entities by the selected features.

Furthermore, to clearly illustrate the advantages of the proposed approach in generating interpretable dimensions within the vector space and to compare these with the dimensions in the original KGEM vector spaces, we visualize both in a 2D space by applying Principal Component Analysis (PCA) [26]. As depicted in Figure 17, the reduced dimensions in the original KGEM space (in this case, ComplEx) fail to convey any meaningful or human-understandable representations for the entity vectors. Moreover, the *person* entities are not clustered according to

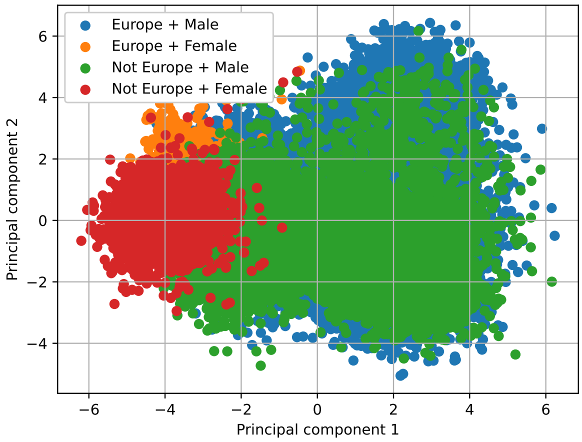


Fig. 17. 2D projection of *ComplEx* vectors for class *person* and features *hasGender* and *wasBornIn "Europe"* in Yago3-10

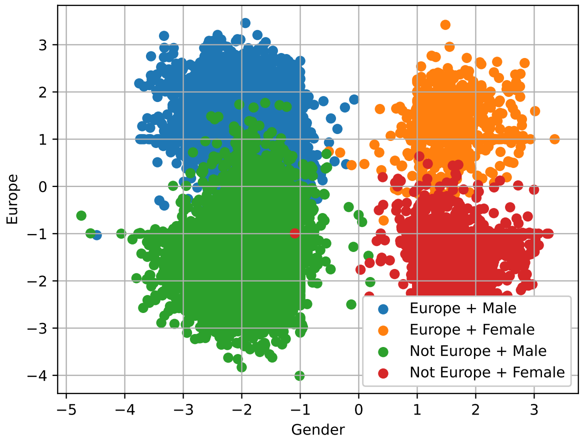


Fig. 18. 2D projection of *InterpretE* vectors for class *person* and features *hasGender* and *wasBornIn "Europe"* in Yago3-10

the *hasGender* and *was-BornIn "Europe"* features. Essentially, these vectors do not yield significant dimensions and do not facilitate the identification or clustering of entities based on specific features. In contrast, the *InterpretE* vectors derived from the ComplEx KGEM vectors as shown in Figure 18 reveal distinct clusters, with the entities within each cluster sharing common features as represented by the dimensions, i.e. they reveal distinct and meaningful clusters, dictated by human-understandable entity aspects as dimensions. We also present several other visualizations for different experiments in Figure 19 and Figure 20 that convey similar characteristics.

6.4. Evaluation of Semantic Similarity

InterpretE vectors are dictated by the selected features for the entities that they represent, as such we evaluated the semantic similarity of the derived vectors (in terms of the features) to measure this desirable characteristic.

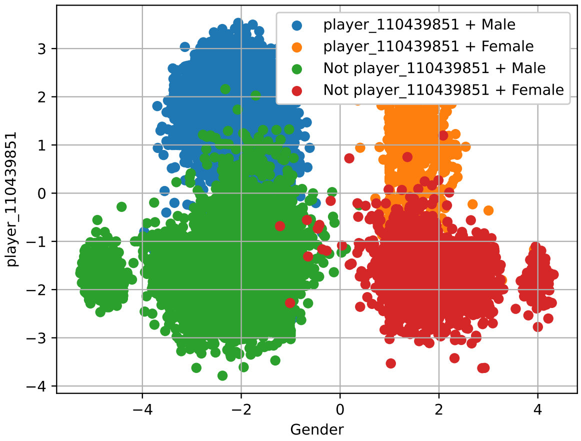


Fig. 19. 2D projection of *InterpretE* vectors for class *player* and feature *has-Gender*in Yago3-10

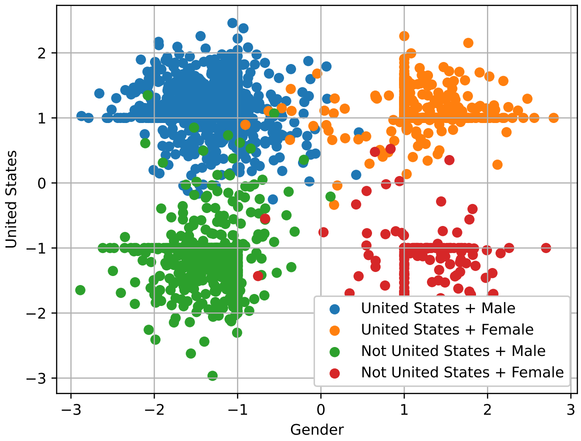
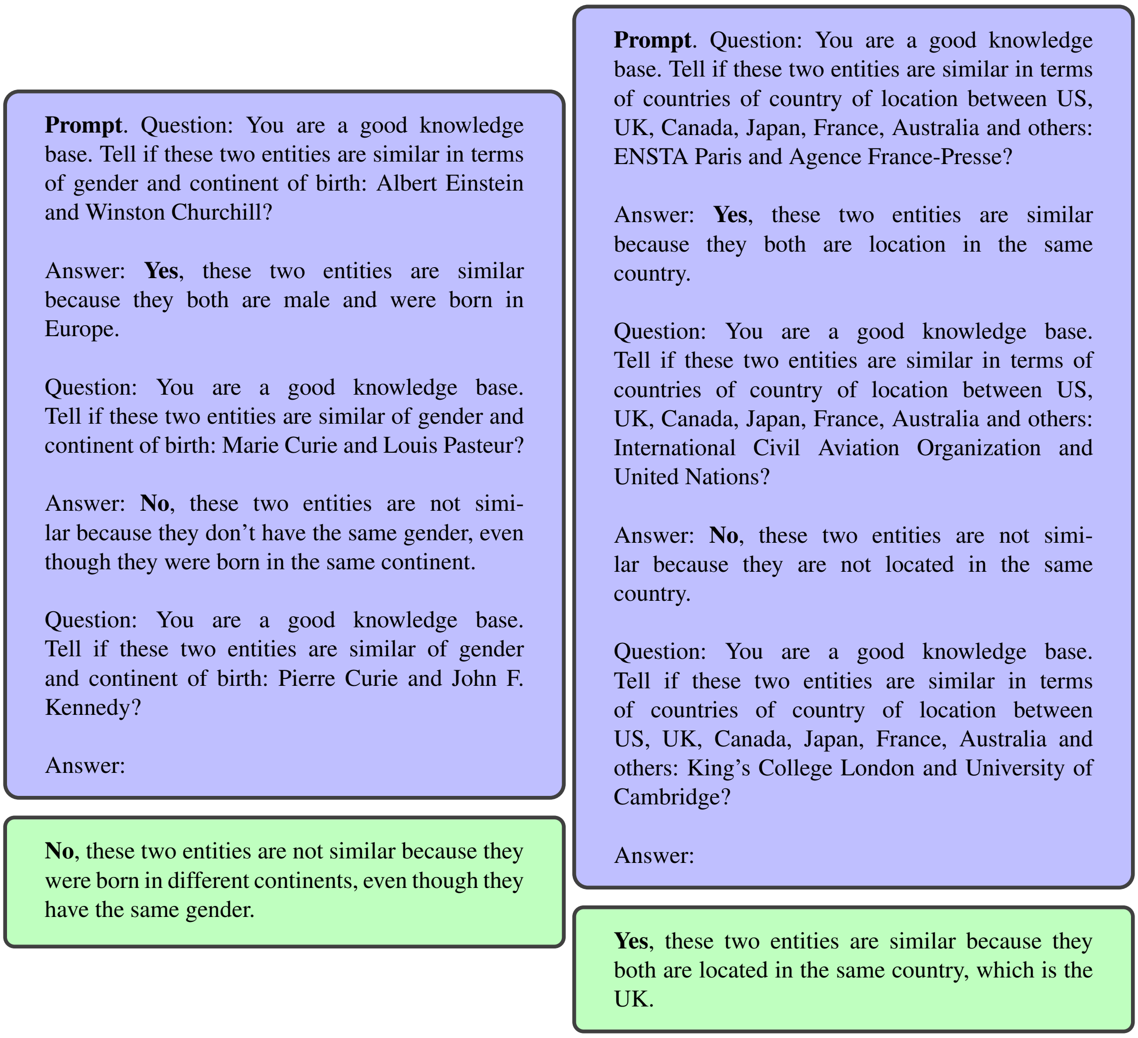


Fig. 20. 2D projection of *InterpretE* vectors for class *person* and features *gender* and *place_of_birth “United States"* in FB15k-237

We propose a simple metric *simtopk* to measure the similarity of entities' neighbors. For each entity, we analyze its neighborhood to estimate the similarity based on the corresponding feature used in the SVM experiment. The parameter k represents the number of neighbors considered. The score assigned to the original entity is calculated as the mean value of the similarities computed with these neighboring entities. This process is repeated for all entities, and the mean value of these scores is computed to serve as the final metric. The proposed *simtopk* metric can be formulated as:

$$simtopk = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{k}\sum_{j\in N_i(k)} f(n_i, n_j)\right) \tag{2}$$



Prompt. Question: You are a good knowledge base. Tell if these two entities are similar in terms of gender and continent of birth: Albert Einstein and Winston Churchill?

Answer: **Yes**, these two entities are similar because they both are male and were born in Europe.

Question: You are a good knowledge base. Tell if these two entities are similar of gender and continent of birth: Marie Curie and Louis Pasteur?

Answer: **No**, these two entities are not similar because they don't have the same gender, even though they were born in the same continent.

Question: You are a good knowledge base. Tell if these two entities are similar of gender and continent of birth: Pierre Curie and John F. Kennedy?

Answer:

No, these two entities are not similar because they were born in different continents, even though they have the same gender.

Prompt. Question: You are a good knowledge base. Tell if these two entities are similar in terms of countries of country of location between US, UK, Canada, Japan, France, Australia and others: ENSTA Paris and Agence France-Presse?

Answer: **Yes**, these two entities are similar because they both are location in the same country.

Question: You are a good knowledge base. Tell if these two entities are similar in terms of countries of country of location between US, UK, Canada, Japan, France, Australia and others: International Civil Aviation Organization and United Nations?

Answer: **No**, these two entities are not similar because they are not located in the same country.

Question: You are a good knowledge base. Tell if these two entities are similar in terms of countries of country of location between US, UK, Canada, Japan, France, Australia and others: King's College London and University of Cambridge?

Answer:

Yes, these two entities are similar because they both are located in the same country, which is the UK.

Fig. 21. Partial example of few-shot prompts with Llama 3 70B using HuggingChat

where:

n : the number of total entities; k : the number of considered neighbours; $N_i(k)$: the k closest neighbours of the i-th entity, determined using a euclidean distance; $f(\cdot, \cdot)$: returns 1 if the two entities are similar in terms of features, 0 otherwise.

The values of this metric for k=10 for the original and the derived *InterpretE* embeddings for the different experiments and the various embedding models are shown in Tables 1 and 2, for Yago3-10. The scores are better for *InterpretE* vectors as compared to the original pre-trained vectors (obtained from various KGEMs) across the board, indicating that similar entities are being represented by vectors that are closer in the new vector space, as desired. The results for FB15k-237 are presented in Tables 3 and 4. Similar to our findings with Yago3-10, we observed enhanced semantic similarity with FB15K-237. This improvement is evidenced by the higher *simtopk* value in the final space compared to the original space.

We also explored an alternative method to evaluate the *simtopk* metric by leveraging LLMs. In a limited experiment, we applied few-shot prompting with Llama3-70B [62] and a RAG pipeline with Mistral7B [37] using LlamaIndex. Our prompt included one positive and one negative example to assess the similarity between an entity and its neighborhood, as illustrated in Figure 21. Although promising, the results were inconsistent and sometimes contradictory, suggesting that further investigation is needed to develop a reliable global evaluation metric, which we plan to address in future work.

6.5. Discussion

The results from the designed experiments for each dataset demonstrate the potential of the proposed approach. However, there are several considerations for the experiment design that depend heavily on the data distributions and characteristics of the underlying KG data. For example, there is often class imbalance in entities concerning selected features (e.g., *hasGender* having more *male* representatives than *female*). These factors can impact the performance of the SVM classifier. Class-based weights have been applied to the data points to address this issue, but it remains a design challenge.

In some experiments, our method achieves a *simtopk* value very close to 1. This indicates that in the resulting space, similar entities are clustered together nearly perfectly. However, this level of clustering is not consistently observed across all experiments. The variability can be explained by the fact that other underlying features, not covered in the current experiment, could contribute to more accurately clustering similar entities. An analogy can be drawn with the well-known kernel trick used in SVMs, where additional dimensions (in our case, the consideration of new features) are introduced to better distinguish different labeled data (in this context, non-similar entities). Another challenge is the abstraction of features, especially if the underlying data is noisy and non-canonicalized (e.g., different labels for the same value such as 'UK' and 'United Kingdom'). Resolving these issues is crucial for creating useful feature categories. A potential limitation of this approach could be scalability. As the size of the knowledge graph (KG) increases, the time complexity of training the SVM also increases. The time complexity of SVM training is $O(n^2d)$, where n is the number of entities and d is the number of dimensions. Despite these challenges, *InterpretE* represents a significant step towards deriving interpretable vector spaces from KGEM vectors. It is flexible and applicable to any KGEM. We aim to further develop this approach to streamline the design and engineering process as well as improving its scalability across various datasets.

7. Conclusion and Future Work

This work attempts to address the oft overlooked issue of lack of semantic interpretability in latent spaces generated by popular KG embedding techniques. The proposed *InterpretE* approach is shown to be capable of deriving interpretable spaces from existing KGEM vectors with human-understable dimensions that are based on the features in the underlying KG. Through the design and evaluation of different experiments, we have showcased the promise of the approach for encapsulating entity features in the vectors for different feature abstraction levels, customizable as per the dataset. By aiming to bridge the gap between entity representations and human-understandable features, *InterpretE* paves the way for enhanced understanding and utilization of KGEMs in various applications. Future research can further explore the implications of this approach and extend its applicability to broader contexts within the field of knowledge representation and reasoning.

By providing interpretable insights into how entities are represented and clustered in knowledge graphs, *InterpretE* approach aims to contribute to the broader goal of AI transparency. This can allow practitioners to trace back decisions to underlying features, identify potential biases, and ensure that AI-driven systems operate in a manner that is both reliable and ethical. This focus on explainability ensures that AI models are not only accurate but also comprehensible, making them more suitable for deployment in critical decision-making contexts.

In this study, we conducted preliminary experiments employing large language models through a retrieval-augmented generation pipeline and few-shot prompting techniques to extract features as well as to assess entity similarity. While these methodologies demonstrated potential, the outcomes were marked by inconsistency and a lack of transparency, falling short of our objective to derive human-interpretable and statistically robust features. Consequently, we have prioritized deterministic, statistical approaches in our current analysis. Nonetheless, we recognize the evolving capabilities of LLMs and intend to explore their application further as the research advances. Furthermore, it would be interesting explore whether sparse autoencoders, as used by Cunningham et al. [28] to identify monosemantic features in LLMs, can be applied to KGEMs to derive more interpretable entity representations.

Acknowledgements.

This work was partly funded by the HE project MuseIT, which has been co-founded by the European Union under the Grant Agreement No 101061441. Views and opinions expressed are, however, those of the authors and do not necessarily reflect those of the European Union or European Research Executive Agency. We are also thankful for access to King's Computational Research, Engineering and Technology Environment (CREATE). King's College London. (2024). Retrieved October 28, 2024, from https://doi.org/10.18742/rnvf-m076.

References

[1] F. Alshargi, S. Shekarpour, T. Soru and A. Sheth, Concept2vec: Metrics for evaluating quality of embeddings for ontological concepts, *Spring Symposium on Combining Machine Learning with Knowledge Engineering (AAAI-MAKE 2019)* (2019). https://doi.org/10.48550/arXiv.1803.04488.

[2] A.B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins et al., Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI, *Information fusion* **58** (2020), 82–115.

[3] J. Baek, A.F. Aji, J. Lehmann and S.J. Hwang, Direct Fact Retrieval from Knowledge Graphs without Entity Linking, in: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, A. Rogers, J. Boyd-Graber and N. Okazaki, eds, Association for Computational Linguistics, Toronto, Canada, 2023, pp. 10038–10055. doi:10.18653/v1/2023.acl-long.558. https://aclanthology.org/2023.acl-long.558/.

[4] A. Bordes, N. Usunier, A. Garcia-Durán, J. Weston and O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, Curran Associates Inc., Red Hook, NY, USA, 2013, pp. 2787–2795–.

[5] A. Boschin, N. Jain, G. Keretchashvili and F.M. Suchanek, Combining embeddings and rules for fact prediction, in: *International Research School in Artificial Intelligence in Bergen*, 2022.

[6] Z. Bouraoui, V. Gutiérrez-Basulto and S. Schockaert, Integrating Ontologies and Vector Space Embeddings Using Conceptual Spaces, in: *International Research School in Artificial Intelligence in Bergen (AIB 2022)*, C. Bourgaux, A. Ozaki and R. Peñaloza, eds, Open Access Series in Informatics (OASIcs), Vol. 99, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 2022, pp. 3:1–3:30. ISSN 2190-6807. ISBN 978-3-95977-228-0. doi:10.4230/OASIcs.AIB.2022.3.

[7] Z. Bouraoui, J. Camacho-Collados, L. Espinosa-Anke and S. Schockaert, Modelling semantic categories using conceptual neighborhood, in: *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34, 2020, pp. 7448–7455.

[8] J. Cao, J. Fang, Z. Meng and S. Liang, Knowledge graph embedding: A survey from the perspective of representation spaces, *ACM Computing Surveys* **56**(6) (2024), 1–42.

[9] C.-C. Chang and C.-J. Lin, LIBSVM: A library for support vector machines, *ACM Trans. Intell. Syst. Technol.* **2**(3) (2011). doi:10.1145/1961189.1961199.

[10] U. Chatterjee, A. Gajbhiye and S. Schockaert, Cabbage Sweeter than Cake? Analysing the Potential of Large Language Models for Learning Conceptual Spaces, in: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023, pp. 11836–11842.

[11] J. Chen, P. Hu, E. Jimenez-Ruiz, O.M. Holter, D. Antonyrajah and I. Horrocks, Owl2vec*: Embedding of owl ontologies, *Machine Learning* **110**(7) (2021), 1813–1845.

[12] J. Cohen, A coefficient of agreement for nominal scales, *Educational and psychological measurement* **20**(1) (1960), 37–46.

[13] Y. Dai, S. Wang, N.N. Xiong and W. Guo, A survey on knowledge graph embedding: Approaches, applications and benchmarks, *Electronics* **9**(5) (2020), 750.

[14] C. d'Amato, N.F. Quatraro and N. Fanizzi, Injecting Background Knowledge into Embedding Models for Predictive Tasks on Knowledge Graphs, in: *ESWC*, 2021.

[15] J. Derrac and S. Schockaert, Inducing semantic relations from conceptual spaces: a data-driven approach to plausible reasoning, *Artificial Intelligence* **228** (2015), 66–94.

[16] T. Dettmers, P. Minervini, P. Stenetorp and S. Riedel, Convolutional 2D Knowledge Graph Embeddings, 2018.

[17] L. Dietz, A. Kotov and E. Meij, Utilizing Knowledge Graphs for Text-Centric Information Retrieval, in: *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, Association for Computing Machinery, New York, NY, USA, 2018, pp. 1387–1390–. ISBN 9781450356572. doi:10.1145/3209978.3210187.

[18] M.H. Gad-Elrab, D. Stepanova, T.-K. Tran, H. Adel and G. Weikum, Excut: Explainable embedding-based clustering over knowledge graphs, in: *International Semantic Web Conference*, Springer, 2020, pp. 218–237.

[19] D. Garg, S. Ikbal, S.K. Srivastava, H. Vishwakarma, H.P. Karanam and L.V. Subramaniam, Quantum Embedding of Knowledge for Reasoning, in: *Neurips*, 2019, pp. 5595–5605.

[20] X. Ge, Y.C. Wang, B. Wang, C.-C.J. Kuo et al., Knowledge graph embedding: An overview, *APSIPA Transactions on Signal and Information Processing* **13**(1) (2024).

[21] N. Guan, D. Song and L. Liao, Knowledge graph embedding with concepts, *Knowledge-Based Systems* **164** (2019), 38–44.

[22] P. Gärdenfors, *Conceptual Spaces: The Geometry of Thought*, The MIT Press, 2000. ISBN 9780262273558. doi:10.7551/mitpress/2076.001.0001.

[23] J. Hao, M. Chen, W. Yu, Y. Sun and W. Wang, Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts, in: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 1709–1719.

[24] J. Hao, M. Chen, W. Yu, Y. Sun and W. Wang, Universal Representation Learning of Knowledge Bases by Jointly Embedding Instances and Ontological Concepts, in: *KDD*, 2019, pp. 1709–1719.

[25] O.M. Holter, E.B. Myklebust, J. Chen and E. Jimenez-Ruiz, Embedding OWL ontologies with OWL2Vec, *CEUR Workshop Proceedings* **2456** (2019), 33–36. http://ceur-ws.org/Vol-2456/.

[26] H. Hotelling, Analysis of a complex of statistical variables into principal components., *Journal of educational psychology* **24**(6) (1933), 417.

[27] S. Hou and D. Wei, Research on Knowledge Graph-Based Recommender Systems, in: *2023 3rd International Symposium on Computer Technology and Information Science (ISCTIS)*, 2023, pp. 737–742. doi:10.1109/ISCTIS58954.2023.10213083.

[28] R. Huben, H. Cunningham, L.R. Smith, A. Ewart and L. Sharkey, Sparse autoencoders find highly interpretable features in language models, in: *The Twelfth International Conference on Learning Representations*, 2023.

[29] N. Hubert, H. Paulheim, A. Brun and D. Monticolo, Do similar entities have similar embeddings?, in: *European Semantic Web Conference*, Springer, 2024, pp. 3–21.

[30] F. Ilievski, K. Shenoy, H. Chalupsky, N. Klein and P. Szekely, A study of concept similarity in Wikidata, *Semantic Web* (2024), 1–20. doi:10.3233/SW-233520.

[31] M. Jackermeier, J. Chen and I. Horrocks, Dual box embeddings for the description logic EL++, in: *Proceedings of the ACM on Web Conference 2024*, 2024, pp. 2250–2258.

[32] N. Jain, J.-C. Kalo, W.-T. Balke and R. Krestel, Do Embeddings Actually Capture Knowledge Graph Semantics?, in: *The Semantic Web*, R. Verborgh, K. Hose, H. Paulheim, P.-A. Champin, M. Maleshkova, O. Corcho, P. Ristoski and M. Alam, eds, Springer International Publishing, Cham, 2021, pp. 143–159. ISBN 978-3-030-77385-4.

[33] N. Jain, T.-K. Tran, M.H. Gad-Elrab and D. Stepanova, Improving knowledge graph embeddings with ontological reasoning, in: *International Semantic Web Conference*, Springer, 2021, pp. 410–426.

[34] M. Jayathilaka, T. Mu and U. Sattler, Ontology-based n-ball concept embeddings informing few-shot image classification, *arXiv preprint arXiv:2109.09063* (2021).

[35] M. Jayathilaka, T. Mu and U. Sattler, Towards knowledge-aware few-shot learning with ontology-based n-ball concept embeddings, in: *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, IEEE, 2021, pp. 292–297.

[36] G. Ji, S. He, L. Xu, K. Liu and J. Zhao, Knowledge Graph Embedding via Dynamic Mapping Matrix, in: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, C. Zong and M. Strube, eds, Association for Computational Linguistics, Beijing, China, 2015, pp. 687–696. doi:10.3115/v1/P15-1067. https://aclanthology.org/P15-1067.

[37] A.Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D.S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L.R. Lavaud, M.-A. Lachaux, P. Stock, T.L. Scao, T. Lavril, T. Wang, T. Lacroix and W.E. Sayed, Mistral 7B, 2023.

[38] J.-C. Kalo, P. Ehler and W.-T. Balke, Knowledge graph consolidation by unifying synonymous relationships, in: *The Semantic Web–ISWC 2019: 18th International Semantic Web Conference, Auckland, New Zealand, October 26–30, 2019, Proceedings, Part I 18*, Springer, 2019, pp. 276–292.

[39] T.N. Kipf and M. Welling, Semi-Supervised Classification with Graph Convolutional Networks, in: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[40] D. Krompaß, S. Baier and V. Tresp, Type-Constrained Representation Learning in Knowledge Graphs, in: *ISWC*, 2015, pp. 640–655.

[41] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel and D. Kiela, Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, in: *Advances in Neural Information Processing Systems*, Vol. 33, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan and H. Lin, eds, Curran Associates, Inc., 2020, pp. 9459–9474.

[42] Y. Lin, Z. Liu, M. Sun, Y. Liu and X. Zhu, Learning entity and relation embeddings for knowledge graph completion, in: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, AAAI Press, 2015, pp. 2181–2187–. ISBN 0262511290.

[43] F. Mahdisoltani, J. Biega and F.M. Suchanek, Yago3: A knowledge base from multilingual wikipedias, in: *CIDR*, 2013.

[44] W.J. Murdoch, C. Singh, K. Kumbier, R. Abbasi-Asl and B. Yu, Definitions, methods, and applications in interpretable machine learning, *Proceedings of the National Academy of Sciences* **116**(44) (2019), 22071–22080. doi:10.1073/pnas.1900654116.

[45] M. Nickel, V. Tresp and H.-P. Kriegel, A Three-Way Model for Collective Learning on Multi-Relational Data, in: *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, L. Getoor and T. Scheffer, eds, ICML '11, ACM, New York, NY, USA, 2011, pp. 809–816. ISBN 978-1-4503-0619-5.

[46] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, Scikit-learn: Machine Learning in Python, *Journal of Machine Learning Research* **12** (2011), 2825–2830.

[47] X. Peng, Z. Tang, M. Kulmanov, K. Niu and R. Hoehndorf, Description logic EL++ embeddings with intersectional closure, *arXiv preprint arXiv:2202.14018* (2022).

[48] M. Porta, Interpretability, Oxford University Press, 2016. ISBN 9780199390069. doi:10.1093/acref/9780199976720.013.2221. https://www.oxfordreference.com/view/10.1093/acref/9780199976720.001.0001/acref-9780199976720-e-2221.

[49] P. Ristoski and H. Paulheim, Rdf2vec: Rdf graph embeddings for data mining, in: *International semantic web conference*, Springer, 2016, pp. 498–514.

[50] A. Rossi, D. Barbosa, D. Firmani, A. Matinata and P. Merialdo, Knowledge graph embedding for link prediction: A comparative analysis, *ACM Transactions on Knowledge Discovery from Data (TKDD)* **15**(2) (2021), 1–49.

[51] A. Rossi, D. Barbosa, D. Firmani, A. Matinata and P. Merialdo, Knowledge Graph Embedding for Link Prediction: A Comparative Analysis, *ACM Trans. Knowl. Discov. Data* **15**(2) (2021). doi:10.1145/3424672.

[52] D. Ruffinelli, S. Broscheit and R. Gemulla, You CAN Teach an Old Dog New Tricks! On Training Knowledge Graph Embeddings, in: *International Conference on Learning Representations*.

[53] T. Safavi and D. Koutra, CoDEx: A Comprehensive Knowledge Graph Completion Benchmark, in: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, B. Webber, T. Cohn, Y. He and Y. Liu, eds, Association for Computational Linguistics, Online, 2020, pp. 8328–8350. doi:10.18653/v1/2020.emnlp-main.669. https://aclanthology.org/2020.emnlp-main.669.

[54] A. Saxena, A. Tripathi and P. Talukdar, Improving Multi-hop Question Answering over Knowledge Graphs using Knowledge Base Embeddings, in: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, D. Jurafsky, J. Chai, N. Schluter and J. Tetreault, eds, Association for Computational Linguistics, Online, 2020, pp. 4498–4507. doi:10.18653/v1/2020.acl-main.412. https://aclanthology.org/2020.acl-main.412.

[55] S. Schramm, C. Wehner and U. Schmid, Comprehensible Artificial Intelligence on Knowledge Graphs: A survey, *Journal of Web Semantics* **79** (2023), 100806. doi:https://doi.org/10.1016/j.websem.2023.100806. https://www.sciencedirect.com/science/article/pii/S1570826823000355.

[56] A. Simhi and S. Markovitch, Interpreting Embedding Spaces by Conceptualization, 2023.

[57] F.Z. Smaili, X. Gao and R. Hoehndorf, Onto2vec: joint vector-based representation of biological entities and their ontology-based annotations, *Bioinformatics* **34**(13) (2018), i52–i60.

[58] F.Z. Smaili, X. Gao and R. Hoehndorf, OPA2Vec: combining formal and informal content of biomedical ontologies to improve similarity-based prediction, *Bioinformatics* **35**(12) (2019), 2133–2140.

[59] Z. Sun, Q. Zhang, W. Hu, C. Wang, M. Chen, F. Akrami and C. Li, A Benchmarking Study of Embedding-based Entity Alignment for Knowledge Graphs, *Proceedings of the VLDB Endowment* **13**(11) (2020).

[60] Z. Sun, Q. Zhang, W. Hu, C. Wang, M. Chen, F. Akrami and C. Li, A benchmarking study of embedding-based entity alignment for knowledge graphs, *arXiv preprint arXiv:2003.07743* (2020).

[61] K. Toutanova and D. Chen, Observed versus latent features for knowledge base and text inference, in: *Workshop on Continuous Vector Space Models and their Compositionality*, 2015. https://api.semanticscholar.org/CorpusID:5378837.

[62] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar et al., Llama: Open and efficient foundation language models, *arXiv preprint arXiv:2302.13971* (2023).

[63] T. Trouillon, J. Welbl, S. Riedel, E. Gaussier and G. Bouchard, Complex Embeddings for Simple Link Prediction, in: *Proceedings of The 33rd International Conference on Machine Learning*, M.F. Balcan and K.Q. Weinberger, eds, Proceedings of Machine Learning Research, Vol. 48, PMLR, New York, New York, USA, 2016, pp. 2071–2080. https://proceedings.mlr.press/v48/trouillon16.html.

[64] Q. Wang, Z. Mao, B. Wang and L. Guo, Knowledge graph embedding: A survey of approaches and applications, *IEEE transactions on knowledge and data engineering* **29**(12) (2017), 2724–2743.

[65] Z. Wang, J. Zhang, J. Feng and Z. Chen, Knowledge graph embedding by translating on hyperplanes, in: *Proceedings of the AAAI conference on artificial intelligence*, Vol. 28, 2014.

[66] K. Wiharja, J.Z. Pan, M.J. Kollingbaum and Y. Deng, Schema aware iterative Knowledge Graph completion, *J. Web Semant.* **65** (2020), 100616.

[67] B. Xiong, N. Potyka, T.-K. Tran, M. Nayyeri and S. Staab, Faithful embeddings for EL++ knowledge bases, in: *International Semantic Web Conference*, Springer, 2022, pp. 22–38.

[68] B. Yang, S.W.-t. Yih, X. He, J. Gao and L. Deng, Embedding Entities and Relations for Learning and Inference in Knowledge Bases, in: *Proceedings of the International Conference on Learning Representations (ICLR) 2015*, 2015.

[69] H. Yang, J. Chen and U. Sattler, TransBox: EL++-closed Ontology Embedding, in: *Proceedings of the ACM Web Conference 2025*, Association for Computing Machinery, United States, 2025.

[70] M. Yasunaga, H. Ren, A. Bosselut, P. Liang and J. Leskovec, QA-GNN: Reasoning with Language Models and Knowledge Graphs for Question Answering, in: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tur, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty and Y. Zhou, eds, Association for Computational Linguistics, Online, 2021, pp. 535–546. doi:10.18653/v1/2021.naacl-main.45. https://aclanthology.org/2021.naacl-main.45/.

[71] H. Zhao, H. Chen, F. Yang, N. Liu, H. Deng, H. Cai, S. Wang, D. Yin and M. Du, Explainability for Large Language Models: A Survey, *ACM Trans. Intell. Syst. Technol.* (2024). doi:10.1145/3639372.

[72] X. Zhu, C. Xu and D. Tao, Where and What? Examining Interpretable Disentangled Representations, in: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 5857–5866. doi:10.1109/CVPR46437.2021.00580.

[73] K. Ziegler, O. Caelen, M. Garchery, M. Granitzer, L. He-Guelton, J. Jurgovsky, P. Portier and S. Zwicklbauer, Injecting Semantic Background Knowledge into Neural Networks using Graph Embeddings, in: *26th IEEE, WETICE*, 2017, pp. 200–205.