# Integrating Knowledge Graph embedding and pretrained Language Models in Hypercomplex Spaces

**Mojtaba Nayyeri**[*]
University of Stuttgart
mojtaba.nayyeri@ipvs.uni-stuttgart.de

**Zihao Wang**[*]
University of Stuttgart
zihao.wang@ipvs.uni-stuttgart.de

**Mst. Mahfuja Akter**[*]
University of Bonn
mahfuja.ruby85@gmail.com

**Mirza Mohtashim Alam**
Institute for Applied Informatics Asociation e.V. (InfAI)
turzo.mohtasim@gmail.com

**Md Rashad Al Hasan Rony**
Fraunhofer IAIS
rashad.rony@iais.fraunhofer.de

**Jens Lehmann**
Institute for Applied Informatics Asociation e.V. (InfAI)
jens.lehmann@cs.uni-bonn.de

**Steffen Staab**
University of Stuttgart
steffen.staab@ipvs.uni-stuttgart.de

## Abstract

Knowledge Graphs, such as Wikidata, comprise structural and textual knowledge in order to represent knowledge. For each of the two modalities dedicated approaches for graph embedding and language models learn patterns that allow for predicting novel structural knowledge. Few approaches have integrated learning and inference with both modalities and these existing ones could only partially exploit the interaction of structural and textual knowledge. In our approach, we build on existing strong representations of single modalities and we use hypercomplex algebra to represent both, (i), single-modality embedding as well as, (ii), the interaction between different modalities and their complementary means of knowledge representation. More specifically, we suggest Dihedron and Quaternion representations of 4D hypercomplex numbers to integrate four modalities namely structural knowledge graph embedding, word-level representations (e.g. Word2vec, Fasttext), sentence-level representations (Sentence transformer), and document-level representations (sentence transformer, Doc2vec). Our unified vector representation scores the plausibility of labelled edges via Hamilton and Dihedron products, thus modeling pairwise interactions between different modalities. Extensive experimental evaluation on standard benchmark datasets shows the superiority of our two new models using abundant textual information besides sparse structural knowledge to enhance performance in link prediction tasks.

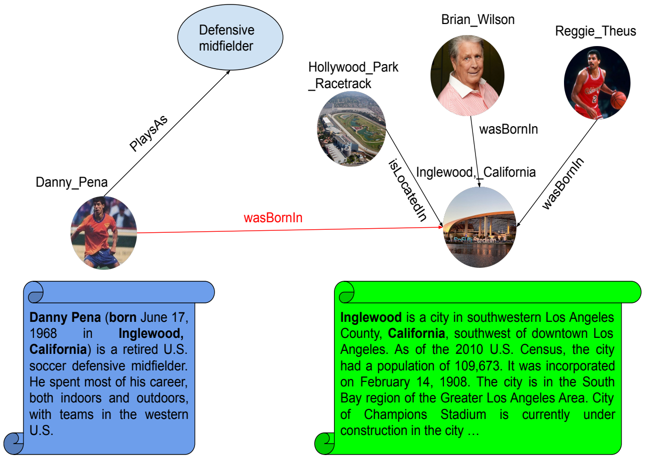
Keywords: Knowledge Graph Embedding; pretrained Language Model; Textual Information

## 1 Introduction

Knowledge Graphs (KGs) have become a core component of many AI systems ranging from question answering and named entity recognition to recommender systems [1–3]. KGs represent knowledge in the form of multi-relational directed labeled graphs where nodes with labels may represent entities, e.g. *"Danny Pena"* and labeled edges represent relations between entities, e.g. *wasBornIn* . In

[*] Equal contribution.

Preprint. Preliminary work.



**Danny Pena** (**born** June 17, 1968 in **Inglewood, California**) is a retired U.S. soccer defensive midfielder. He spent most of his career, both indoors and outdoors, with teams in the western U.S.

**Inglewood** is a city in southwestern Los Angeles County, **California**, southwest of downtown Los Angeles. As of the 2010 U.S. Census, the city had a population of 109,673. It was incorporated on February 14, 1908. The city is in the South Bay region of the Greater Los Angeles Area. City of Champions Stadium is currently under construction in the city ...

**Figure 1:** Knowledge Graph with textual description of entities. The entity "Danny Pena" lacks proper structural information. However, it contains rich textual description, which may help to predict the place of birth.

this way, a fact may be represented as a triple, *(node, edge, node)*, e.g. *(Danny Pena, wasBornIn, Inglewood California)*.

To enable machine learning to act on KGs with symbolic information [4], Knowledge Graph embedding (KGE) maps each node and each edge of a KG into a low dimensional vector (embedding) space. Such embedding, which are assumed to capture semantic and structural knowledge, support machine learning tasks such as link prediction, entity linking, or question answering. Although KGs contain many facts, they are still highly incomplete when compared to which facts hold in the world, which adversely affects the performance of KGE in downstream tasks.

Consider Figure 1 and assume that the red link (Danny Pena, wasBornIn, Inglewood,_California) is unknown because the entity "Danny Pena" is only connected to one other entity, "Defensive Midfielder". While graph structural information alone cannot help to bridge from "Danny Pena" to "Inglewood_California", a second modality, such as further textual knowledge (e.g. from Wikipedia[2]) might come to the rescue.

Previous work such as DKRL [5] and ConMask [6] are among the KGE models which go beyond structural graph knowledge and also incorporate textual information for link prediction. These models employ deep learning methods such as convolutional neural network (CNN) and attention mechanism to map the textual information into the same space of KGE. Therefore the link prediction task takes advantage of both structural information from KGE and semantic information from text. However, these models cannot benefit from recent pretrained language models such as BERT [7].

Another line of work [8–11] propose different approaches to incorporate pretrained language models into KGE models. They provide different approaches that incorporate KGE and masked language models by unifying the two loss functions for each of the models. Although effective, the mentioned work considering pretrained language models only focus on a single representation (e.g., word representation) with a single pretrained language model (e.g., Word2Vec). However, integration of various representations of textual information (e.g., various features of the same text captured by different pretrained language models aiming at word, sentence and document embedding) together with structural information can further enhance the performance of KGE models. Such an integration requires sophisticated mathematical tools to efficiently model all representations and their pairwise correlations.

In this paper, we employ a 4D space of hypercomplex numbers to integrate the knowledge from KGs and text in a unified representation. We first exploit a variety of pretrained language models to extract

[2] https://en.wikipedia.org/wiki/Danny_Pena

different levels of representations from entity descriptions. Then we integrate these representations of textual information with structural information of KGE to obtain a rich and unified 4D representation for each entity in a hypercomplex space. Specifically, we employ Dihedron and Quaternion as 4D hypercomplex spaces to represent all of the mentioned representations uniformly. Finally, we conduct fruitful experiments, ablation studies and analysis on various datasets to confirm the advantage of incorporating different pretrained language models into KGE models.

## 2 Related Work

### 2.1 Structural Knowledge Graph Embedding Models

We first discuss KGE models considering only structural information of KG. TransE [12] is the first KGE work that introduces a distance-based scoring function in the real vector space. It minimizes the distance between the tail entity and the query formulated by the addition of the head entity and relation. TransH [13] further enhances TransE by projecting entities to relation-specific hyperplanes, and thus an entity has different representations w.r.t. different relations.

ComplEx [14] is the first work modeling the KGE beyond the real space. It uses the Hermitian dot product of complex space to model the scoring function. With the same set of head and tail entities, the non-symmetric relations have different scores given different orders of entities in the complex space. RotatE [15] further models relation compositions where multiple individual relations combine to a unique relation. It naturally models relations as the rotation in the complex space.

Moreover, AttE and AttH [16] employ an attention mechanism on KGE in the Euclidean and hyperbolic space. QuatE [17] and Dihedral [18] separately model KGE with Quaternion and Dihedron. These novel geometry-aware KGE models further enhance the representation of KGE.

### 2.2 Text-enhanced Knowledge Graph Embedding Models

There are also a variety of KGE works considering both textual information such as description and name of entities and structural information of KG. As an early work, DKRL [5] enhances the TransE model by considering the entity descriptions. The entity descriptions are encoded by a CNN and jointly optimized together with the scoring function of TransE. ConMask [6] deals with the issue that an entity may involve multiple relations. It extracts relation-specific information from the entity description by training a well-designed network that computes attention over entity descriptions.

Many recent approaches exploit pretrained language models such as BERT to further enhance the performance of KGE models. KG-Bert [8] is the first model that utilizes BERT. It regards triples in the KG as a sequence of text and then it obtains the representation of triplets with BERT. PretrainKGE [9] extracts knowledge from the BERT by representing the entity descriptions and relation names with the embedding from BERT. The embedding from the BERT can be utilized to further enhance different algorithms of KGE. More recent approaches [10, 11] also explore different methods to combine pretrained language models and KGE.

## 3 Preliminaries

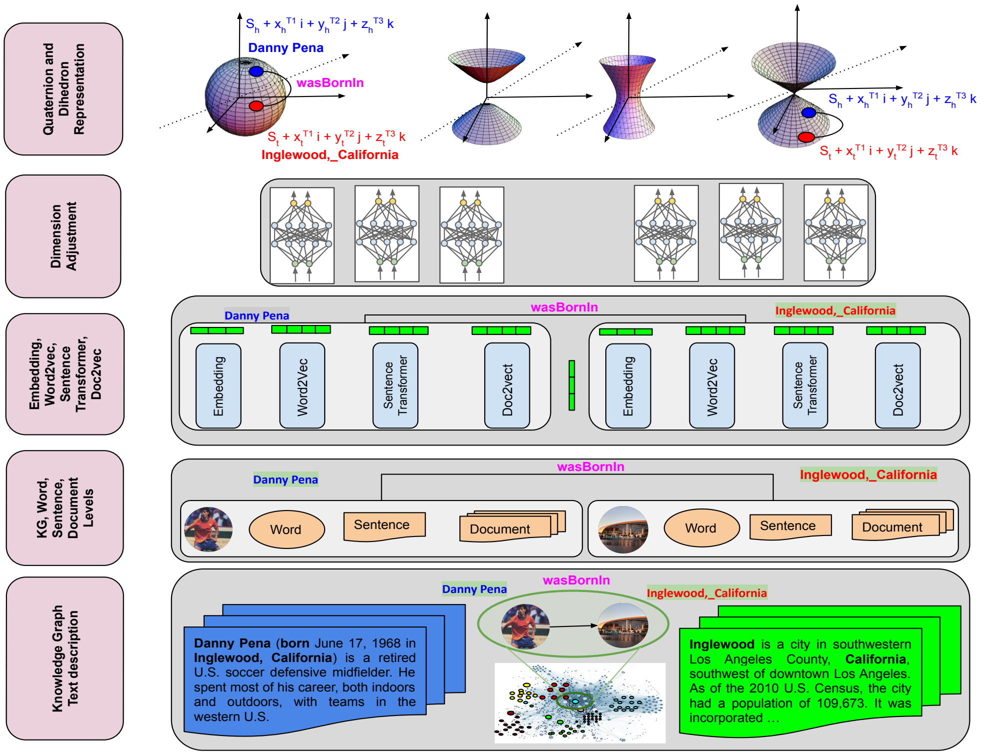
In this section, we present the preliminaries required for proposing our models.

$\boldsymbol{Knowledge\ \ Graph}$ **:** A knowledge graph is a set of triples $\mathcal{K} = \{(h, r, t) | h, t \in \mathcal{E}, r \in \mathcal{R}\} \subset \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ where $\mathcal{E}, \mathcal{R}$ are the set of all entities and relations in a KG respectively.

$\boldsymbol{Textual\ \ Knowledge\ \ Graph}$ **:** For a KG $\mathcal{K}$, we can collect word, sentence or documents for each given node/relations and construct the textual KG $\mathcal{TK}$ as follows

$$\mathcal{TK} = \{(h^T, r^T, t^T) | h^T, t^T \in \mathcal{E}^T, r^T \in \mathcal{R}^T\} \subset \mathcal{E}^T \times \mathcal{R}^T \times \mathcal{E}^T, \tag{1}$$

where $h^T, r^T, t^T$ are the word ($T = W$), sentence ($T = S$) or document ($T = D$) representation of entities and relations. For example, $h = Berlin$ is an entity in the KG that has the word representation $h^W = {}''Berlin''$, sentence representation $h^S =$ *"Berlin is the capital and largest city of Germany by both area and population"*, and document representation $h^D =$ *"Berlin is the capital and largest city of Germany by both area and population. Its 3.7 million inhabitants make it the European*


Quaternion and Dihedron Representation
$S_h + x_h^{T1} i + y_h^{T2} j + z_h^{T3} k$
Danny Pena
wasBornIn
$S_t + x_t^{T1} i + y_t^{T2} j + z_t^{T3} k$
Inglewood,_California
$S_h + x_h^{T1} i + y_h^{T2} j + z_h^{T3} k$
$S_t + x_t^{T1} i + y_t^{T2} j + z_t^{T3} k$
Dimension Adjustment
Embedding, Word2vec, Sentence Transformer, Doc2vec
Danny Pena
wasBornIn
Inglewood,_California
Embedding
Word2Vec
Sentence Transformer
Doc2vect
Embedding
Word2Vec
Sentence Transformer
Doc2vect
KG, Word, Sentence, Document Levels
Danny Pena
wasBornIn
Inglewood,_California
Word
Sentence
Document
Word
Sentence
Document
Knowledge Graph Text description
Danny Pena
wasBornIn
Inglewood,_California

**Danny Pena** (**born** June 17, 1968 in **Inglewood, California**) is a retired U.S. soccer defensive midfielder. He spent most of his career, both indoors and outdoors, with teams in the western U.S.

**Inglewood** is a city in southwestern Los Angeles County, **California**, southwest of downtown Los Angeles. As of the 2010 U.S. Census, the city had a population of 109,673. It was incorporated ...

**Figure 2:** The overall architecture of our proposed model. The architecture considers different representations of entity description in word, sentence, and document levels and maps them into the same space by adjusting their dimensions. Then different representations are unified by Dihedron or Quaternion into the geometric space of Hyperbolic or sphere.

*Union's most populous city, according to the population within city limits. One of Germany's sixteen constituent states, Berlin ...".*

***Knowledge Graph Embedding*** **:** A KG $\mathcal{K}$ can be embedded into a (low-dimensional) vector representation, shown as $\mathcal{KGE} = \{(\mathbf{h}, \mathbf{r}, \mathbf{t}) | \mathbf{h}, \mathbf{t} \in \mathbf{E}, \mathbf{r} \in \mathbf{R}\} \subset \mathbf{E} \times \mathbf{R} \times \mathbf{E}$ where $\mathbf{E}, \mathbf{R}$ are the embedding sets of entities and relations in the KG which are $n_e \times d_e$ and $n_r \times d_r$ dimensions respectively. $n_e, n_r$ are the number of entities and relations, and $d_e, d_r$ are the embedding dimension of entities and relations respectively.

***Pretrained Language Model KG Embedding*** **:** Similar to KGE, the word, sentence, and document representations of triples can be represented in vector forms by using pretrained (word embedding and language) models. Therefore, using the pretrained language models, we can represent the embedding of $\mathcal{TK}$ as the following set $\mathcal{TKE} = \{(\mathbf{h}^T, \mathbf{r}, \mathbf{t}^T) | \mathbf{h}^T, \mathbf{t}^T \in \mathbf{E}^T, \mathbf{r} \in \mathbf{R}\} \subset \mathbf{E}^T \times \mathbf{R} \times \mathbf{E}^T\}$ where $\mathbf{E}^T$ are the embedding set of all word/sentence/document level of all entities in the KG which are obtained by feeding the word/sentence/document of each entity into a pretrained language or word embedding models.

***Quaternion and Dihedron Algebra*** **:** To integrate graph, word, sentence and document representation of an entity or a relation in a unified representation, we require a 4D algebra which has been well-studied in the area of hypercomplex numbers. In particular, we focus on Quaternion $Q$ and Dihedron $D$ algebra [19] as 4D hypercomplex numbers defined as follows $u = s + x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k}$ where $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$ are the three imaginary parts. In Quaternion representation, we have

$$\begin{aligned} &\boldsymbol{i}^2 = \boldsymbol{j}^2 = \boldsymbol{k}^2 = \boldsymbol{ijk} = \bar{1},\ \boldsymbol{ij} = \boldsymbol{k},\ \boldsymbol{jk} = \boldsymbol{i},\ \boldsymbol{ki} = \boldsymbol{j}, \\ &\boldsymbol{ji} = \bar{\boldsymbol{k}},\ \boldsymbol{kj} = \bar{\boldsymbol{i}},\ \boldsymbol{ik} = \bar{\boldsymbol{j}}. \end{aligned} \tag{2}$$

where $\bar{a} = -a$. In Dihedron representation, we have

$$\begin{aligned} &\boldsymbol{i}^{\mathbf{2}} = \bar{\mathbf{1}},\, \boldsymbol{j}^{\mathbf{2}} = \boldsymbol{k}^{\mathbf{2}} = \mathbf{1},\, \boldsymbol{ij} = \boldsymbol{k},\, \boldsymbol{jk} = \bar{\boldsymbol{i}},\, \boldsymbol{ki} = \boldsymbol{j}, \\ &\boldsymbol{ji} = \bar{\boldsymbol{k}},\, \boldsymbol{kj} = \boldsymbol{i},\, \boldsymbol{ik} = \bar{\boldsymbol{j}}. \end{aligned} \tag{3}$$

Each Quaternion and Dihedron have their own product which are defined as follows:

$Quaternion\ \ Product$ : This product is also known as the Hamilton product. Lets $u = s_u + x_u\boldsymbol{i} + y_u\boldsymbol{j} + z_u\boldsymbol{k}, v = s_v + x_v\boldsymbol{i} + y_v\boldsymbol{j} + z_v\boldsymbol{k}$ be two Quaternion numbers. The Hamilton product between $u, v$ is defined as follows:

$$\begin{aligned} u \otimes_Q v := &\left(s_u s_v - x_u x_v - y_u y_v - z_u z_v\right) \\ &+ \left(s_u x_v + x_u s_v + y_u z_v - z_u y_v\right)\mathbf{i} \\ &+ \left(s_u y_v - x_u z_v + y_u s_v + z_u x_v\right)\mathbf{j} \\ &+ \left(s_u z_v + x_u y_v - y_u x_v + z_u s_v\right)\mathbf{k}. \end{aligned} \tag{4}$$

$Dihedron\ \ Product$ : Let $u, v$ be two Dihedron numbers. The Dihedron product is defined as

$$\begin{aligned} u \otimes_D v := &\left(s_u s_v - x_u x_v + y_u y_v + z_u z_v\right) \\ &+ \left(s_u x_v + x_u s_v - y_u z_v + z_u y_v\right)\mathbf{i} \\ &+ \left(s_u y_v - x_u z_v + y_u s_v + z_u x_v\right)\mathbf{j} \\ &+ \left(s_u z_v + x_u y_v - y_u x_v + z_u s_v\right)\mathbf{k}. \end{aligned} \tag{5}$$

$Inner\ \ Product$ : In both Dihedron and Quaternion, the inner product is defined as

$$\langle u, v\rangle = u \cdot v := s_u s_v + x_u x_v + y_u y_v + z_u z_v. \tag{6}$$

$Conjugate$ : The conjugate in both representations is $\bar{u} = s - x\boldsymbol{i} - y\boldsymbol{j} - z\boldsymbol{k}$.
$Norm$ : The norm in Quaternion and Dihedron representations is defined as $\|u\| = \langle u, \bar{u}\rangle$ is $\sqrt{s_u^2 + x_u^2 + y_u^2 + z_u^2}$ and $\sqrt{s_u^2 + x_u^2 - y_u^2 - z_u^2}$ respectively.
Both Quaternion and Dihedron provide various geometric representations. While Quaternion numbers represent hyperspheres, Dihedron numbers represent various shapes namely circle, one-sheet, two-sheet and Conical Surface. Therefore, Dihedron is more expressive in terms of geometric representation.

## 4 Our Method

In this section, we present a class of embedding models based on Dihedron and Quaternion algebra. Our models act on 4D space which enables modeling the four representations of entities, from node level in graph to word, sentence, and document levels in text. Incorporating embedding from word, sentence, and document levels enables modeling various features from unstructured data to capture both content and contextualized information. Those features, together with embedding from KG provide a rich representation for each entity in the KG. Additionally, our formulation enables capturing the pairwise correlation between different elements mutually to provide a rich feature representation (by a mixture of various features) while measuring the likelihood of links. Figure 2 illustrates our overall framework. To measure the likelihood of a link (edge) e.g., (Danny_Pena, wasBornIn, Inglewood,_California), we split the triple into two parts of triple pattern e.g., (Danny_Pena, wasBornIn,?) and tail e.g., Inglewood,_California. Because the triple pattern corresponds to a question e.g., Where was Danny Pena borne?, we also call it a query. We then compute query and answer (tail) embedding considering textual and structural information and then calculate the final score by measuring the distance between the query and the corresponding answer in the embedding space.

In the rest of this section, we will present our model according to the following steps a) entity representation, b) relation and query representation, c) triple plausibility measurement (answering query), and d) dimension adjustment.

| Name | query $(h, r, ?)$ | embedding |
|---|---|---|
| Tetra | $\mathbf{q} = \mathbf{h} \otimes \mathbf{r}^{\triangleleft}$, | $\mathbf{h}, \mathbf{r} \in \mathcal{D}^d \ \ or \ \ \mathcal{Q}^d, \mathbf{h} = \mathbf{s}_h + \mathbf{x}_h^{\mathcal{T}_1}\boldsymbol{i} + \mathbf{y}_h^{\mathcal{T}_2}\boldsymbol{j} + \mathbf{z}_h^{\mathcal{T}_3}\boldsymbol{k}, \mathbf{r}^{\triangleleft} = \frac{\mathbf{r}}{\lvert\mathbf{r}\rvert}$ |
| Robin | $\mathbf{q} = \bar{\mathbf{h}} \otimes \boldsymbol{h}_{lt}^{\triangleleft^{\mathcal{T}_1}} + \boldsymbol{h}_l^{\mathcal{T}_2} + \boldsymbol{h}_t^{\mathcal{T}_3} + \boldsymbol{r}$, | $\boldsymbol{h}_{lt} = \mathbf{s}_h^{l_1} + \mathbf{x}_h^{l_2}\boldsymbol{i} + \mathbf{y}_h^{t_1}\boldsymbol{j} + \mathbf{z}_h^{t_2}\boldsymbol{k}, \boldsymbol{h}_{lt}^{\triangleleft} = \frac{\boldsymbol{h}_{lt}}{\lvert\boldsymbol{h}_{lt}\rvert}$ |
| Lion | $\mathbf{q} = \bar{\mathbf{h}} \otimes \boldsymbol{h}_{t_1t_2}^{\triangleleft^{\mathcal{T}_1\mathcal{T}_2}} + \boldsymbol{h}_{t_1}^{\mathcal{T}_1} + \boldsymbol{h}_{t_2}^{\mathcal{T}_2} + \boldsymbol{r}$, | $\boldsymbol{h}_{lt} = \mathbf{s}_h^{t_1^1} + \mathbf{x}_h^{t_2^1}\boldsymbol{i} + \mathbf{y}_h^{t_1^2}\boldsymbol{j} + \mathbf{z}_h^{t_2^2}\boldsymbol{k}, \boldsymbol{h}_{t_1t_2}^{\triangleleft} = \frac{\boldsymbol{h}_{t_1t_2}}{\lvert\boldsymbol{h}_{t_1t_2}\rvert}$ |

**Table 1:** Query representation of our models.

### 4.1 Entity Representation

We represent each entity $e$ as $4 * d_e$ dimensional vector as follows

$$\mathbf{e} = \mathbf{s}_e + \mathbf{x}_e^{\mathcal{T}_1}\boldsymbol{i} + \mathbf{y}_e^{\mathcal{T}_2}\boldsymbol{j} + \mathbf{z}_e^{\mathcal{T}_3}\boldsymbol{k}, \tag{7}$$

where $^{\mathcal{T}_i}, i = 1, 2, 3$ can be selected from the model set

$\mathcal{M} =$
$\{$_, Word2Vec, Fasttext, Sentence Transformer, Doc2Vec$\}$.

where "_" refers to trainable embedding from KG structure.

Each of the models in $\mathcal{M}$ are pretrained and we feed the corresponding text into the models to obtain the pretrained embedding vector $\mathbf{x}_e^{\mathcal{T}_1}$. Note that the dimension of the vectors $\mathbf{s}_e, \mathbf{x}_e^{\mathcal{T}_1}, \mathbf{y}_e^{\mathcal{T}_2}, \mathbf{z}_e^{\mathcal{T}_3}$ should be same. In the case of mismatched dimensions, we use a multi-layer perceptron to adjust the dimension before integrating it into the entity vector $\mathbf{e}$. In Figure 2, both of the entities "Danny_Pena" and "Inglewood,_California" contain textual descriptions. We process the textual description of the entities to extract word, sentence and document level representation. Using the models in $\mathcal{M}$, we obtain the 4D representation of the entities.

$Relation \ and \ Query \ Representation$ : Each relation is represented as a rotation in a hypercomplex space which is

$$\mathbf{r} = \frac{\mathbf{s}_r + \mathbf{x}_r^{\mathcal{T}_1}\boldsymbol{i} + \mathbf{y}_r^{\mathcal{T}_2}\boldsymbol{j} + \mathbf{z}_r^{\mathcal{T}_3}\boldsymbol{k}}{\sqrt{\mathbf{s}_r^2 + \mathbf{x}_r^{\mathcal{T}_1^2} + \mathbf{y}_r^{\mathcal{T}_2^2} + \mathbf{z}_r^{\mathcal{T}_3^2}}}. \tag{8}$$

To measure the likelihood of a link $(h, r, t)$, we split it into two pieces of query $(h, r, ?)$ and answer $t$. Each of these two elements are embedded into 4D spaces. Table 1 presents three variants of our models for query representation of a triple pattern in the form of $(h, r, ?)$. We propose the following approaches for query representation:

$Text-Enhanced \ RelaTional \ RotAtion \ (Tetra)$ : This model represents each entity by four parts, namely embedding learned from KG structure $\mathbf{s}_h$, and the parts $(\mathbf{x}_{\mathbf{h}}^{\mathcal{T}_2}, \mathbf{y}_h^{\mathcal{T}_3}, \mathbf{z}_h^{\mathcal{T}_1})$ which are learned from textual description (word, sentence and document) of the entities by three of the models in $\mathcal{M}$. The model computes the query by providing relation-specific rotating of the head in Dihedron or Quaternion spaces. $\otimes$ refers to Hamilton or Dihedron products for Quaternion and Dihedron spaces. According to equation 5 and 4, the obtained query (the first row of the Table 1) inherently contains the pairwise correlations between each of $\mathbf{s}_h, \mathbf{x}_{\mathbf{h}}^{\mathcal{T}_1}, \mathbf{y}_h^{\mathcal{T}_1}, \mathbf{z}_h^{\mathcal{T}_1}$ and provides a rich feature representation for the corresponding query (mixture of sub-features). Figure 2 shows the representation of this model.

$Text-RelatiOn-Based \ RotatIon \ and \ TranslatioN (Robin)$ : In this model, we incorporate the embedding of the entity name (e.g., "Berlin") indexed by $l$ and the embedding of the textual description indexed by $t$. We perform rotation $\boldsymbol{h}_{lt}^{\triangleleft^{\mathcal{T}_1}}$ and translation $\boldsymbol{h}_l^{\mathcal{T}_2} + \boldsymbol{h}_t^{\mathcal{T}_3} + \boldsymbol{r}$ by these two source of information. We use two different word embedding and language models from $\mathcal{M}$ (details in experiment section) to embed name and textual description. Therefore, the query is computed by head entity embedding from KG, relation embedding, name, and description of the entity by considering translation and rotation in Quaternion or Dihedron spaces. Combining translation and rotation improves the expressivity power of the model to learn more complex structures than a single rotation [16, 20].

$Language\ model - RelatIon\ ROtation\ and\ TranslatioN\ (Lion)$ : In this model, for the same textual description of an entity, we use two language/word embedding models (indexed by $l$) to embed the same description. We use description embedding as translation and rotation for query representation in Dihedron or Quaternion spaces.

The above mentioned query representations are related to the triple pattern $(h, r, ?)$. For the triple pattern $(?, r, t)$, we use $(t, r^{-1}, ?)$ for query representation, and utilize the equations in the Table 1 for query computation. $r^{-1}$ is a reverse relation corresponding to the relation $r$, which are added to the training set [21, 22].

Note that each of the introduced models in the Table 1 has its advantage and can be used depending on various characteristics of the KG such as sparsity and density, quality of textual description, etc. For example, if the KG has a complex structure, a more expressive model is required. Therefore, the model Robin and Lion are used due to having a more degree of expressivity obtained by mixing the translation and rotation, compared to a single rotation Tetra.

### 4.2 Triple Plausibility Measurement

To measure the plausibility of a triple $(h, r, t)$, the distance between the query $\mathbf{q}$ and the corresponding answer $\boldsymbol{t}$ is computed as follows

$$f(h, r, t) = -d(\boldsymbol{q}, \boldsymbol{t}) + b_h + b_t, \tag{9}$$

where $b_h, b_t$ are entity-specific biases proposed in [23]. In Figure 2, we see several geometric representations (on top of the figure) in which the query and the corresponding answer are matched (from the blue circle to the red circle). In Figure 2, in the description of "Danny_Pena", we observe the mention of the entity "Inglewood,_California" as well as "born" which is closely related to the "wasBornIn" relation. In the description of "Inglewood,_California", we see the entity mention "Inglewood" as well as "California". The descriptions are highly correlated and cover the mention of the triple elements (head, relation, tail). Therefore, from the descriptions of the entities, it can be inferred that "Danny_Pena" was born in "Inglewood California".

### 4.3 Dimension Adjustment

The assumption of equation 7 is that the vectors $\mathbf{s}_e, \mathbf{x}_e^{\mathcal{T}_1}, \mathbf{y}_e^{\mathcal{T}_2}, \mathbf{z}_e^{\mathcal{T}_3}$ have same dimension. However, by using pretrained language models, the mentioned vectors may have different dimensions. To overcome this problem, we employ a neural network to adjust the dimension. Therefore, equation 7 is written as follows

$$\mathbf{e} = \mathbf{s}_e + NN(\mathbf{x}_e^{\mathcal{T}_1})\boldsymbol{i} + NN(\mathbf{y}_e^{\mathcal{T}_2})\boldsymbol{j} + NN(\mathbf{z}_e^{\mathcal{T}_3})\boldsymbol{k}, \tag{10}$$

where $NN$ is a multilayer perceptron which takes a vector with dimension $d_{\mathcal{T}_i}$ and returns a vector with dimension $d_e$.

## 5 Experiments

This section presents an evaluation of our models and other KGE models. The evaluation and various ablation studies have been done in several text based KGs introduced in the following part.

### 5.1 Experimental Setup

$Datasets$ : We consider four different datasets in our experiment: NATIONS, Diabetes, FB-ILT and YAGO-10. **NATIONS** [24] is a small dataset that represents relations between countries. The entity names are given and we collect entity descriptions from Wikipedia data. We construct **Diabetes** from the original data source [25] that contains all types of disease information . We extracted names and descriptions for each entity. **FB-ILT** is a subset of Freebase dataset FB15k [12]. We collected triples and entity descriptions from [26]. The entity names were collected from [3]. **YAGO-10** is a subset

[3] https://git.uwaterloo.ca/jimmylin/BuboQA-data/raw/master/FB5M.name.txt.bz2

of YAGO dataset [27]. We collected triples and entity names from [4] and entity descriptions from datasets/YAGO-10 plus/Textual data.txt from the implementation [5].

Table 2 summarizes the overall number of entities, relations, training, validation, and test triples in these datasets.

| Dataset | #ent | #rel | #train | #val | #test |
|---|---|---|---|---|---|
| NATIONS | 14 | 55 | 1,592 | 199 | 201 |
| Diabetes | 7,886 | 67 | 56,830 | 7,103 | 7,103 |
| FB-ILT | 11,757 | 1231 | 285,850 | 29,580 | 34,863 |
| YAGO-10 | 103,222 | 30 | 155,370 | 2,295 | 2,292 |

**Table 2:** The statistics of our datasets.

$Environment\ and\ Hyperparameters$ : Our models are implemented with PyTorch [28] on the top of the Chami's framework, [29]. For pretrained language models, we consider *Word2Vec* [30] and *Doc2Vec* [31] in python Gensim library, *FastText* [6], and *Sentence Transformer* with pretrained model "distilbert-base-nli-mean-tokens" [7].

We use Adagrad optimizer to train the model and early-stopping on the validation dataset to prevent overfitting.

The hyperparameters we use for both low($d = 32$) and high dimension($d = 500$) setup are shown in the Table 3, where $b$ is batch size, $l_r$ is the learning rate and $n$ is the negative sampling size. Specifically, $n = -1$ indicates that we use full cross-entropy loss instead of negative sampling as [16].

$Evaluation\ Metrics$ : Following previous works, we evaluate our models on the link prediction task with the following metrics: Mean Reciprocal Rank(MRR), Hits@K, K = {1, 3, 10}. Specifically, given all queries in the dataset, MRR measures the mean reciprocal ranks of correct entities, and Hits@K measures the proportion of correct entities ranking at top-K. During the models' evaluation, filtering setup [32] is performed to remove the existing triples in the dataset during ranking.

$Baselines\ and\ Our\ Ablation\ Models$ : For comparison, we consider three baselines and state of the art KGE models without considering the textual information: TransE [12], AttE and AttH [16]. We also consider three baselines and state of the art KGE models considering the textual information: DKRL [5], ConMask [6] and PretrainKGE [9].

We also consider the ablation study of different variants of our model. For simplicity, we abbreviate the pretrained language models {**W**ord2Vec, **F**asttext, **D**oc2Vec, **S**entence Transformer, } as $\{W, F, D, S\}$. We compare 4 different variants of Robin that use one of $\{W, F, D, S\}$ individually. For Lion, we compare 2 variants that utilize two pretrained language or word embedding models $\{S, D\}$ or $\{F, S\}$. For all variants of Tetra, we always use the same pretrained language or word embedding models from $\{W, S, D\}$ to embed the three pieces of information namely word, sentence, and document. Specifically, Tetra is the model that we only incorporate $\{W, S, D\}$ to represent different levels of textual information only for the head entity, and Tetra-tail is the model that we do so for both head and tail entities.

### 5.2 Link Prediction results and analysis

$Low\ Dimensional\ Results\ On\ Large\ Scale\ Datasets$ : Among four different datasets, the size of Diabetes, FB-ILT and YAGO-10 are significantly larger than NATIONS. We first compare different variants of our model (Robin, Lion and Tetra) with baselines on these three datasets.

Table 4 presents the results with the embedding dimension $d = 32$. On three datasets, different variants of our model consistently outperform all competitors, which proves that our model is able to enhance the quality of KGE. Among different variants of our model, we find that Lion and Tetra

[4] https://github.com/TimDettmers/ConvE

[5] https://github.com/pouyapez/mkbe

[6] https://dl.fbaipublicfiles.com/fasttext/vectors-wiki/wiki.simple.zip

[7] https://huggingface.co/sentence-transformers

| Dataset | k | b | $l_r$ | n |
|---|---|---|---|---|
| Diabetes | 32 | 100 | 0.25 | -1 |
| | 500 | 100 | 0.01 | -1 |
| FB-ILT | 32 | 100 | 0.25 | -1 |
| | 500 | 100 | 0.01 | -1 |
| NATIONS | 32 | 400 | 0.01 | 100 |
| | 500 | 400 | 0.01 | 10 |
| YAGO-10 | 32 | 400 | 0.2 | 100 |
| | 500 | 128 | 0.01 | -1 |

**Table 3:** Optimal hyperparameters for all datasets

| **Elements** | **Model** | **Diabetes** | | | | **FB-ILT** | | | | **YAGO-10** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | MRR | H@1 | H@3 | H@10 | MRR | H@1 | H@3 | H@10 | MRR | H@1 | H@3 | H@10 |
| Triples | TransE | 0.166 | 0.089 | 0.182 | 0.322 | 0.630 | 0.55 | 0.674 | 0.776 | 0.368 | 0.284 | 0.403 | 0.534 |
| | AttE | 0.149 | 0.076 | 0.163 | 0.302 | 0.616 | 0.532 | 0.665 | 0.773 | 0.364 | 0.289 | 0.394 | 0.518 |
| | AttH | 0.153 | 0.080 | 0.166 | 0.304 | 0.619 | 0.536 | 0.666 | 0.777 | 0.380 | 0.300 | 0.415 | 0.538 |
| Triples + Text | DKRL | 0.158 | 0.085 | 0.171 | 0.310 | 0.614 | 0.535 | 0.657 | 0.763 | 0.339 | 0.255 | 0.373 | 0.509 |
| | ConMask | 0.170 | 0.092 | 0.188 | 0.326 | 0.629 | 0.545 | 0.681 | 0.783 | 0.362 | 0.294 | 0.389 | 0.504 |
| | Pretrain-KGE | 0.151 | 0.078 | 0.164 | 0.300 | 0.609 | 0.529 | 0.652 | 0.759 | 0.349 | 0.274 | 0.380 | 0.502 |
| Triples + Name + Text | Robin_W | 0.173 | 0.096 | 0.188 | 0.333 | 0.631 | 0.550 | 0.679 | 0.780 | 0.363 | 0.281 | 0.396 | 0.528 |
| | Robin_F | 0.173 | 0.095 | 0.186 | 0.338 | **0.635** | 0.554 | **0.684** | **0.784** | 0.365 | 0.285 | 0.397 | 0.524 |
| | Robin_S | 0.173 | 0.095 | 0.188 | 0.333 | 0.628 | 0.546 | 0.676 | 0.778 | 0.363 | 0.282 | 0.398 | 0.528 |
| | Robin_D | 0.173 | **0.097** | 0.187 | 0.333 | 0.632 | 0.552 | 0.679 | 0.780 | 0.366 | 0.286 | 0.401 | 0.528 |
| Triples + Text | Lion_SD | 0.167 | 0.090 | 0.180 | 0.330 | **0.635** | **0.556** | 0.682 | 0.783 | **0.397** | **0.314** | **0.441** | **0.554** |
| | Lion_FS | **0.175** | **0.097** | **0.190** | **0.340** | 0.633 | 0.552 | 0.681 | 0.781 | 0.395 | **0.314** | 0.439 | 0.548 |
| | Tetra | 0.140 | 0.068 | 0.150 | 0.288 | 0.492 | 0.401 | 0.533 | 0.671 | 0.169 | 0.089 | 0.178 | 0.343 |
| | Tetra-tail | 0.151 | 0.080 | 0.162 | 0.295 | 0.612 | 0.527 | 0.662 | 0.768 | 0.331 | 0.252 | 0.362 | 0.491 |

**Table 4:** Link prediction Evaluation of multimodal models on Diabetes, FB-ILT, and YAGO-10 datasets for d = 32

outperform Robin. This result indicates that better performance can be obtained by incorporating different pretrained language models of a single modality. Please remember that Robin, Lion and Tetra separately incorporates one, two, and three pretrained language models. Furthermore, the performance of Tetra-tail is much better than Tetra because Tetra-tail considers pretrained model for both head and tail entities, but Tetra only does so for the head entity. Regarding the transformation function and geometric point of view, Lion and Robin use translation and rotation simultaneously, while Tetra only uses rotation. Therefore, on large dataset with complex patterns and structures together with a low embedding dimension, Lion and Robin outperform Tetra.

| **Elements** | **Model** | **Diabetes** | | | | **FB-ILT** | | | | **YAGO-10** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | MRR | H@1 | H@3 | H@10 | MRR | H@1 | H@3 | H@10 | MRR | H@1 | H@3 | H@10 |
| Triples | TransE | 0.179 | 0.098 | 0.197 | 0.342 | 0.739 | 0.675 | 0.777 | 0.852 | 0.421 | 0.351 | 0.461 | 0.556 |
| | AttE | 0.176 | 0.097 | 0.190 | 0.341 | 0.674 | 0.604 | 0.711 | 0.805 | 0.356 | 0.294 | 0.389 | 0.471 |
| | AttH | 0.115 | 0.054 | 0.118 | 0.250 | 0.610 | 0.521 | 0.658 | 0.779 | 0.313 | 0.256 | 0.336 | 0.431 |
| Triples + Text | DKRL | 0.162 | 0.083 | 0.176 | 0.328 | 0.718 | 0.631 | 0.779 | 0.868 | 0.333 | 0.239 | 0.371 | 0.520 |
| | ConMask | 0.177 | 0.094 | 0.194 | 0.349 | 0.698 | 0.620 | 0.747 | 0.842 | 0.381 | 0.306 | 0.421 | 0.519 |
| | Pretrain-KGE | 0.159 | 0.082 | 0.172 | 0.323 | 0.739 | 0.673 | 0.780 | 0.857 | 0.320 | 0.231 | 0.353 | 0.495 |
| Triples + Name + Text | Robin_W | 0.180 | 0.098 | 0.196 | 0.351 | 0.767 | 0.712 | 0.805 | 0.864 | 0.402 | 0.327 | 0.442 | 0.541 |
| | Robin_F | 0.179 | 0.095 | 0.198 | 0.353 | 0.769 | 0.713 | 0.806 | 0.865 | 0.402 | 0.327 | 0.442 | 0.541 |
| | Robin_S | 0.182 | 0.097 | 0.200 | 0.355 | 0.771 | 0.713 | 0.809 | 0.873 | 0.436 | 0.365 | 0.471 | 0.571 |
| | Robin_D | 0.181 | 0.099 | 0.198 | 0.355 | 0.766 | 0.712 | 0.803 | 0.862 | 0.35 | 0.272 | 0.395 | 0.500 |
| Triples + Text | Lion_SD | 0.192 | 0.107 | 0.208 | **0.370** | 0.777 | 0.722 | 0.815 | 0.874 | 0.433 | 0.363 | 0.471 | 0.562 |
| | Lion_FS | **0.193** | **0.108** | **0.212** | 0.369 | **0.779** | 0.723 | **0.816** | **0.878** | **0.440** | **0.367** | **0.478** | **0.577** |
| | Tetra | 0.168 | 0.089 | 0.183 | 0.335 | 0.722 | 0.652 | 0.770 | 0.844 | 0.335 | 0.256 | 0.374 | 0.492 |
| | Tetra-tail | 0.160 | 0.088 | 0.173 | 0.307 | 0.776 | **0.731** | 0.800 | 0.857 | 0.415 | 0.352 | 0.446 | 0.535 |

**Table 5:** Link prediction Evaluation of multimodal models on Diabetes, FB-ILT, and YAGO-10 datasets for d = 500

$High\ Dimensional\ Results\ On\ Large\ Scale\ Datasets$ : Next we perform evaluation on the same datasets where embedding dimension $d = 500$. The results can be viewed in Table 5.

In the setting of high dimension, our model still outperforms all competitors. Among different variants of our model, Lion is still the best one, because it considers both translation and rotation compared with Tetra and it incorporates more pretrained language models than Robin. When compared with low-dimension cases, all of our ablation models have a larger increment of performance. The result is that the default dimensions of most pretrained language models are much larger than $d = 32$. When we incorporate them with KGE models in the low dimension setting, a significant dimension reduction must be applied to representations from these pretrained language models, and thus lots of semantic information is dropped. Additionally, in both low and high dimension, only on FB-ILT does the Quaternion performs better than Dihedron. This is consistent with the findings in [16**?** ] that rotation with more degree of hyperbolicity is more suitable for Freebase dataset. In other words, spherical rotation is more suitable for modeling the structures and patterns in Freebase datasets.

| | NATION (d=32) | | | | NATION (d=500) | | | |
|---|---|---|---|---|---|---|---|---|
| **Model** | MRR | H@1 | H@3 | H@10 | MRR | H@1 | H@3 | H@10 |
| TransE | 0.684 | 0.542 | 0.779 | 0.99 | 0.712 | 0.590 | 0.789 | 0.990 |
| RotE | 0.648 | 0.488 | 0.741 | 0.980 | 0.795 | 0.699 | 0.858 | 0.993 |
| RotH | 0.728 | 0.610 | 0.804 | 0.990 | 0.789 | 0.684 | 0.861 | 0.995 |
| DKRL | 0.660 | 0.505 | 0.774 | 0.998 | 0.706 | 0.582 | 0.786 | 0.990 |
| ConMask | 0.662 | 0.505 | 0.761 | 0.988 | 0.713 | 0.587 | 0.808 | 0.993 |
| Pretrain-KGE | 0.674 | 0.540 | 0.756 | 0.985 | 0.718 | 0.592 | 0.803 | 0.993 |
| Robin_W | 0.730 | 0.610 | 0.801 | 0.990 | 0.731 | 0.614 | 0.796 | 0.993 |
| Robin_F | 0.732 | 0.609 | 0.811 | 0.993 | 0.721 | 0.597 | 0.791 | 0.995 |
| Robin_S | 0.732 | 0.612 | 0.799 | 0.980 | 0.730 | 0.614 | 0.786 | 0.993 |
| Robin_D | 0.728 | 0.610 | 0.789 | 0.993 | 0.729 | 0.612 | 0.801 | 0.995 |
| Lion_SD | 0.736 | 0.624 | 0.801 | 0.988 | 0.726 | 0.605 | 0.801 | 0.993 |
| Lion_FS | 0.727 | 0.605 | 0.801 | 0.993 | 0.725 | 0.602 | 0.801 | 0.993 |
| Tetra | **0.786** | **0.669** | **0.876** | **0.995** | **0.826** | 0.729 | **0.910** | **1.0** |
| Tetra-tail | 0.775 | 0.667 | 0.848 | 0.993 | 0.822 | **0.731** | 0.893 | 0.993 |

**Table 6:** Link prediction evalua on NATIONS

*Results On the Small Scale Dataset* : We also evaluate our model on a small-scale dataset NATIONS. The results of baselines and different variants of our model can be viewed in Table 6, where both cases of embedding dimension $d = \{32, 500\}$ are considered.

Our model still outperforms all baselines on NATIONS. Unlike the large datasets above, on NATIONS Tetra outperforms Lion by a large margin. The reason is that on such a small dataset, structural information from KG is insufficient and the exploitation of additional textual information becomes more important. Given insufficient structural information, Lion cannot learn the translation well. However, Tetra performs better because it incorporates three different pretrained models and thus it exploits the textual information better.

*Summary Of Link Prediction Experiments* : We make a summary about all the link prediction experiments above.

1. On all datasets, with the setting of both low and high dimensions, our proposed model always outperforms other models. Specifically, our model performs better with the setting of high dimension.
2. Lion and Tetra generally outperform Robin, which indicates incorporating more pretrained language models enhances the performance.
3. When dataset is large, Lion outperforms Tetra because Lion learns extra translation compared with Tetra. However, when a dataset is small, Tetra performs better because it incorporates more pretrained language models and better utilizes textual information when there is a lack of proper structural information.

*Effect Of Textual Information In Entity Representation* : In this visualization, we demonstrate the effect of pretrained language models with Tetra-tail. We select two triples from the FB-ILT dataset: *(40th Academy Awards, locations, Los Angeles California USA)* and *(Atlantic Canada, contained by, Canada)*. The heat maps in Figure 3 show the cosine similarities between four parts of our Dihedron representation $s_h, x_h^{\mathcal{T}_1}, y_h^{\mathcal{T}_2}, z_h^{\mathcal{T}_3}$: *entity embedding*, *word embedding*, *sentence embedding*, *document embedding*. According to four similarities in the diagonal, we understand the contribution of each part when matching the query and tail in the link prediction task, where Y-axis of the heating map is for the query and X-axis is for the tail.

Comparing the similarity of entity embedding $s_h$ and textual embedding $\{x_h^{\mathcal{T}_1}, y_h^{\mathcal{T}_2}, z_h^{\mathcal{T}_3}\}$, we find out textual embedding plays an important role when matching the information from query and tail. On the one hand, in the first heat map, the head entity *40th Academy Awards* is semantically different than the tail entity *Los Angeles California USA*, and textual information helps a lot in bridging the gap between query and tail. We can view that the cosine similarity between sentence embedding (0.63) is much larger than the similarity between entity embedding (0.44). On the other hand, in the second heat map, the head entity *Atlantic Canada* is semantically similar to the tail entity *Canada*,

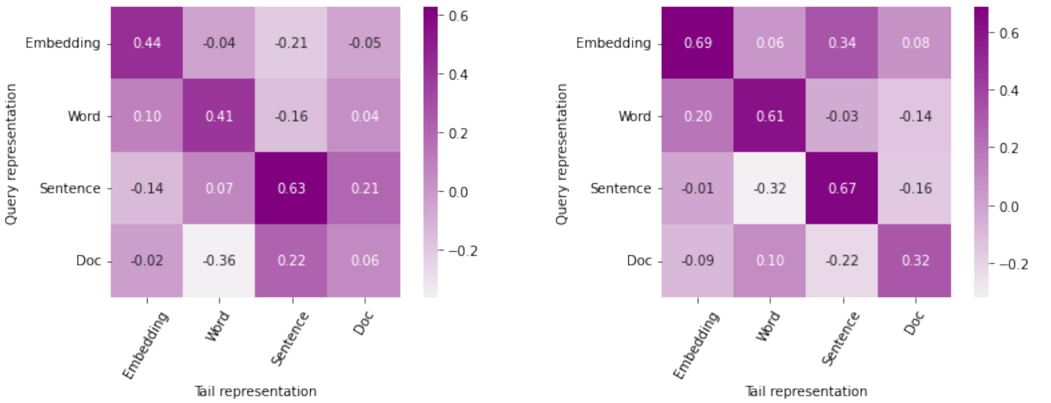


**Figure 3:** Cosine similarities for queries and answer of two triples. Up: *(40th Academy Awards, locations, Los Angeles California USA)*. Down: *(Atlantic Canada, contained by, Canada)*

and thus entity embedding naturally has the highest similarity (0.69). Moreover, matching between textual embeddings such as word and sentence embedding is also effective in this case.

In both examples, among three levels of textual representations $\{x_h^{\mathcal{T}_1}, y_h^{\mathcal{T}_2}, z_h^{\mathcal{T}_3}\}$, we observe that sentence embedding $y_h^{\mathcal{T}_2}$ has the largest similarity scores. However, the document embedding $z_h^{\mathcal{T}_3}$ shows very low similarity scores to the other three. Therefore we conclude that sentence embedding captures most semantic information from the entity description during the matching between query and tail. The other two levels of textual representations generally capture relatively less information for matching the query and tail. Specifically, document embedding has the least effect, because it is too coarse to condense all information from the description into a single embedding in the pretrained language model (Doc2Vec). However, from the second heat map, we observe that document embedding can partially help to improve the link prediction in some triples.

### 5.3 Semantic Clustering Of Entity Embedding

We further demonstrate the semantic of learned entity embedding from our best performing model Lion_SD on FB-ILT dataset. The target is to view whether semantically similar entities also have close distance in the space of trained entity embedding. Specifically, we perform K-means clustering algorithm and TSNE dimension reduction to these trained entity embedding so that they can be visualized on 2-dimension figure. The result can be viewed in the Figure 4.

From the figure, we can see that semantically relevant points are close to each other. For example, universities and languages separately gather in the blue and red clusters, respectively. We can conclude that the KGE learned by our model possesses semantic information.

| Triple | Sentence rank | Sentence source | Sentence |
|---|---|---|---|
| Mars_Callahan,<br>created,<br>Zigs_(film) | 1 | tail | **Zigs** is a 2001 English language drama starring Jason Priestley Peter Dobson and Richard Portnow and **directed by Mars Callahan**. |
| | 2 | tail | The film received an r rating by the MPAA. |
| | 3 | head | At the age of eleven Callahan toured with a children’s musical group through thirty-seven states. |
| Margaret_of_Geneva,<br>isMarriedTo,<br>Thomas_Count_of_Savoy | 1 | tail | **Thomas Tommaso I was count of savoy** from 1189 to 1233 he is sometimes numbered Thomas I to distinguish him from his son of the same name who governed savoy but was not count. |
| | 2 | head | **Margaret of Geneva** 1180-1252 countess of savoy was the daughter of William I count of Geneva. |
| | 3 | head | When her father was **escorting** her to France in May 1195 Thomas I of savoy **carried her off**. |
| Gilbert_Mushangazhike,<br>isAffiliatedTo,<br>Mpumalanga_Black_Aces_F.C. | 1 | tail | **Aces** usually played their home games in the **Mpumalanga** province but were trained in Johannesburg. |
| | 2 | head | **Gilbert Mushangazhike** born 11 August 1975 in Harare is a Zimbabwean footballer. |
| | 3 | head | The association football striker recently **played** in south Africa for manning rangers F.C. Orlando Pirates and **Mpumalanga Black Aces** . |

**Table 7:** Examples about sentence contribution of our model, where a sentence with higher rank (smaller number) is more important for model. Sentence source indicates whether a sentence comes from the description of head or tail entity.

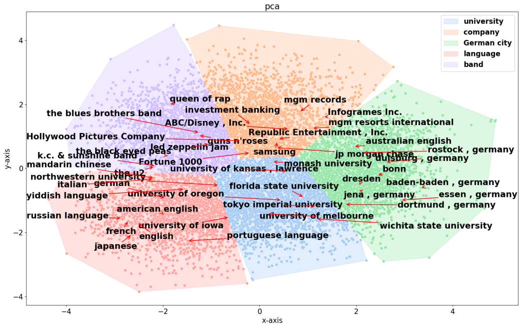


**Figure 4:** Trained entity embedding clustering of Lion_SD model of FB-ILT dataset where dimension $d = 32$. In the figure, each cluster has a topic, each point represents an entity and each red arrow points to the corresponding point of an entity name.

### 5.4 Contribution of individual sentences

We consider the Tetra-tail model and YAGO-10 dataset in this analysis. The target is to explore how our model exploits sentence-level representation by selecting important sentences. To achieve this, for each triple, we collect sentences from the descriptions of its head and tail entities. Then we measure the importance of each sentence with Shapley value [33] and rank all sentences according to their Shapley values from high to low. Then we manually check the semantics of each sentence. If a semantically important sentence obtains a high rank, we can conclude that our model exploits sentence-level representation for the corresponding triple.

In Table 7, we select 3 examples, and in each example, we only demonstrate the top-3 important sentences among a maximum of 8 sentences in the description. For the first triple (Mars_Callahan, created, Zigs_(film)), we can see that the top-1 sentence comes from the tail description, and the keywords about the head entity "directed by Mars Callahan" is within it. Our model selects this sentence because it is important when matching the query with the tail.

For the second triple (Margaret_of_Geneva, isMarriedTo, Thomas_Count_of_Savoy), we can find that the top-1 and top-2 sentences refer to the definition of both entities. The keywords "escorting" and "carried her off" in the top-3 sentence can be regarded as implicit information about the relation "isMarriedTo", even though they are not literary the same. In addition, "count of" in head and "countess of" in tail implicitly represents marriage relations. With the help of entity definition and information about the relation, our model grasps the overall picture of the triple.

The third triple (Gilbert_Mushangazhike, isAffiliatedTo, Mpumalanga_Black_Aces_F.C.), our model also exploits top-1 and top-2 sentences to represent the head and tail entities. Furthermore, in the third sentence from the head description, the keywords "Mpumalanga Black Aces" are directly related to the tail entity, and "played" can be regarded as a synonym of the relation.

From these examples, we demonstrate that our model exploits sentence-level representation when representing the triple.

## 6 Conclusion

In this work, we enhance KGE models with different levels of semantic information from entity descriptions and names by incorporating multiple pretrained language models. In order to incorporate different semantic information with entity embedding, we propose a unified framework with the help of hypercomplex numbers. We also discuss several ablation models under this framework. Extensive experimental results, ablation studies, and analysis demonstrate that our framework effectively utilizes different levels of semantic information as well as structural knowledge and outperforms other KGE models.

## Acknowledgement

The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Zihao Wang.

## References

[1] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 2021. 1

[2] Shivani Choudhary, Tarun Luthra, Ashima Mittal, and Rajat Singh. A survey of knowledge graph embedding and their applications. *arXiv preprint arXiv:2107.07842*, 2021.

[3] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2015. 1

[4] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29 (12):2724–2743, 2017. 2

[5] Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. Representation learning of knowledge graphs with entity descriptions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016. 2, 3, 8

[6] Baoxu Shi and Tim Weninger. Open-world knowledge graph completion. *arXiv preprint arXiv:1711.03438*, 2017. 2, 3, 8

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1423. URL `https://doi.org/10.18653/v1/n19-1423`. 2

[8] Liang Yao, Chengsheng Mao, and Yuan Luo. Kg-bert: Bert for knowledge graph completion. *arXiv preprint arXiv:1909.03193*, 2019. 2, 3

[9] Zhiyuan Zhang, Xiaoqian Liu, Yi Zhang, Qi Su, Xu Sun, and Bin He. Pretrain-KGE: Learning knowledge representation from pretrained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 259–266, Online, November 2020. Association for Computational Linguistics. 3, 8

[10] Mengyao Li, Bo Wang, and Jing Jiang. Siamese pre-trained transformer encoder for knowledge base completion. *Neural Processing Letters*, 53(6):4143–4158, 2021. 3

[11] Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194, 2021. 2, 3

[12] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26, 2013. 3, 7, 8

[13] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, pages 1112–1119. Citeseer, 2014. 3

[14] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*, pages 2071–2080, 2016. 3

[15] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 3

[16] Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. Low-dimensional hyperbolic knowledge graph embeddings. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6901–6914. Association for Computational Linguistics, 2020. 3, 6, 8, 10

[17] Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. Quaternion knowledge graph embeddings. In *NeurIPS*, 2019. 3

[18] Canran Xu and Ruijiang Li. Relation embedding with dihedral group in knowledge graph. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 263–272. Association for Computational Linguistics, 2019. 3

[19] Gabor Toth and Gâabor Tâoth. *Glimpses of algebra and geometry*, volume 1810. Springer, 2002. 4

[20] Mojtaba Nayyeri, Chengjin Xu, Sahar Vahdati, Nadezhda Vassilyeva, Emanuel Sallinger, Hamed Shariat Yazdi, and Jens Lehmann. Fantastic knowledge graph embeddings and how to find the right space for them. In *International Semantic Web Conference*, pages 438–455. Springer, 2020. 6

[21] Seyed Mehran Kazemi and David Poole. Simple embedding for link prediction in knowledge graphs. *Advances in neural information processing systems*, 31, 2018. 7

[22] Timothée Lacroix, Nicolas Usunier, and Guillaume Obozinski. Canonical tensor decomposition for knowledge base completion. In *International Conference on Machine Learning*, pages 2863–2872. PMLR, 2018. 7

[23] Ivana Balazevic, Carl Allen, and Timothy Hospedales. Multi-relational poincaré graph embeddings. *Advances in Neural Information Processing Systems*, 32, 2019. 7

[24] Stanley Kok and Pedro Domingos. Statistical predicate invention. In *Proceedings of the 24th international conference on Machine learning*, pages 433–440, 2007. 7

[25] Chaoyu Zhu, Zhihao Yang, Xiaoqiong Xia, Nan Li, Fan Zhong, and Lei Liu. Multimodal reasoning based on knowledge graph embedding for specific diseases. *Bioinformatics*, 38 (8):2235–2245, 02 2022. ISSN 1367-4803. doi: 10.1093/bioinformatics/btac085. URL https://doi.org/10.1093/bioinformatics/btac085. 7

[26] Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. Representation learning of knowledge graphs with entity descriptions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1), Mar. 2016. URL https://ojs.aaai.org/index.php/AAAI/article/view/10329. 7

[27] Pouya Pezeshkpour, Liyan Chen, and Sameer Singh. Embedding multimodal relational data for knowledge base completion. *CoRR*, abs/1809.01341, 2018. URL http://arxiv.org/abs/1809.01341. 8

[28] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 8

[29] Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. Low-dimensional hyperbolic knowledge graph embeddings. *CoRR*, abs/2005.00545, 2020. URL https://arxiv.org/abs/2005.00545. 8

[30] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality, 2013. 8

[31] Quoc V. Le and Tomás Mikolov. Distributed representations of sentences and documents. *CoRR*, abs/1405.4053, 2014. URL http://arxiv.org/abs/1405.4053. 8

[32] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper/2013/file/1cecc7a77928ca8133fa24680a88d2f9-Paper.pdf. 8

[33] Lloyd S Shapley. *17. A value for n-person games*. Princeton University Press, 2016. 12