Embedding Knowledge Graphs in Degenerate Clifford Algebras

Louis Mozart Kamdem Teyou, Caglar Demir and Axel-Cyrille Ngonga Ngomo

Department of Computer Science, Paderborn University
ORCID (Louis Mozart Kamdem Teyou): https://orcid.org/0000-0001-7975-8794, ORCID (Caglar Demir): https://orcid.org/0000-0001-8970-3850, ORCID (Axel-Cyrille Ngonga Ngomo): https://orcid.org/0000-0001-7112-3516

Abstract. Clifford algebras are a natural extension of division algebras, including real numbers, complex numbers, quaternions, and octonions. Previous research in knowledge graph embeddings has focused exclusively on Clifford algebras of a specific type, which do not include nilpotent base vectors—elements that square to zero. In this work, we introduce a novel approach by incorporating nilpotent base vectors with a nilpotency index of two, leading to a more general form of Clifford algebras named degenerate Clifford algebras. This generalization to degenerate Clifford algebras does allow for covering dual numbers and as such include translations and rotations models under the same generalization paradigm for the first time. We develop two models to determine the parameters that define the algebra: one using a greedy search and another predicting the parameters based on neural network embeddings of the input knowledge graph. Our evaluation on seven benchmark datasets demonstrates that this incorporation of nilpotent vectors enhances the quality of embeddings. Additionally, our method outperforms state-of-the-art approaches in terms of generalization, particularly regarding the mean reciprocal rank achieved on validation data. Finally, we show that even a simple greedy search can effectively discover optimal or near-optimal parameters for the algebra.

1 Introduction

Knowledge graphs (KGs) are used in an increasing number of applications and domains [24]. While several formalizations of KGs exist [12], we consider knowledge graphs $K \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, where $\mathcal{E}$ and $\mathcal{R}$ represent a set of entities and relations respectively. The elements of a knowledge graph are called *assertions* (sometimes also facts), and are triples $\langle \mathtt{x}, \mathtt{y}, \mathtt{z} \rangle$ where $\mathtt{x}$ is called the head (also called subject), $\mathtt{y}$ the relation (also called predicate) and $\mathtt{z}$ the tail (also called object) of the assertion [12]. For example, a KG may contain the triple $\langle \mathtt{Berlin}, \mathtt{capitalOf}, \mathtt{Germany} \rangle$, which states that Berlin is the capital of Germany. This triple can be used to answer questions such as "What is the capital of Germany?" or "What is Berlin?" [6]. While knowledge graphs have existed since for decades [12], the term was popularized by Google in 2012 [20], and its use has since surged [6, 12].

KGs are often embedded into vector spaces to make them amenable to classical machine learning [24]. While initial approaches operated in $\mathbb{R}$ [14], it is evident from the existing literature that other division algebras facilitate the modelling of complex relations patterns, e.g., symmetry, and asymmetry. For example, the ability to model symmetric and asymmetric relations is conferred to ComplEx [22] by its use of complex numbers. Recent embedding models have hence moved from real numbers to more complex number systems such as $\mathbb{C}$, $\mathbb{H}$, multi-vectors [26] and even Clifford Algebras $Cl_{p,q}(\mathbb{R})$ [9].

In particular, embeddings in Clifford algebras $Cl_{p,q}(\mathbb{R})$ have recently been shown to achieve a significant improvement over the state of the art when used in combination with dimension scaling thanks to their ability to generalize over all normed division algebras [9] (see Table 1 for more details). The resulting approach, KECI, was shown to be a strict generalization of existing multiplicative embeddings approaches such as DistMult and ComplEx. However, none of the approaches based on dual numbers (e.g., [3]) could be generalized by this approach. This paper addresses this weakness by discarding the assumption of a non-degenerate quadratic form Q, which underpins KECI (see Section 4 for more details). In contrast, we assume that the quadratic form Q that underpins our algebra can be degenerate, thus leading to r base vectors being degenerate (e.g., nilpotent vectors e_k with $e_k^2 = 0$). Our novel embedding approach, dubbed DECAL (Embedding in degenerate Clifford algebras), thus computes embeddings in $Cl_{p,q,r}(\mathbb{R})$.

2 Related Work

Knowledge graph embeddings (KGE) models typically map an input KG K into a low-dimensional and continuous vector space. According to the review in [6], the plethora of KGE methods existing in the literature can be categorized into two main groups: translational models and multiplicative models. One of the first translational models is TransE [2]: Given an assertion $\langle \mathtt{x}, \mathtt{y}, \mathtt{z} \rangle \in K$, TransE's idea boils down to optimizing for $\mathbf{x} + \mathbf{y} \approx \mathbf{z}$ if $\langle \mathtt{x}, \mathtt{y}, \mathtt{z} \rangle$ holds, where $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbb{R}^d$ and represent the embeddings of $\mathtt{x}, \mathtt{y}$ and $\mathtt{z}$ respectively. The publication of TransE led to the development of several other similar models (e.g., TransH, TransR, TransD and RotatE) that address some of its main shortcomings, e.g., its poor modeling of reflexive, one-to-many, many-to-one and many-to-many relationships [6, 13, 25]. TransH [25] embeds knowledge graph by projecting entities and relations onto a hyperplane that is specific to each relation. Doing so allows TransH to effectively capture the mapping properties of relations, such as one-to-many and many-to-one, which TransE cannot handle [6, 25]. TransR [18] introduces separate spaces for en-

tities and relations, connected by a shared projection matrix. TransD [13] uses independent projection vectors for each entity and relation, which reduces the amount of computation compared to TransR. RotatE [21] embeds the entities and relations into complex space and replaces the addition in TransE with the complex multiplication.

Multiplicative models like RESCAL, DistMult, ComplEx, QuatE, and OctE use bilinear transformations to score triples. RESCAL [19] employs a scoring function $\mathbf{x}^T \mathbf{A}_\mathbf{y} \mathbf{z}$, where $\mathbf{x}$ and $\mathbf{z}$ are entity embeddings, and $\mathbf{A}_\mathbf{y}$ is the relation matrix. DistMult [27] simplifies this by using a diagonal relation matrix, $\mathbf{D}_\mathbf{y} = \text{diag}(\mathbf{y})$, which is effective for symmetric relations but struggles with anti-symmetric ones. ComplEx [23] addresses this by embedding entities and relations in the complex space $\mathbb{C}$, using the real part for symmetry and the imaginary part for anti-symmetry. QuatE [28] extends these capabilities to the quaternion space $\mathbb{H}$, allowing for the modeling of complex relationships such as inversion. OctE builds on QuatE by operating in the octonion space $\mathbb{O}$. However, both QuatE and OctE can face scaling challenges inherent to quaternion and octonion spaces. QMult and OMult [10] address this limitation through batch normalization, effectively mitigating the bottleneck.

In addition to translational and multiplicative models, the literature also features hyperbolic embedding methods such as RotH [5] and MuRP [1], which maps entities and relations from a knowledge graph onto a hyperbolic space, leveraging the properties of hyperbolic geometry to capture hierarchical structures in KGs. Dual quaternion methods like DualE [4] use dual quaternions [15] to embed entities and relations, offering a representation that combines the advantages of both hyperbolic and complex spaces. Rotational methods like RotE [5] focus on learning rotation operations to capture relational patterns, while Euclidean methods such as MuRE [1], which is a variant of MuRP, operate in Euclidean space, offering a simpler alternative for certain types of knowledge graphs.

3 Motivation and Contribution

The choice of the sub-algebra or space for embedding given any input knowledge graph plays a crucial role in computing an accurate representation of the input data as indicated in [9]. For instance, if a KG does not contain anti-symmetric relations, employing a complex-valued approach like ComplEx is likely to be less effective than using a simpler real-valued approach like DistMult. To address this, Demir et al. [9] proposed incorporating the sub-algebra selection into the learning process by performing embeddings in Clifford Algebras $Cl_{p,q}(\mathbb{R})$ via the KECI model. Thanks to parameters p and q, the KECI model is able to determine which space is appropriate to embed an input knowledge graph. As shown in Table 1, KECI can decide or not if the embeddings will be carried out in $\mathbb{R}$, $\mathbb{C}$, $\mathbb{H}$, $\mathbb{O}$ and even beyond (for instance by setting $p = 3$ and $q = 4$).

We build upon this idea via two main contributions:

- We drop KECI's assumption that the quadratic form underlying our Clifford algebra must not be degenerate and show how to embed even in degenerate Clifford algebras. Therewith, we can generalize over approaches based on dual numbers in addition to generalizing over KECI itself giving rise to more degree of freedom for our approach.
- Moreover, we address the main weakness of dimension scaling: low dimension weights mean that particular dimensions barely contribute to the total score of KECI. Instead of learning p and q concurrently to the optimizing of the embeddings—hence effectively discarding dimensions without replacement—we present 2 approaches to predict p, q, and r before we run DECAL, and hence make full use of all dimensions available.
- Our implementation of DECAL is publicly available to ensure that all results and experiments presented herein can be replicated. [1]

Table 1: Relation between Clifford algebras and division algebras. N/A indicates no possible relationship between the spaces.

Space	$\subseteq$	$Cl_{p,q}(\mathbb{R})$	$\equiv$	$Cl_{p,q,0}(\mathbb{R})$
$\mathbb{R}$		$Cl_{0,0}(\mathbb{R})$		$Cl_{0,0,0}(\mathbb{R})$
$\mathbb{C}$		$Cl_{0,1}(\mathbb{R})$		$Cl_{0,1,0}(\mathbb{R})$
$\mathbb{H}$		$Cl_{0,2}(\mathbb{R})$		$Cl_{0,2,0}(\mathbb{R})$
$\mathbb{O}$		$Cl_{1,3}(\mathbb{R})$		$Cl_{1,3,0}(\mathbb{R})$
$\mathbb{R}(\epsilon)$		N/A		$Cl_{0,0,1}(\mathbb{R})$
$\mathbb{R}^4(\epsilon)$		N/A		$Cl_{0,3,1}(\mathbb{R})$

In the following sections, we present the technical details of DECAL and provide evidence of its efficacy in link prediction of KG tasks.

4 Clifford Algebras

4.1 Definition

A Clifford Algebra, denoted as $Cl(V, Q)$ [8, 7], is generated by a vector space V and a quadratic form Q. When V is a real vector space and Q is a non-degenerate quadratic form, $Cl(V, Q)$ can be represented as $Cl_{p,q}(\mathbb{R})$ (where p and q are natural integers), signifying that V possesses an orthogonal basis with $p + q$ vectors. Here, p vectors e_i satisfy $e_i^2 = 1$, and q vectors e_j satisfy $e_j^2 = -1$. The space $Cl_{p,q}(\mathbb{R})$ thereby extends traditional spaces such as the real and the complex space (see Table 1). In this paper, *we refrain from assuming that the quadratic form Q is non-degenerate*. Then, the orthogonal basis of our algebra consists of $p + q + r$ vectors, of which p vectors (denoted e_i) are such that $e_i^2 = 1$, q vectors (denoted e_j) are such that $e_j^2 = -1$, and r vectors (denoted e_k) are such that $e_k^2 = 0$. Hence, the space $Cl(V, Q)$ is now denoted $Cl_{p,q,r}(\mathbb{R})$. Clearly, the algebra $Cl_{p,q,r}(\mathbb{R})$ is a super-algebra of $Cl_{p,q}(\mathbb{R})$ for any $p, q, r \geq 0$.

4.2 Norm

We consider $\mathbf{x} \in Cl_{p,q,r}(\mathbb{R}^d)$ that we represent using $p + q + r$ orthogonal vectors as $\mathbf{x} = x_0 + \sum_{i=1}^{p} x_i e_i + \sum_{j=p+1}^{p+q} x_j e_j + \sum_{k=p+q+1}^{p+q+r} x_k e_k$, where $x_{(\cdot)} \in \mathbb{R}^{\lfloor \frac{d}{1+p+q+r} \rfloor}$. The norm of $\mathbf{x}$, denoted $\|\mathbf{x}\|$, is defined using the quadratic form Q that govern $Cl_{p,q,r}$. Since Q is degenerate, the norm only operates on non-degenerate vectors [17] , i.e.

$$\|\mathbf{x}\|^2 = Q(\mathbf{x}) = x_0^2 + \sum_{i=1}^{p} x_i^2 e_i^2 - \sum_{j=p+1}^{p+q} x_j^2 e_j^2 = \sum_{i=0}^{p+q} x_i^2. \quad (1)$$

4.3 Inner Product and Clifford Product

If we take also $\mathbf{y} \in Cl_{p,q,r}(\mathbb{R}^d)$ such that

$$\mathbf{y} = y_0 + \sum_{i=1}^{p} y_i e_i + \sum_{j=p+1}^{p+q} y_j e_j + \sum_{k=p+q+1}^{p+q+r} y_k e_k, \quad (2)$$

the inner product $\mathbf{x} \cdot \mathbf{y}$ between $\mathbf{x}$ and $\mathbf{y}$ is given by

$$x_0 \cdot y_0 + \sum_{i=1}^{p} x_i \cdot y_i + \sum_{j=p+1}^{p+q} x_j \cdot y_j + \sum_{k=p+q+1}^{p+q+r} x_k \cdot y_k.$$

[1] https://github.com/dice-group/dice-embeddings

The Clifford product $\mathbf{x} \circ \mathbf{y}$ between vectors $\mathbf{x}$ and $\mathbf{y}$ involves element-wise multiplication of their components across different basis elements (see supplementary material [16] for mathematical details).

5 Approach

To ensure comparability with standard approaches, we address the embedding problem in a d-dimensional vector space. Consequently, DistMult performs embeddings into $\mathbb{R}^d$, ComplEx into $\mathbb{C}^{d/2}$, QMult (which has demonstrated slightly better results than QuatE) and OMult (which has shown slightly better results than OctE) [10] into $\mathbb{H}^{d/4}$ and $\mathbb{O}^{d/8}$ respectively). Additionally, KECI embeds into $Cl_{p,q}(\mathbb{R}^{m'})$, where $m' = \lfloor \frac{d}{1+p+q} \rfloor$, and DECAL into $Cl_{p,q,r}(\mathbb{R}^m)$, with $m = \lfloor \frac{d}{1+p+q+r} \rfloor$ (see Table 3).

5.1 Embedding in Degenerate Clifford Algebras

Considering a triple $\langle \mathrm{x}, \mathrm{y}, \mathrm{z} \rangle \in K$, we represent the embeddings $\mathbf{x}$ and $\mathbf{y}$ of x and y in the space $Cl_{p,q,r}(\mathbb{R}^m)$ as:

$$\mathbf{x} = x_0 + \sum_{i=1}^{p} x_i e_i + \sum_{j=p+1}^{p+q} x_j e_j + \sum_{j=p+q+1}^{p+q+r} x_k e_k \tag{3}$$

$$\mathbf{y} = y_0 + \sum_{i=1}^{p} y_i e_i + \sum_{j=p+1}^{p+q} y_j e_j + \sum_{j=p+q+1}^{p+q+r} y_k e_k, \tag{4}$$

with $x_{(\cdot)}, y_{(\cdot)} \in \mathbb{R}^m$ and

$$\begin{cases} e_i^2 = 1, \ \forall i \in \{1, \cdots, p\} \\ e_j^2 = -1, \ \forall j \in \{p+1, \cdots, p+q\} \\ e_k^2 = 0, \ \forall k \in \{p+q+1, \cdots, p+q+r\} \\ e_\ell e_n = -e_n e_\ell, \ \forall n \neq \ell. \end{cases} \tag{5}$$

Using Equations 5 and adopting the notations from [9] the Clifford multiplication between the head and relation representation is:

$$\begin{aligned} \mathbf{x} \circ \mathbf{y} = \sigma_0 + \sigma_p + \sigma_q + \sigma_r + \sigma_{p,p} + \sigma_{q,q} + \sigma_{r,r} + \sigma_{p,q} + \\ \sigma_{p,r} + \sigma_{q,r}. \end{aligned}$$

The terms σ_0, σ_p, σ_q, $\sigma_{p,p}$, $\sigma_{q,q}$, and $\sigma_{p,q}$ are as in [9], with new terms derived in supplementary material [16] and defined as follows:

$$\sigma_r = \sum_{k=p+q+1}^{p+q+r} (x_0 y_k + x_k y_0) \, e_k \tag{6}$$

$$\sigma_{r,r} = \sum_{k=p+q+1}^{p+q+r-1} \sum_{k'=k+1}^{p} (x_k y_{k'} - x_{k'} y_k) e_k e_{k'} \tag{7}$$

$$\sigma_{p,r} = \sum_{i=1}^{p} \sum_{k=p+q+1}^{p+q+r} (x_i y_k - x_k y_i) e_i e_k \tag{8}$$

$$\sigma_{q,r} = \sum_{j=p+1}^{p+q} \sum_{j=p+q+1}^{p+q+r} (x_j y_k - x_k y_j) e_j e_k. \tag{9}$$

5.2 Scoring Function Derivation

The scoring function of DECAL consists of taking the Clifford multiplication between the embeddings of the head and the relation, followed by a scalar product with the tail embeddings $\mathbf{z}$ i.e.

$$\text{DECAL}\big(\langle \mathrm{x}, \mathrm{y}, \mathrm{z} \rangle\big) = (\mathbf{x} \circ \mathbf{y}) \cdot \mathbf{z}. \tag{10}$$

Since the Clifford multiplication $\mathbf{x} \circ \mathbf{y}$ generate multi-vectors coordinates, we represent $\mathbf{z}$ in order to perform the above scalar product as

$$\mathbf{z} = z_0 + \sum_{i=1}^{p} z_i e_i + \sum_{j=p+1}^{p+q} z_j e_j + \sum_{j=p+q+1}^{p+q+r} z_k e_k + \vec{Cte} \tag{11}$$

where $z_{(\cdot)} \in \mathbb{R}^m$ and $\vec{Cte}$ represent a unitary multi-vector [16]. That is, in the tail representation, all coefficients of the multi-vectors involved in the scalar product are set to one. Hence, the scoring function of DECAL can be deduced from the score of KECI as,

$$\begin{aligned} \text{DECAL}\big(\langle \mathrm{x}, \mathrm{y}, \mathrm{z} \rangle\big) = \text{KECI}\big(\langle \mathrm{x}, \mathrm{y}, \mathrm{z} \rangle\big) + \sigma^*_{r,r} + \sigma^*_{p,r} + \sigma^*_{q,r} \\ + \sum_{k=p+q+1}^{p+q+r} (x_0 y_k z_k + x_k y_0 z_k), \end{aligned}$$

where $\sigma^*_{r,r}, \sigma^*_{p,r}$ and $\sigma^*_{q,r}$ represents the sum of the coordinates of the multi-vectors $\sigma_{r,r}, \sigma_{p,r}$ and $\sigma_{q,r}$ respectively.

6 Embedding Space Search

Finding adequate values for p, q, and r is bound to be of central importance when embedding using DECAL. An exhaustive search over these parameters results in $\frac{1}{6}(d+1)(d+2)(d+3) \in O(d^3)$ possible combinations, where d represents the embedding dimension. For instance, with $d = 16$, the model would need to be run 969 times, proving an inefficient and time-consuming approach. To address this challenge, we developed four strategies to navigate the parameter space defined by p, q, and r. While two of these serve as baseline approaches, the other two represent our key contributions, focusing on finding configurations that yield the highest mean reciprocal rank on the validation data by learning from the training set and applying this combination on the test set.

Local Exhaustive Search (LES) The local exhaustive search involves systematically exploring all potential parameter (without constraint) combinations for DECAL in a subspace of the parameter space $\{0, 1, \cdots, d\}^3$ with $0 \leq p + q + r \leq d$.

Global Search with Divisibility Criterion (GSDC) This approach subsamples the space covered by the local exhaustive search by only visiting spaces where $(1 + p + q + r)$ divides d. The motivation behind this approach is to only consider configuration which fully exploits the total number of dimensions available to DECAL. For $d = 16$, the approach visits 186 Clifford algebras.

Greedy Search (GS) Our first approach to optimize the hyperparameters p, q and r of DECAL consists of the greedy search algorithm described in Algorithm 1. The Algorithm starts with the initial configuration $(p, q, r) = (1, 1, 1)$ and iteratively generates new configurations (unseen configurations) in a local neighbourhood by adding 1, -1 or 0 to p, q, and r. Then, we evaluate the mean reciprocal rank (short: MRR) for each unseen configuration and append these scores to a queue which is then sorted in descending order based on the MRR. The next configuration to evaluate is then selected from the highest-scoring configurations. This process is repeated until convergence or the maximum number of iterations is reached. The algorithm terminates if the best configuration remains unchanged, indicating a local maximum.

A major limitation of this approach is that its can only detect local maxima close to its starting point. While it performs well in practice (see Section 7), we devise another approach able to optimize p, q, and r using global information.

Algorithm 1 GreedySearch

function OPTIMAL_PARAMS(max_iterations)
  $p_0, q_0, r_0 \leftarrow 1, 1, 1$ ▷ Initialization
  seen_conf $\leftarrow$ []
  prior_query $\leftarrow$ []
  for $i \in [0, \text{max\_iterations})$ **do**
    to_score $\leftarrow$ GENERATECONF(seen_conf, p, q, r)
    prior_query $\leftarrow$ SCORE(to_score, prior_query)
    prior_queue $\leftarrow$ sort(prior_query)
    p, q, r, max_MRR $\leftarrow$ prior_queue[0]
    if $(p, q, r) = (p_0, q_0, r_0)$ **then**
      break ▷ Local maximum found
    else
      $p_0, q_0, r_0 \leftarrow p, q, r$ ▷ Update parameters
      seen_conf = seen_conf $\cup$ to_score
    end if
  end for
  return $(p, q, r, \text{max\_MRR})$
end function
function GENERATECONF(queue, p, q, r)
  $\ell \leftarrow$ []
  for $p_i, q_i, r_i \in [-1, 0, 1]$ **do**
    if $(p + p_i, q + q_i, r + r_i) \notin$ queue **then**
      $\ell = \ell \cup \{(p + p_i, q + q_i, r + r_i)\}$
    end if
  end for
  return ℓ
end function
function SCORE(queue, prior)
  for $(p, q, r) \in$ queue **do**
    if $(p \geq 0) \wedge (q \geq 0) \wedge (r \geq 0)$ **then**
      prior = prior $\cup \{(p, q, r, \text{MRR}(p, q, r))\}$
    end if
  end for
  return prior_queue
end function
function MRR(p, q, r)
  return DECAL.MRR(p,q,r) ▷ MRR Evaluation
end function

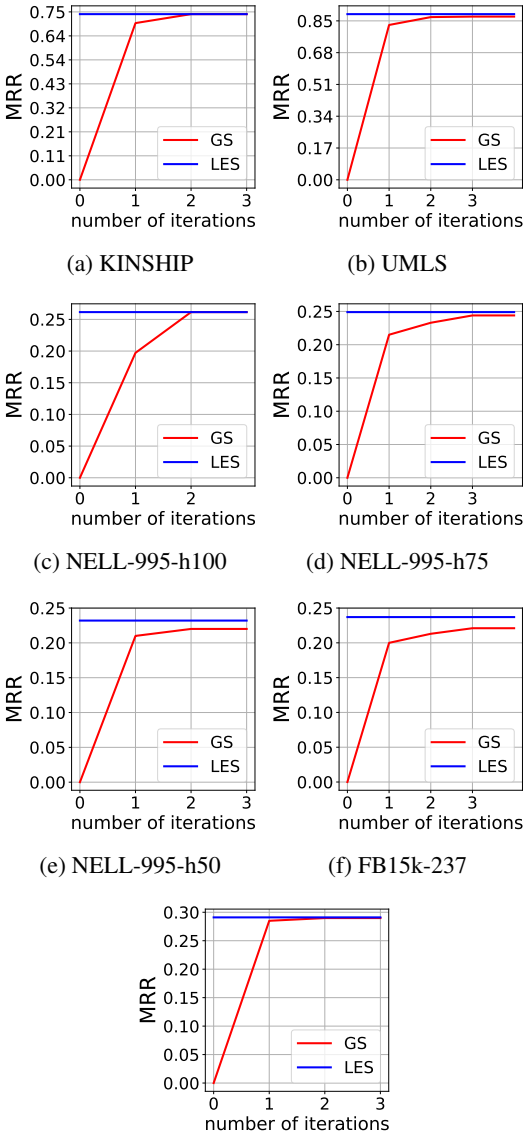
Vector Space Prediction (VSP) Given an input knowledge graph K, this approach aims to predict the optimal values of p, q, and r for K w.r.t. the MRR. To achieve this goal, our approach begins by computing an embedding of K in $Cl_{1,1,1}$. The embedding of each triple $\langle \mathtt{x}, \mathtt{y}, \mathtt{z} \rangle \in K$ is the concatenation of the embedding of $\mathtt{x}$, $\mathtt{y}$ and $\mathtt{z}$. The embeddings of all triples in K are finally used as input for a pre-trained recurrent neural network, which predicts the values of p, q and r for K.

7 Results and Discussion

7.1 Datasets and Experimental Setup

We evaluate our model on seven benchmark datasets, comprising five large datasets (WN18-RR, FB15k-237, NELL-995-h100, NELL-995-h50, and NELL-995-h75) and two smaller datasets (UMLS, and KINSHIP). For detailed statistics on these datasets, please refer to Table 2.

We conducted two series of experiments. First, we wanted to quantify how the different approaches for determining p, q and r performed. To this end, we evaluated the combination of DECALwith



(a) KINSHIP (b) UMLS

(c) NELL-995-h100 (d) NELL-995-h75

(e) NELL-995-h50 (f) FB15k-237

(g) WN18-RR

Figure 1: GS convergence speed to LES across benchmark datasets.

Table 2: Overview of benchmark datasets.

Dataset	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{G}^{\text{Train}}\rvert$	$\lvert\mathcal{G}^{\text{Validation}}\rvert$	$\lvert\mathcal{G}^{\text{Test}}\rvert$
WN18-RR	40,943	22	86,835	3,034	3,134
FB15k-237	14,541	237	272,115	17,535	20,466
NELL-995-h50	34,667	86	72,767	5,440	5,393
NELL-995-h75	28,085	114	59,135	4,441	4,389
NELL-995-h100	22,411	86	50,314	3,763	3,746
UMLS	135	46	5,216	652	661
KINSHIP	104	25	8,544	1,068	1,074

the approaches presented in Section 6. For the vector space prediction (VSP), we used a leave-one-out training setting, where we used 1000 subgraphs containing 5000 triples from 6 benchmark datasets for training and the remaining dataset for testing. We trained three distinct models: LSTM (Long Short-Term Memory), GRU (Gated Recurrent Unit), and a concatenation of both [11].

In our second series of experiments, we conducted a comparison between DECAL and state-of-the-art algorithms. we reported the performance of each algorithm on the test data, and for results on train

Table 3: Embedding spaces and model time complexity comparison. GSDC serves as the reference method, with relative times indicating the efficiency of each model compared to GSDC. Note that the recorded time of VSP reflects only its prediction phase.

Models	Embedding Space	Relative Time[2]
GSDC	$Cl_{p,q,r}(\mathbb{R}^m), m = \lfloor \frac{d}{1+p+q+r} \rfloor$	100 %
LES		67.6%
GS		24.3%
VSP		5.2%
Keci	$Cl_{p,q}(\mathbb{R}^m),\ m = \lfloor \frac{d}{1+p+q} \rfloor$	56.7%
DistMult	$\mathbb{R}^d$	10.1%
RotE		12.5%
MuRE		20.2%
MuRP		22.4%
CompEx	$\mathbb{C}^{d/2}$	10.9%
QMult	$\mathbb{H}^{d/4}$	11.4%
DualE	$\mathbb{R}^d(\epsilon)$	12.5%
OMult	$\mathbb{O}^{d/8}$	11.6%

and validation data, refer to the supplementary material [16]. We used the same set of parameters for all approaches: $d = 16$, number of epochs = 250, batch size = 1024, learning rate = 0.1, Adam optimizer and the KvsAll training technique. Like previous works [24, 9], we used the Mean Reciprocal Rank (MRR) and hits at 1, 3, and 10 as performance measures. All experiments presented in this paper were conducted on a virtual machine equipped with two NVIDIA H100-80C GPUs, each with 80 GB of memory.

7.2 *Comparison of Different Variants*

We evaluated the performance of DECAL in combination with different strategies for discovering p, q, and r. An overview of these results can be found in the four bottom lines of Tables 5 and 4. The found Clifford space as well as the computational time of the different search algorithms of DECAL and KECI on all data sets can be found in Table 1 of the supplementary material [16]. The combinations of DECAL with LES and GSDC can be regarded as upper bounds of the performance of our algorithm. Our approach outperformed all other variants (as well as the state of the art) when combined with GSDC except on UMLS, where DECAL + LES performed best. We conclude that GSDC is the approach of choice when aiming to determine useful values for p, q and r. However, the number of combinations of these parameters that need to be checked can be prohibitively large, especially if d is a highly composite number.

In our experiments, using GS was less time-consuming than GSDC (see Table 3) but led to slightly worse results w.r.t. the MRR on the benchmark datasets. For example, GS outperformed GSDC on the UMLS dataset, achieved the same performance as GSDC on KINSHIP, and reached over 91.6% of GSDC's performance on average. This suggests that GS can indeed be used for configuring DECAL if the number of combinations of p, q, r to explore is to be kept low. Note that GS only needed at most three iterations in our experiments to find a local maximum (see Figure 1). This strength of GS is also its main weakness as the convergence of this method depends significantly on the initial starting point. The MRR function as shown in Figure 1 in the supplementary material [16], shows multiple local maxima, making the search for the global maximum challenging. Still, our results indicate that local maxima generally suffice to achieve a state-of-the-art performance.

Figure 2 illustrates the loss evolution during the training phase for each model on all datasets with FB15k-237 left out. On the test data, the LSTM, GRU, and concatenated models achieved a prediction accuracy of 44.5%, 42.0%, and 35.5%, respectively, where a prediction was considered correct if it returned the (p, q, r) combination suggested by LES. Employing an ensemble approach with weights of 0.75, 0.2, and 0.05 for the LSTM, GRU, and concatenated models, our predictor achieved a superior performance of 45.0%. The combination of the concatenated model of VSP and DECAL turned out to achieve approximately 84.5% of the MRR of DECAL + GDSC on average. The results are not surprising given that the approach was trained on a small number of samples. We hypothesize that the performance of the approach can be much improved with the availability of more diverse training data. Still, the leave-one-out strategy we employed for learning suggests that a universal predictor can indeed be trained to predict suitable values of the Clifford algebra parameters. This will be the subject of future works.

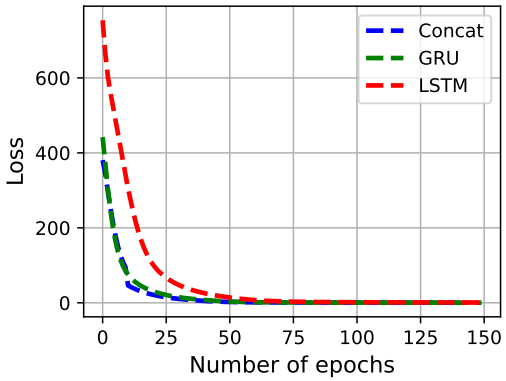


Figure 2: Loss function curves.

Table 4: Link prediction results on WN18-RR at the test time. Bold and underlined results indicate the best and second-best results respectively.

Models	WN18-RR			
	MRR	H@1	H@3	H@10
DistMult	0.231	0.191	0.249	0.305
ComplEx	0.274	0.233	0.295	0.349
QMult	0.242	0.199	0.262	0.318
OMult	0.121	0.080	0.135	0.203
MuRE	0.259	0.202	0.280	0.366
MuRP	0.212	0.160	0.234	0.309
RotH	0.230	0.177	0.247	0.332
RotE	0.258	0.203	0.280	0.357
DualE	0.157	0.118	0.174	0.228
KECI	0.285	0.245	0.302	0.357
DECAL + LES	0.291	0.249	0.313	0.364
DECAL + GSDC	**0.296**	<u>0.250</u>	**0.323**	**0.381**
DECAL + GS	<u>0.295</u>	**0.252**	<u>0.317</u>	<u>0.372</u>
DECAL + VSP	0.285	0.245	0.302	0.357

7.3 *Comparison with other Approaches*

Tables 5 and 4 present link prediction results on the datasets presented in Table 2. In the following, we mainly focus on the performance of DECAL + GSDC and DECAL + GS when comparing our

Table 5: Link prediction results on UMLS, KINSHIP, NELL-995-h100, NELL-995-h75, NELL-995-h50 and FB15k-237 on the test data. Bold and underlined results indicate the best and second-best results respectively.

Models	**UMLS**				**KINSHIP**				**NELL-995-h100**			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
DistMult	0.756	0.647	0.834	0.952	0.514	0.346	0.606	0.878	0.185	0.120	0.205	0.320
ComplEx	0.835	0.728	0.939	0.976	0.725	0.594	0.821	0.963	0.168	0.109	0.187	0.285
QMult	0.859	0.773	0.936	0.980	0.624	0.481	0.715	0.911	0.125	0.079	0.135	0.214
OMult	0.827	0.756	0.874	0.954	0.488	0.365	0.536	0.747	0.137	0.088	0.151	0.237
MuRE	0.876	0.782	**0.964**	0.990	0.664	0.526	0.754	0.933	0.236	0.167	0.261	0.374
MuRP	0.874	**0.824**	0.908	0.970	0.676	0.532	0.779	0.946	0.237	0.161	0.261	0.395
RotH	0.802	0.689	0.899	0.981	0.592	0.427	0.689	0.932	0.192	0.132	0.214	0.310
RotE	0.866	0.766	0.962	**0.994**	0.709	0.569	0.814	**0.967**	0.225	0.157	0.250	0.361
DualE	0.866	0.774	0.955	0.985	0.591	0.443	0.675	0.905	0.173	0.113	0.188	0.295
KECI	0.875	0.798	0.944	0.983	**0.743**	**0.621**	**0.830**	0.964	0.252	0.174	0.288	0.408
DECAL + LES	**0.883**	0.799	0.962	0.991	**0.743**	**0.621**	**0.830**	0.964	0.256	0.184	0.280	0.399
DECAL + GSDC	0.878	0.799	0.947	0.989	**0.743**	**0.621**	**0.830**	0.964	**0.270**	**0.196**	**0.299**	**0.417**
DECAL + GS	0.879	0.797	0.951	0.987	**0.743**	**0.621**	**0.830**	0.964	0.256	0.184	0.280	0.399
DECAL + VSP	0.843	0.732	0.944	0.986	0.674	0.540	0.766	0.942	0.235	0.167	0.259	0.374

Models	**NELL-995-h75**				**NELL-995-h50**				**FB15k-237**			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
DistMult	0.163	0.104	0.179	0.282	0.151	0.097	0.168	0.256	0.147	0.093	0.158	0.260
ComplEx	0.139	0.085	0.156	0.245	0.163	0.107	0.183	0.275	0.137	0.087	0.147	0.239
QMult	0.158	0.106	0.171	0.261	0.101	0.0596	0.114	0.185	0.122	0.077	0.128	0.208
OMult	0.129	0.082	0.142	0.223	0.119	0.076	0.131	0.210	0.086	0.059	0.089	0.138
MuRE	0.229	0.163	0.256	0.357	0.215	0.151	0.237	0.345	0.209	0.147	0.224	0.333
MuRP	0.230	0.161	0.255	0.370	0.217	0.151	0.239	0.348	0.204	0.141	0.220	0.329
RotH	0.163	0.104	0.184	0.279	0.154	0.098	0.169	0.270	0.144	0.097	0.151	0.233
RotE	0.208	0.144	0.230	0.335	0.218	0.152	0.243	0.349	0.197	0.133	0.211	0.324
DualE	0.178	0.120	0.196	0.291	0.176	0.116	0.195	0.298	0.185	0.125	0.199	0.299
KECI	0.237	0.172	0.260	0.365	**0.250**	**0.177**	**0.281**	**0.392**	0.235	0.164	0.255	0.376
DECAL + LES	0.245	0.177	0.274	0.374	0.232	0.166	0.253	0.368	0.235	0.164	0.258	0.375
DECAL + GSDC	**0.251**	**0.178**	**0.281**	**0.396**	**0.250**	**0.177**	**0.281**	**0.392**	**0.241**	**0.171**	**0.263**	**0.380**
DECAL + GS	0.230	0.170	0.268	0.376	0.163	0.106	0.183	0.273	0.217	0.150	0.236	0.349
DECAL + VSP	0.202	0.136	0.225	0.335	0.203	0.138	0.229	0.333	0.144	0.091	0.156	0.251

approach with the state of the art.

The prediction results on the WN18RR datasets are shown in Table 4. The findings indicate that all variations of our approach outperformed the state-of-the-art model in all metrics, except for DECAL + VSP, which predicted the same space for embeddings as KECI, resulting in similar performance.

The prediction results for all other datasets are given in Table 5. In the upper part of the table, we display the link prediction results for UMLS, KINSHIP, and NELL-995-h100. On UMLS, DECAL + GS showed the second-best MRR, trailing behind DECAL + LES. MuRP, MuRE and RotH achieved the highest performance for Hits at 1, 3 and 10 respectively. Notably, DECAL + GSDC performed less effectively than DECAL+LES only on UMLS data.

For the KINSHIP dataset, nearly all variations of our approach— except for DECAL + VSP— find the embedding space $Cl_{0,1,0}(\mathbb{R}^8)$ to be the most fitting. Due to the relationship between $Cl_{0,1,0}$, $Cl_{0,1}$, and $\mathbb{C}$ (refer to Table 1), the performances of DECAL + GSDC, DECAL + GS, and KECI were similar. However, the scaling effect used by KECI [9] resulted in slightly superior performance, making ComplEx the second-best. Similar observations can be made for the NELL-995-h50 dataset in the bottom part of the table, where DECAL + GSDC performed embeddings into $Cl_{13,2,0}(\mathbb{R})$ which is theoretically isomorphic to $Cl_{13,2}(\mathbb{R})$, resulting in similar performance to KECI. Surprisingly, we remark that this is the only dataset where DECAL + GS performed less than DECAL + VSP, showing very close performance to Complex and better results than DistMult, OMult, QMult and DualE.

Moving on to NELL-995-h100, DECAL + GSDC achieved the highest performance, surpassing KECI by over 6% in MRR. The second-best results were shared between DECAL + LES and DECAL + GS, which had similar performances.

On FB15k-237, DECAL + GSDC demonstrated superior performance across all metrics. Remarkably, its Mean Reciprocal Rank (MRR) score is, on average, 40% better than that of other approaches. While DECAL + LES and KECI exhibited identical performance in Hits at 1 and MRR, DECAL + LES took second place

overall. This is attributed to its outperformance of KECI by more than 1% in Hits at 3, despite KECI being only 0.2% better in Hits at 10. Additionally, it's noteworthy that this is the only dataset in the study where DistMult outperformed one variant of DECAL (DECAL + VSP) across all metrics. This suggests that nilpotent vectors help capture the embeddings better on this dataset.

8 Conclusion and Future Work

In this paper, we introduce DECAL, the first model specifically designed for embeddings in degenerate Clifford algebras. As a result, DECAL emerges as the most comprehensive divisional algebra in the literature for computing such embeddings.

We implemented four variants of DECAL for optimal parameter search. The first two variants, DECAL+LES and DECAL+GSDC, considered as baselines use exhaustive searches and serve as upper bounds for our approach, yielding the best possible results. The third variant, DECAL+GS, optimizes parameters within a subdomain of the parameter space, while the fourth variant, DECAL+VSP, employs neural networks to predict parameters based on input data.

Evaluation in link prediction tasks across seven benchmark datasets consistently demonstrates the superiority of DECAL over state-of-the-art models on all datasets. This underscores the potential of leveraging nilpotent vectors in Clifford algebras to enhance representation learning and inference capabilities in knowledge graphs.

Among the DECAL variants, VSP generally yields slightly inferior results; however, its predicted results are consistently superior to most of the state-of-the-art models. We believe that this performance could be significantly improved with additional training data, thereby motivating future work.

Furthermore, in this study, we used only single base vectors (i.e. $1 + p + q + r$ vectors) for entities and relations representation in $Cl_{p,q,r}(\mathbb{R})$ Clifford Algebras. Future work will explore the incorporation of multi-vectors, aiming to capture more intricate interactions in entities and relations by considering the full spectrum of the 2^{p+q+r} base vectors within a Clifford space $Cl_{p,q,r}(\mathbb{R})$.

Acknowledgements

This project received funding from the European Union's Horizon Europe research and innovation programme through two grants (Marie Skłodowska-Curie grant No. 101073307 and grant No. 101070305). It was also supported by the "WHALE" project (LFN 1-04), funded by the Lamarr Fellow Network programme and the Ministry of Culture and Science of North Rhine-Westphalia (MKW NRW).

References

[1] I. Balazevic, C. Allen, and T. Hospedales. Multi-relational poincaré graph embeddings. *Advances in Neural Information Processing Systems*, 32, 2019.

[2] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26, 2013.

[3] Z. Cao, Q. Xu, Z. Yang, X. Cao, and Q. Huang. Dual quaternion knowledge graph embeddings. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 6894–6902. AAAI Press, 2021. doi: 10.1609/AAAI.V35I8.16850. URL https://doi.org/10.1609/aaai.v35i8.16850.

[4] Z. Cao, Q. Xu, Z. Yang, X. Cao, and Q. Huang. Dual quaternion knowledge graph embeddings. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 6894–6902, 2021.

[5] I. Chami, A. Wolf, D.-C. Juan, F. Sala, S. Ravi, and C. Ré. Low-dimensional hyperbolic knowledge graph embeddings. *arXiv preprint arXiv:2005.00545*, 2020.

[6] Z. Chen, Y. Wang, B. Zhao, J. Cheng, X. Zhao, and Z. Duan. Knowledge graph completion: A review. *Ieee Access*, 8:192435–192456, 2020.

[7] K. Clifford. Clifford algebra. *Quantum Algebra and Symmetry: Quantum Algebraic Topology, Quantum Field Theories and Higher Dimensional Algebra*, page 127, 2010.

[8] W. K. Clifford. *Mathematical Papers by William Kingdon Clifford: Edited by Robert Tucker, with an introduction by HJ Stephen Smith*. Macmillan and Company, 1882.

[9] C. Demir and A.-C. Ngonga Ngomo. Clifford embeddings–a generalized approach for embedding in normed algebras. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 567–582. Springer, 2023.

[10] C. Demir, D. Moussallem, S. Heindorf, and A.-C. N. Ngomo. Convolutional hypercomplex embeddings for link prediction. In *Asian Conference on Machine Learning*, pages 656–671. PMLR, 2021.

[11] R. Fu, Z. Zhang, and L. Li. Using lstm and gru neural network methods for traffic flow prediction. In *2016 31st Youth academic annual conference of Chinese association of automation (YAC)*, pages 324–328. IEEE, 2016.

[12] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutierrez, S. Kirrane, J. E. L. Gayo, R. Navigli, S. Neumaier, A. N. Ngomo, A. Polleres, S. M. Rashid, A. Rula, L. Schmelzeisen, J. F. Sequeda, S. Staab, and A. Zimmermann. Knowledge graphs. *ACM Comput. Surv.*, 54(4):71:1–71:37, 2022. doi: 10.1145/3447772. URL https://doi.org/10.1145/3447772.

[13] 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)*, pages 687–696, 2015.

[14] S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Networks Learn. Syst.*, 33(2):494–514, 2022. doi: 10.1109/TNNLS.2021.3070843. URL https://doi.org/10.1109/TNNLS.2021.3070843.

[15] Y.-B. Jia. Dual quaternions. *Iowa State University: Ames, IA, USA*, 2013.

[16] L. M. KAMDEM TEYOU, C. Demir, and A.-C. Ngonga Ngomo. Appendix: Embedding Knowledge Graphs in Degenerate Clifford Algebras, Aug. 2024. URL https://doi.org/10.5281/zenodo.13341499.

[17] H. A. Keller and H. Ochsenius. The structure of norm clifford algebras. *Mathematica Slovaca*, 62(6):1105–1120, 2012.

[18] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29, 2015.

[19] M. Nickel, V. Tresp, H.-P. Kriegel, et al. A three-way model for collective learning on multi-relational data. In *Icml*, volume 11, pages 3104482–3104584, 2011.

[20] A. Singhal. Introducing the knowledge graph: Things, not strings. Google Blog, 2012. Retrieved from https://www.blog.google/products/search/introducing-knowledge-graph-things-not/.

[21] Z. Sun, Z.-H. Deng, J.-Y. Nie, and J. Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*, 2019.

[22] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard. Complex embeddings for simple link prediction. In M. Balcan and K. Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org, 2016. URL http://proceedings.mlr.press/v48/trouillon16.html.

[23] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard. Complex embeddings for simple link prediction. In *International conference on machine learning*, pages 2071–2080. PMLR, 2016.

[24] Q. Wang, Z. Mao, B. Wang, and L. Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Trans. Knowl. Data Eng.*, 29(12):2724–2743, 2017. doi: 10.1109/TKDE.2017.2754499. URL https://doi.org/10.1109/TKDE.2017.2754499.

[25] 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*, volume 28, 2014.

[26] C. Xu, M. Nayyeri, Y.-Y. Chen, and J. Lehmann. Knowledge graph

embeddings in geometric algebras. *arXiv preprint arXiv:2010.00989*, 2020.

[27] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*, 2014.

[28] S. Zhang, Y. Tay, L. Yao, and Q. Liu. Quaternion knowledge graph embeddings. *Advances in neural information processing systems*, 32, 2019.