SWITCH SPACES: LEARNING PRODUCT SPACES WITH SPARSE GATING

Anonymous authors
Paper under double-blind review

ABSTRACT

Aligning the geometric inductive bias well with the underlying structure of data is critical for representation learning. To achieve this goal, we propose *switch spaces*, a data-driven representation learning approach. Switch space is a generalization of product space which is composed of multiple euclidean and non-euclidean (e.g., hyperbolic, spherical) spaces. Given N spaces, our model utilizes a sparse gating mechanism to let each input data point choose $K(K < N)$ spaces and combines them to form a product space. In doing so, ${}_NC_K$ product spaces are generated automatically in a single model and each input data point is processed by one of them with greater specialization, making the spaces switchable. In addition, switch space models are efficient and have a constant computational complexity regardless of the model size. We apply switch spaces to the knowledge graph (KG) completion task and propose *SwisE* which obtains state-of-the-art performance on benchmark KG datasets. We also show that switch space can help achieve promising results on the item recommendation task. Model analysis is conducted to inspect the inner workings of switch space.

1 INTRODUCTION

Many machine learning applications involve representation learning (Bengio et al., 2013). Euclidean and recently non-Euclidean spaces have demonstrated successful applications in modeling data with certain structures in various fields. Euclidean space has been extensively studied in the literature and has been serving as the workhorse for decades in representation learning (Bordes et al., 2013; Mikolov et al., 2013; Hsieh et al., 2017). Non-Euclidean space has gained increasing attention as it excels at modeling various structural patterns in data (Wilson et al., 2014; Nickel & Kiela, 2017; Chami et al., 2020b; Vinh Tran et al., 2020; Meng et al., 2019; Ganea et al., 2018). For example, hyperbolic space, as a continuous version of discrete tree, is more suitable for modeling hierarchies while spherical space is fitting for cyclical structures.

Recently, efforts have been made to combine multiple different types of spaces (i.e., Euclidean, spherical, hyperbolic) into *product spaces* for representation learning. Nevertheless, we argue that aligning the geometry of the input data with its suitable space is a nontrivial task and product spaces lack this capability. To be specific, after the signatures of product spaces are determined (e.g., spherical space $\times$ hyperbolic space), training samples are used to fit the learnable parameters in all these spaces, regardless of the appropriateness (e.g., parameters in both spherical and hyperbolic spaces will be updated even if the input data only aligns with the spherical space). Clearly, data points with different geometric structures should be handled with different component spaces. It is important for component space models to calibrate their internal parameters by focusing on the subset of the training data that fits the manifold and ignoring cases they are not good at modeling. Nevertheless, manually checking the underlying geometry of real-world data is almost impossible due to its intricacy and large size. Therefore, we need a model that can automatically allocate data points to their suitable spaces.

To achieve this goal and better model the structural patterns in data, we propose *switch spaces* which can address the weakness of product spaces. Switch spaces are data-driven and can automatically choose suitable spaces for each input data point based on the input-output relationship. In other words, we let the input data inform the spaces allocation and choose whatever spaces it favours, resulting in better specialization. In switch space, a sparse gating mechanism is introduced for automatic

spaces selection and, as a result, a number of product spaces with different signatures can be formed automatically, allowing for better diversity and expressiveness. Since only a certain number of spaces are active in the calculation for each given data point, the computational cost of switch spaces is linear to the number of active spaces, thus making building larger models possible.

The contributions of this paper are: (a) We propose switch spaces which learn to choose, combine, and switch spaces via a sparse gating mechanism. It effectively aligns different types of geometric spaces with the characteristics of the input data, while remaining computationally efficient; (b) We propose a KG embedding model **SwisE**, short for **Swi**tch **s**pace based KG **E**mbedding, and it obtains state-of-the-art performance on benchmarking KG datasets, outperforming a number of recent published baselines; We also demonstrate that switch spaces can enhance the effectiveness of conventional recommendation models; (c) We conduct model analysis to inspect the inner workings of switch spaces and study the effects of certain modeling choices.

2 Related Work

Non-Euclidean Geometric Representation Learning Non-Euclidean geometric including hyperbolic and spherical representation learning has gained increasing interests in a wide spectrum of applications (Nickel & Kiela, 2017; Vinh Tran et al., 2020; Tay et al., 2018; Balazevic et al., 2019a; Chami et al., 2019; Liu et al., 2019; Chami et al., 2020a). Hyperbolic space is reminiscent of continuous version of trees and excels in modeling hierarchical structures, while spherical space is a more suitable choice for directional similarity modeling and cyclical-structured data. We can find applications of spherical spaces in text embeddings (Meng et al., 2019), texture mapping (Wilson et al., 2014), time-warping functions embedding (Wilson et al., 2014). As indicated, different space has its own specialities and the choice of spaces varies based on the characteristics of data. To combine the best of different worlds, product spaces with mixed curvatures are proposed (Gu et al., 2019; Skopek et al., 2020; Bachmann et al., 2020). The component spaces in product spaces are selected from Euclidean space, hyperbolic space, and spherical space. Each component of product space has a constant curvature while the resulting mixed space has non-constant curvature, which makes it possible to capture a wide range of patterns. Empirically, product spaces demonstrate its efficacy in graph reconstruction, wording embedding with low dimensions (Gu et al., 2019), node classification in graphs (Bachmann et al., 2020), and image reconstruction (Skopek et al., 2020).

Mixture of Experts Another related line of work is mixture-of-experts (MOE) (Jacobs et al., 1991; Jordan & Jacobs, 1994). MOE is established based on the divide-and-conquer principle. It divides problem into homogeneous regions and an expert is responsible for each region. The final prediction is arrived based on the cooperation between experts via a gating network. As a controller, the gating networks control the contribution of each experts via a probabilistic gating function. MOE has been extensively studied in a range of applications such as language modeling and machine translation (Shazeer et al., 2017; Fedus et al., 2021), multi-task learning (Ma et al., 2018), etc. Various MOE architectures have also been proposed in the past decades including hierarchical structure (Yao et al., 2009), sequential experts (Aljundi et al., 2017), deep MOE (Eigen et al., 2013), sparsely-gated MOE (Shazeer et al., 2017), etc.

Product space model is designed for data with mixed patterns. Yet, it is unclear how to construct a product space model that makes data (i.e, triplets in KG) with certain patterns handled by its suitable space. To enhance the alignment between data and its suitable geometric space and calibrate each component model, the proposed switch spaces take the advantages of the sparse gating mechanism of MOE (Shazeer et al., 2017). That is, given a pool of spaces, a sparse gating mechanism is integrated to activate a few spaces for each given data point, assigning data to its suitable spaces without manual inspection of the inner geometry of input data. Switch space can be viewed as a generalization of product space but offers better specialization, higher degree of freedom, and superior expressiveness.

3 Preliminaries: The Geometry of Product Spaces

Stereographic Projection model In general, there are three types of constant curvature spaces with respect to the sign of the curvature. Common realizations are Euclidean space $\mathbb{E}$ (flat), hypersphere $\mathbb{S}$ (positively curved) and hyperboloid $\mathbb{H}$ (negatively curved). For the latter two, we prefer their

stereographic projection model: projected sphere $\mathbb{D}$ and Poincare ball $\mathbb{P}$. These models are easier to optimize, avoiding the problem of non-convergence of norm of points with curvature close to 0, and the projection is conformal, i.e., does not affect the angles between points (Nickel & Kiela, 2017; Skopek et al., 2020).

An alternative to vector space in non-Euclidean geometry is gyrovector space (Ungar, 1991), which defines operations such as vector addition and multiplication. For $\mathbb{D}$ and $\mathbb{P}$ (jointly denoted as $\mathcal{M}_c$, where c denotes curvature), the addition between two points $\mathbf{x}, \mathbf{y} \in \mathcal{M}_c$, also known as Möbius addition $\oplus_c$ (for both signs of c), is defined as:

$$\mathbf{x} \oplus_c \mathbf{y} = \frac{(1 - 2c\langle \mathbf{x}, \mathbf{y} \rangle - c\|\mathbf{y}\|_2^2)\mathbf{x} + (1 + c\|\mathbf{x}\|_2^2)\mathbf{y}}{1 - 2c\langle \mathbf{x}, \mathbf{y} \rangle + c^2\|\mathbf{x}\|_2^2\|\mathbf{y}\|_2^2}, \tag{1}$$

Where $\langle , \rangle$ is Euclidean inner product. The distance between points in the gyrovector space is defined as:

$$d_{\mathcal{M}_c}(\mathbf{x}, \mathbf{y}) = \frac{2}{\sqrt{|c|}} \tan_c^{-1}(\sqrt{|c|}\| - \mathbf{x} \oplus_c \mathbf{y}\|_2), \tag{2}$$

where $\tan_c$ stands for $\tan$ if $c > 0$ and $\tanh$ if $c < 0$. In both spaces, we have Euclidean geometry when $c \to 0$. It is easy to prove that: $d_{\mathcal{M}_c}(\mathbf{x}, \mathbf{y}) \xrightarrow{c \to 0} 2\|\mathbf{x} - \mathbf{y}\|_2$, which means the gyrospace distance converges to Euclidean distance when limiting c to zero.

Let $\mathcal{T}_{\mathbf{x}}\mathcal{M}_c$ be the tangent space to the point $\mathbf{x} \in \mathcal{M}_c$. Mapping between (Euclidean) tangent space and hyperbolic/spherical space is performed with exponential map: $\mathcal{T}_{\mathbf{x}}\mathcal{M}_c \to \mathcal{M}_c$ and logarithmic map: $\mathcal{M}_c \to \mathcal{T}_{\mathbf{x}}\mathcal{M}_c$, which are defined as:

$$\begin{aligned} \log_{\mathbf{x}}^c(\mathbf{y}) &= \frac{2}{\sqrt{|c|}\lambda_{\mathbf{x}}^c} \tan_c^{-1}(\sqrt{|c|}\| - \mathbf{x} \oplus_c \mathbf{y}\|_2) \frac{-\mathbf{x} \oplus_c \mathbf{y}}{\| - \mathbf{x} \oplus_c \mathbf{y}\|_2}, \\ \exp_{\mathbf{x}}^c(\mathbf{v}) &= \mathbf{x} \oplus_c (\tan_c(\sqrt{|c|}\frac{\lambda_{\mathbf{x}}^c\|\mathbf{v}\|_2}{2})\frac{\mathbf{v}}{\sqrt{|c|}\|\mathbf{v}\|_2}), \end{aligned} \tag{3}$$

where $\lambda_{\mathbf{x}}^c$ is a conformal factor, defined as $\lambda_{\mathbf{x}}^c = 2/(1 + c\|\mathbf{x}\|_2^2)$, which is used to transform the metric tensors between (Euclidean) tangent space and non-Euclidean space.

Product Spaces The product space is defined as the Cartesian product of multiple spaces with varying dimensionality and curvature. Let $\mathcal{P}$ denote a product space composed by N independent component spaces $\mathcal{M}^{(1)}$, $\mathcal{M}^{(2)}$,...,$\mathcal{M}^{(N)}$. The mixed space $\mathcal{P}$ has the form: $\mathcal{P} = \times_{i=1}^{N} \mathcal{M}^{(i)} = \mathcal{M}^{(1)} \times \mathcal{M}^{(2)} \times ... \times \mathcal{M}^{(N)}$. The product space $\mathcal{P}$ is also equipped with distance functions. The squared distance between points $\mathbf{x}, \mathbf{y} \in \mathcal{P}$ is defined as: $d_{\mathcal{P}}^2(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{N} d_{\mathcal{M}^{(i)}}^2(\mathbf{x}_{\mathcal{M}^{(i)}}, \mathbf{y}_{\mathcal{M}^{(i)}})$, where $\mathbf{x}_{\mathcal{M}^{(i)}}$ and $\mathbf{y}_{\mathcal{M}^{(i)}}$ denote the corresponding vectors on the component space $\mathcal{M}^{(i)}$.

Other operations such as exponential map and logarithmic map are element-wise, which means that we can decompose the points into component spaces and apply operations on each component space and then compose them back (e.g., concatenation) to the product space. The signature (i.e., parametrization) of a product space refers to the types of space, the dimensionality, and curvature of each space.

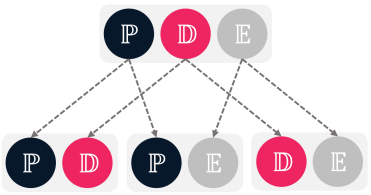

Figure 1: Example of a switch space where $N = 3, K = 2$ and the initial three spaces are Poincare, Spherical, and Euclidean spaces. It generates ${}_3C_2 = 3$ resulting product spaces including $\mathbb{P} \times \mathbb{D}$ (or written as $\mathbb{PD}$), $\mathbb{P} \times \mathbb{E}$, and $\mathbb{D} \times \mathbb{E}$.

4 The Proposed Switch Spaces

4.1 The Switch Space Framework

Suppose we have N spaces of different types and $\mathcal{M}_c^{(i)} \in \{\mathbb{E}, \mathbb{D}, \mathbb{P}\}, i = \{1, ..., N\}$. For simplicity, we assume all spaces have the same dimensionality, b. The goal of switch spaces is to select $K (1 \leqslant K < N)$ spaces out from the given N spaces for each incoming triple (h, r, t) and the

selected K spaces will form a product space. As such, each triple will be handled by different product space. Theoretically, there will be ${}_NC_K = \frac{N!}{(N-K)!K!}$ possible combinations of product spaces, allowing a greater degree of freedom. Figure 1 illustrates a simple example of a switch space which can generate three product spaces automatically.

The proposed switch spaces usually consist of an embedding layer, a sparse gating network, and a space combination component. In what follows, we will detail the switch space framework using two real-world applications, knowledge graph completion and item recommendation.

4.2 Switch Spaces for Knowledge Graph Completion

Knowledge graph has emerged as an effective way to integrate disparate data sources and model underlying relationships. Encoding the entities and relations of KGs into low-dimensional vector spaces is essential to downstream applications such as missing facts completion, question answering, information extraction, and semantic reasoning. The learned embeddings are expected to preserve the key information and relational/structural patterns of KGs.

Given a knowledge graph $\mathcal{G}$ with a set of entities $\mathcal{E}$ and a set of relations $\mathcal{R}$. Each triple, abbreviated as (h, r, t), in $\mathcal{G}$ is composed by two entities (i.e., head entity $h \in \mathcal{E}$ and tail entity $t \in \mathcal{E}$), and the relationship $r \in \mathcal{R}$ between them.

Embedding Layer In the i^{th} space, entities h, t are represented by vectors $\mathbf{e}_h^{(i)}, \mathbf{e}_t^{(i)} \in \mathbb{R}^b$ and relation r is represented by two translation vectors $\boldsymbol{\alpha}_r^{(i)}, \boldsymbol{\beta}_r^{(i)} \in \mathbb{R}^b$ and a rotation vector $\boldsymbol{\gamma}_r^{(i)} \in \mathbb{R}^b$. Also, each head (tail) entity is associated with a bias term $b_h(b_t) \in \mathbb{R}$. We initialize all the embedding vectors in tangent space and exponential map will be applied to recover the hyperbolic/spherical parameters when necessary. There are two benefits of doing so: On the one hand, the embedding layer can be easily utilized by the sparse gating network defined in Euclidean space. On the other, standard Euclidean optimization techniques can be applied thanks to the convenient exponential and logarithmic maps. The overall framework can be optimised in a uniform fashion, avoiding the cumbersome Riemannian optimization (Bonnabel, 2013).

Sparse Gating Network The sparse gating network is a differentiable data-driven tool used to select the most suitable K spaces for each data point. The input for the sparse gating network is constructed from the head entity embedding (tail entity embedding is not used as it shares the same embedding matrix with the head entity) and the relation embeddings (the rotation vector is not used as it will be normalized for rotation). The input is defined as:

$$\mathbf{x} = [\mathbf{e}_h^{(1)}, ..., \mathbf{e}_h^{(N)}, \boldsymbol{\alpha}_r^{(1)}, ..., \boldsymbol{\alpha}_r^{(N)}, \boldsymbol{\beta}_r^{(1)}, ..., \boldsymbol{\beta}_r^{(N)}]^\top, \tag{4}$$

where $\mathbf{x}$ is a two-dimensional matrix of shape $3N \times b$. Then 2-D convolutions with linear transformations are applied over $\mathbf{x}$ to transform it into a vector of dimensionality N. Convolutional operations can deal with large dimensionality b without introducing too much additional free parameters. In formal, the last hidden layer of the gating network has the following form:

$$f(\mathbf{x}) = f_1(\mathbf{x}) + \text{randn}() \cdot \ln(1 + \exp(f_2(\mathbf{x}))), \tag{5}$$

where f_1 and f_2 represent CNN layers and $f_*(\mathbf{x}) \in \mathrm{R}^N$; function "randn()" creates samples from the standard normal distribution; the second term is a tunable Gaussian noise (only for training) that is added to improve the load balancing, i.e., balance the number of samples accepted by each space.

To impose sparsity, we employ a TopK function to obtain the largest K elements from $f(\mathbf{x})$:

$$f(\mathbf{x}) \leftarrow \text{TopK}(f(\mathbf{x})), \tag{6}$$

where the TopK function returns the original value if the element is in the top K list, otherwise it returns $-\infty$. We need N gates $g_1, g_2, .., g_N$ to generate probabilities to control the activeness of each space. Simply, a softmax gating network is employed. For $-\infty$, it will output a zero gate value.

$$g_i(\mathbf{x}) = \frac{\exp(f(\mathbf{x})_i)}{\sum_{j=1}^{N} \exp(f(\mathbf{x})_j)}, i = \{1, ..., N\}. \tag{7}$$

Only spaces with nonzero gate values will be active while other spaces are idle and not computed. This approach provides the possibility to enlarge the model size without incurring much computational cost as long as the number of active spaces K is fixed.

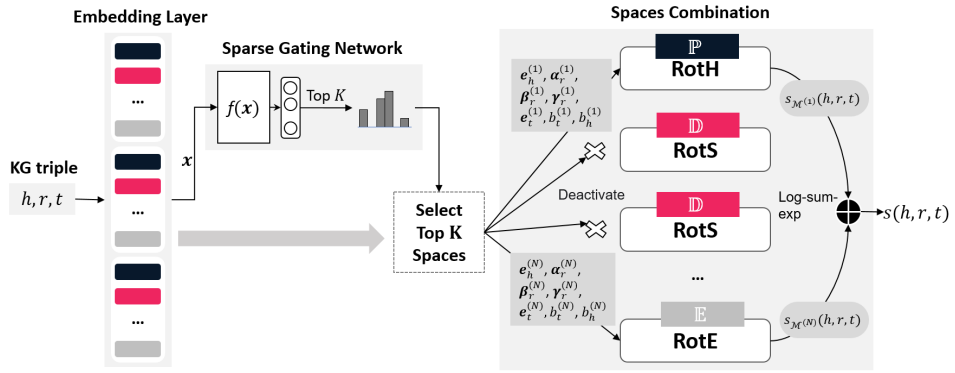

Figure 2: Architecture of the proposed SwisE. Only K out of N spaces are active for each KG triple.

RotE, RotH, RotS, and SwisE The base models RotE and RotH proposed by (Chami et al., 2020b) are models of the same form but defined in Euclidean and Poincare spaces, respectively. We contribute two new models based on them, RotS and SwisE. RotS has the same form as RotE/RotH but is defined in the spherical space. SwisE is a model that operates in switch space and it uses RotE, RotH, and RotS as the component model. The architecture of SwisE is shown in Figure 2.

To define RotS, let the i^{th} space $\mathcal{M}^{(i)}$ be spherical. In RotS, the head entity is translated twice via Möbius addition in the spherical space and rotated once. In formal, the head entity is processed as follows.

$$Q^{(i)}(h, r) = \text{ROTATE}(\exp_{\mathbf{0}}^{c}(\mathbf{e}_h^{(i)}) \oplus_c \exp_{\mathbf{0}}^{c}(\boldsymbol{\alpha}_r^{(i)}), \boldsymbol{\gamma}_r^{(i)}) \oplus_c \exp_{\mathbf{0}}^{c}(\boldsymbol{\beta}_r^{(i)}), \tag{8}$$

where $c > 0$ and $\exp_{\mathbf{0}}^{c}$ is the exponential map over origin. ROTATE is a rotation function (same as RotE/RotH) and $\boldsymbol{\gamma}_r^{(i)}$ is the rotation matrix. The transformed head entity is then compared with the tail entities using squared spherical distance. The final scoring function is as follows:

$$s_{\mathcal{M}^{(i)}}(h, r, t) = -d^2_{\mathcal{M}^{(i)}}(Q^{(i)}(h, r), \exp_{\mathbf{0}}^{c}(\mathbf{e}_t^{(i)})) + b_h^{(i)} + b_t^{(i)}, \tag{9}$$

Same as RotH, we make the curvature relation specific and trainable. That is, each relation has a curvature parameter and they are trained simultaneously with the model. In the implementation, we use softplus to convert curvatures to positive values and then enforce it to be positive or negative based on the type of space we use.

In SwisE, we have N models in total selected from RotE ($c = 0$), RotH($c < 0$), and RotS ($c > 0$). Let $s_{\mathcal{M}^{(i)}}(h, r, t)$ denote the output of the i^{th} model. Since only K out of N models will be active, we set the outputs of the inactive models to 0. Similar to the definition of squared distance on the product space, the scoring function of SwisE also decomposes and we define it as:

$$s(h, r, t) = \log(\sum_{i=1}^{N} g_i(\mathbf{x}) \exp(s_{\mathcal{M}^{(i)}}(h, r, t))), \tag{10}$$

where a log-sum-exp technique is adopted to improve the numerical stability and to avoid problems such as underflow and overflow. Note that multiplying by the gating probability is optional in our framework. If $s_{\mathcal{M}^{(i)}}$ is calculated with squared distance and $K = N$, we can recover the squared distance function of product space by removing $g_i(\mathbf{x})$ and the log-sum-exp technique.

Objective Function of SwisE We adopt the following cross-entropy loss as the objective function of SwisE.

$$\mathcal{L} = \sum_{(h,r,t)\in\Omega} \log(1 + \exp(-Y_{(h,r,t)} s(h, r, t))), \tag{11}$$

where $Y_{(h,r,t)} \in \{1, -1\}$ is a binary label indicating whether a triple is factual (1) or not (-1). Ω represents the training collection including both positive and negative triples. Also, we add an additional regularization for load balancing (discussed in Appendix A). The whole model is optimized end-to-end in tangent space with the Euclidean Adam optimizer (Kingma & Ba, 2014).

$\mathcal{M}$	Model	WN18RR				FB15K-237			
		MRR	HR@1	HR@3	HR@10	MRR	HR@1	HR@3	HR@10
$\mathbb{R}$	TransE	0.223	0.013	0.401	0.529	0.332	0.233	0.372	0.531
	BoxE	0.451	0.400	0.472	0.541	0.337	0.238	0.374	0.538
	DistMult	0.430	0.390	0.440	0.490	0.241	0.155	0.263	0.419
	ConvE	0.430	0.400	0.440	0.520	0.325	0.237	0.356	0.501
	TuckER	0.470	0.443	0.482	0.526	0.358	0.266	0.394	0.544
	RotE	0.494	0.446	0.512	0.585	0.346	0.251	0.381	0.538
$\mathbb{C}$	ComplEx-N3	0.480	0.435	0.495	0.572	0.357	0.264	0.392	0.547
	RotatE	0.476	0.428	0.492	0.571	0.338	0.241	0.375	0.533
$\mathbb{Q}$	QuatE	0.488	0.438	0.508	0.582	<u>0.366</u>	0.271	<u>0.401</u>	0.556
$\mathbb{S}$	MuRS	0.454	0.432	0.482	0.550	0.338	0.249	0.373	0.525
$\mathbb{P}$	MurP	0.481	0.440	0.495	0.566	0.335	0.243	0.367	0.518
	RotH	<u>0.496</u>	<u>0.449</u>	<u>0.514</u>	<u>0.586</u>	0.344	0.246	0.380	0.535
♣	M^{2}GNN	0.485	0.444	0.498	0.572	0.362	<u>0.275</u>	0.398	<u>0.565</u>
★	**SwisE**	**0.526** $\pm.002$	**0.479** $\pm.004$	**0.549** $\pm.004$	**0.611** $\pm.003$	**0.530** $\pm.006$	**0.500** $\pm.008$	**0.545** $\pm.006$	**0.590** $\pm.005$

Table 1: The performance of switch space on WN18RR is achieved with model $(\mathbb{D}^{100})^4\mathbb{E}^{100}$(K=2), and that on FB15K-237 is with $(\mathbb{P}^{100})^3(\mathbb{D}^{100})^2(K=4)$. $\mathbb{R}$:real coordinate space; $\mathbb{C}$:complex number space; $\mathbb{Q}$:quaternion space. ♣ denotes mixed curvature product space. ★ denotes switch space.

4.3 Switch Spaces for Recommender Systems

Recommender system is an important part in modern e-commerce platforms. It can improve customer experience via reducing information overload and increase revenue for companies. Learning representations that reflect users' preferences and items characteristics have been a central theme in the field. We simplify the model details since it follows the same procedure as that of SwisE.

Given a collection of users and items and the interactions observed between them, the goal is to generate personalized items recommendations for each user. We are interested in metric learning based collaborative filtering (Hsieh et al., 2017; Vinh Tran et al., 2020). Taken a user vector $\mathbf{u}_u^{(i)} \in \mathbb{R}^b$ and an item vector $\mathbf{v}_v^{(i)} \in \mathbb{R}^b$ as input. The the preference of user u towards item v in each space is measured by the squared distance between them:

$$s_{\mathcal{M}^{(i)}}(u,v) = -d^2_{\mathcal{M}^{(i)}}(\exp^c_{\mathbf{0}}(\mathbf{u}_u), \exp^c_{\mathbf{0}}(\mathbf{v}_v)). \tag{12}$$

The input of the gating network is the concatenation of the user and item embeddings. Specifically, a vector of size R^{2Nb} is used as the input and a linear layer is used in the gating network. We use $s(u,v)$ to denote the final prediction score and optimize the model with a max-margin hinge loss:

$$\mathcal{L} = \sum_{(u,v_1,v_2)\in\Psi} \max(0, s(u,v_1) + m - s(u,v_2)), \tag{13}$$

where Ψ is the collection of training samples; v_1 is the item that u liked and v_2 represents a negative (unobserved) item for the user. $\max(0,x)$ is also known as the *ReLU* function. m is a margin value.

5 Performance Evaluation

We evaluate switch spaces on real-world datasets. The statistics of datasets and hyper-parameter settings are reported in Appendix C.

5.1 Results on Knowledge graph completion

Baselines We compare SwisE with a number of baselines, including Euclidean methods TransE (Bordes et al., 2013), DistMult (Yang et al., 2015), ConvE (Dettmers et al., 2018), TuckER (Balazevic et al., 2019b), RotE (Chami et al., 2020b), BoxE (Abboud et al., 2020); complex number based methods ComplEx-N3, (Lacroix et al., 2018) and RotatE (Sun et al., 2019); quaternion model QuatE (Zhang et al., 2019); spherical models MuRS (Wang et al., 2021); hyperbolic methods MurP (Balazevic et al., 2019a), RotH (Chami et al., 2020b); and product space model M^{2}GNN (Wang et al., 2021).

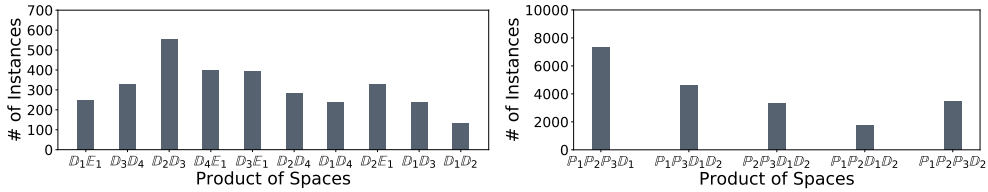

Figure 3: Distribution of the learned spaces on the test data of WN18RR (left figure, $(\mathbb{D}^{100})^4\mathbb{E}^{100}(K=2)$) and FB15K-237 (right figure, $(\mathbb{P}^{100})^3(\mathbb{D}^{100})^2(K=4)$) using switch space.

Metrics The performance of different models are evaluated using two standard metrics including mean reciprocal rank (MRR) and hit rate (HR) with cut-off values $\{1, 3, 10\}$. The standard filtered setting (Bordes et al., 2013) is adopted.

Comparison with the State-of-the-arts Table 1 compares SwisE with baselines. We make the following observations. Firstly, SwisE achieves the new state-of-the-art results across all metrics on both datasets. SwisE outperforms Euclidean, hyperbolic, spherical, complex-valued, product spaces approaches, and obtains a clear performance gain over the second best model. Secondly, the second best performers scatter among QuatE, RotH, and M^{2}GNN. On the WN18RR dataset, RotH achieves the second best results. On the FB15K-237 dataset, QuatE and M^{2}GNN achieve on par performances. Thirdly, the improvement of SwisE over the second best model on FB15K-237 is larger than that on WN18RR. One possible explanation is that FB15K-237 has a more diverse set of relations and a higher structure heterogeneity (Balazevic et al., 2019a).

$\mathcal{M}$	Model	WN18RR		FB15K-237	
		MRR	HR@3	MRR	HR@3
Single	$\mathbb{D}^{500}$	0.492	0.514	0.293	0.322
Product	$(\mathbb{P}^{100})^3(\mathbb{D}^{100})^2$	0.484	0.498	0.311	0.344
	$(\mathbb{D}^{100})^4\mathbb{E}^{100}$	0.479	0.497	0.312	0.344
	$(\mathbb{P}^{100})^4\mathbb{D}^{100}$	0.468	0.488	0.321	0.356
	$(\mathbb{P}^{100})^2(\mathbb{D}^{100})^2\mathbb{E}^{100}$	0.479	0.496	0.308	0.342
Switch	$(\mathbb{P}^{100})^3(\mathbb{D}^{100})^2$	0.500	0.521	0.530	0.545
	$(\mathbb{D}^{100})^4\mathbb{E}^{100}$	0.526	0.549	0.525	0.531
	$(\mathbb{P}^{100})^4\mathbb{D}^{100}$	0.504	0.526	0.515	0.527
	$(\mathbb{P}^{100})^2(\mathbb{D}^{100})^2\mathbb{E}^{100}$	0.522	0.546	0.526	0.535

Table 2: Performance comparison with product space models (a few signatures are reported due to length constraints), K is set to 4 on WN18RR and 2 for FB15K-237.

Comparison with Product Space In the first place, we compare SwisE with the product space model M^{2}GNN (its signature is $\mathbb{P}^{200}\mathbb{D}^{200}\mathbb{E}^{200}$). As shown in Table 1, SwisE outperforms M^{2}GNN on all metrics. Moreover, compared with the single space model, QuatE and RotH, M^{2}GNN does not show much advantages. This observation confirms that using pure product space will not offer much merit for data representation on KG.

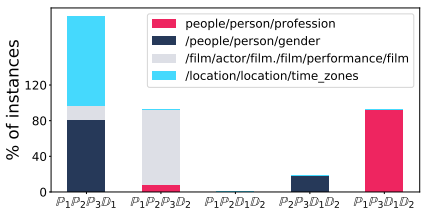

Figure 4: Distribution of the learned spaces of four relations on FB15K-237. Counts of instances are normalized.

Then, we compare SwisE with product space models with varying signatures in Table 2. We observe that the switch space model constantly outperforms pure product spaces with the same prior signatures. It is worth noting that SwisE requires merely two/four active spaces to outperform product space models with five active spaces. Additionally, we find that spherical space dominant models can also obtain very competitive performances on both datasets. our assumption is that the spherical model can not only model cyclical structures, but can also capture hyperbolic structures to some extent. We provide a proof using graph distance ratio in Appendix B.

5.2 Results on Recommender Systems

Baselines We compare our method with a number of baselines including Bayesian personalized ranking based matrix factorization (BPRMF) (Rendle et al., 2009), Euclidean, hyperbolic, spherical, and product space models. $\mathbb{E}^n$ corresponds to CML (Hsieh et al., 2017), $\mathbb{P}^n$ is equivalent to HyperML (Vinh Tran et al., 2020).

Space	Model	MovieLens 100K			MovieLens 1M		
		MAP	P@10	R@10	Map	P@10	R@10
Single	BPRMF	.211±.003	.254±.001	.176±.004	.120±.001	.185±.001	.078±.001
	$\mathbb{E}^{100}$(CML)	.195±.001	.233±.003	.165±.003	.138±.002	.208±.002	.092±.003
	$\mathbb{P}^{100}$(HyperML)	.157±.001	.194±.003	.136±.004	.113±.001	.167±.002	.075±.002
	$\mathbb{D}^{100}$	.161±.001	.193±.002	.137±.001	.116±.001	.170±.004	.077±.001
Product	$(\mathbb{E}^{20})^5$	.199±.002	.240±.001	.168±.003	.150±.001	.221±.002	.100±.001
	$(\mathbb{P}^{20})^5$	.169±.003	.210±.004	.146±.003	.128±.003	.189±.002	.083±.001
	$(\mathbb{D}^{20})^5$	.177±.003	.212±.002	.149±.005	.131±.001	.188±.001	.085±.004
	$(\mathbb{D}^{20})^3(\mathbb{P}^{20})^2$	.173±.001	.209±.001	.151±.001	.133±.001	.193±.001	.085±.001
	$(\mathbb{P}^{20})^2(\mathbb{D}^{20})^2\mathbb{E}^{20}$	.176±.001	.208±.002	.149±.003	.134±.002	.192±.003	.088±.002
Switch (K=4)	$(\mathbb{E}^{20})^5$	**.218**±.003	**.267**±.003	**.188**±.002	.158±.001	.236±.001	.105±.002
	$(\mathbb{P}^{20})^5$	.191±.002	.233±.005	.162±.004	.142±.003	.213±.005	.094±.005
	$(\mathbb{D}^{20})^5$	.194±.002	.233±.005	.162±.002	.172±.003	**.253**±.003	**.116**±.002
	$(\mathbb{D}^{20})^3(\mathbb{P}^{20})^2$	.203±.003	.246±.002	.174±.003	**.174**±.001	.252±.002	**.116**±.001
	$(\mathbb{P}^{20})^2(\mathbb{D}^{20})^2\mathbb{E}^{20}$	.201±.002	.245±.004	.171±.001	.172±.001	.249±.002	.114±.002

Table 3: Recommendation performance on two benchmark datasets.

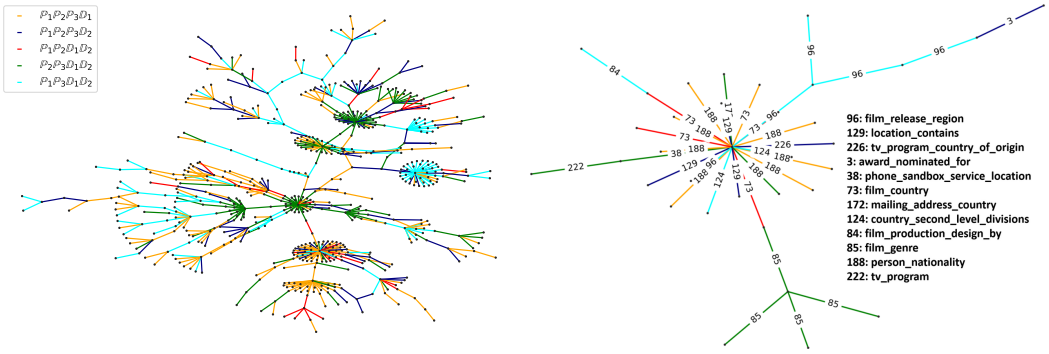

Figure 5: Visualization of the spaces distribution on subsets of the FB15K-237.

Metrics We measure the performance based on the widely adopted metrics in recommender systems: mean average precision (MAP), precision ($P@5$ and $P@10$), and recall ($R@5$ and $R@10$).

Results Table 3 summarizes the performances. The merits of switch spaces can be further observed from the comparison. On both datasets, switch space based models achieve the best performances, outperforming their pure product space counterparts with the same signature. Another clear pattern is that product space models usually perform better than single space models. Specifically, on MovieLens 100K, Euclidean space is the most effective. While, on MovieLens 1M, spherical space dominated models outperform models with other space combinations. This observation is consistent with the global average curvature we estimated (Appendix C). Interestingly, we find that the matrix factorization approach (BPRMF) remains competitive on the two datasets and can sometimes achieve better results than other single space models.

6 Model Analysis

We conduct model analysis using SwisE and dataset FB15K-237.

6.1 Distribution of the Learned Spaces

Figure 3 presents the distribution of the learned space combinations. It is obvious that KG triples are not randomly distributed. Instead, certain product spaces are preferred in each case. For example, the product space $\mathbb{D}_2\mathbb{D}_3$ is the most popular on WN18RR. While on FB15K-237, the product space $\mathbb{P}_1\mathbb{P}_2\mathbb{P}_3\mathbb{D}_1$ is preferable. To show more fine-grained examples, we randomly select four relations from FB15K-237 and visualize the distribution of product spaces in Figure 4. We find that the relations have their own desirable product spaces as well. For example, relation "*/location/location/time_zones*" prefers space $\mathbb{P}_1\mathbb{P}_2\mathbb{P}_3\mathbb{D}_1$ but relation "*people/person/profession*" prefers space $\mathbb{P}_1\mathbb{P}_3\mathbb{D}_1\mathbb{D}_2$. This reconfirms the specification capability of SwisE.

Furthermore, we randomly extract two connected graph components from the test set of FB15-237 and visualize them in Figure 5. Each triple can be represented with an edge and its two end nodes. We render edges with different colors based on the type of product spaces that are used to process the corresponding triples. From the left graph, we can see some clear clusters and find that each cluster usually has a dominant color, which suggests that neighborhood triples are more likely to be processed with the same product space. From the right graph, we find that different relations are processed by different spaces even if they are located near to each other. This observation indicates that the type of relations plays a critical role in determining which spaces to use. More examples are presented in Appendix C.

6.2 Effects of Total Spaces Number

Figure 6 (left) reports the test MRR and inference time varied across the different N values with K=2 using switch space model $(\mathbb{P}^{100})^N$ on WN18RR. We observe that: (a) the model performance does not benefit from increasing model size on this dataset; (b) the inference time remains nearly constant when we increase N, confirming its efficiency. This property is important for tasks that prefer a larger model size.

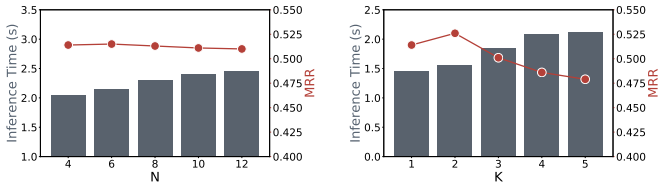

Figure 6: Effect of N (left) and K (right) on WN18RR.

6.3 Effects of Active Spaces Number

We show the effects of the active space number K using model $(\mathbb{D}^{100})^4\mathbb{E}^{100}$ on WN18RR in Figure 6 (right). We find that, compared with N, K has a higher impact on the model performance and inference time. On WN18RR, a small K (e.g., 2) value is enough to obtain the optimal performance. Increasing K will generally degrade the performance.

6.4 Effects of Embedding Dimension

We study the impact of the embedding dimension on WN18RR and report the results in Figure 7 (Upper). In general, the embedding dimension has a high impact on the model accuracy. We also find that SwisE performs worse than RotH when the embedding dimension is small. This might because that SwisE only has two active spaces. For example, the active dimension is only 40 for SwisE when the overall dimension is 100, while RotH has 100 active dimensions in this case. As such, we suggest that it is better to keep the overall active dimensionality in a reasonable size.

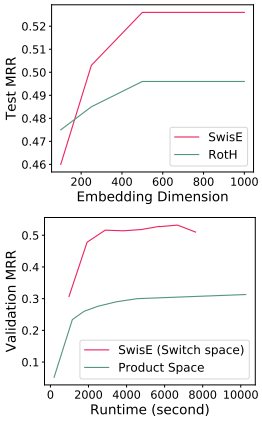

Figure 7: Effects of embedding dimensionality (Upper) and model Convergence (Lower).

6.5 Convergence

The convergence of SwisE and pure product space model on FB15K-237 is shown in in Figure 7(Lower). Clearly, SwisE will not incur additional computational cost and converges as fast as the product space model.

7 Conclusion

In this paper, we propose a novel representation learning framework, switch space. Switch space makes use of a sparse gating network to enhance the alignment between data and geometric spaces and ensure the given data to be handled by suitable spaces, allowing a greater extent of specification in the representation while remaining efficient. With switch spaces, we manage to obtain a state-of-the-art performance on the knowledge graph completion task and promising results on the item recommendation task. We believe that switch spaces hold promise for more expressive and generalizable representation learning and can be applied to many other areas.

REFERENCES

Ralph Abboud, Ismail Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. Boxe: A box embedding model for knowledge base completion. *Advances in Neural Information Processing Systems*, 33, 2020.

Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3366–3375, 2017.

Gregor Bachmann, Gary Bécigneul, and Octavian Ganea. Constant curvature graph convolutional networks. In *International Conference on Machine Learning*, pp. 486–496. PMLR, 2020.

Ivana Balazevic, Carl Allen, and Timothy Hospedales. Multi-relational poincaré graph embeddings. In *Advances in Neural Information Processing Systems*, pp. 4463–4473, 2019a.

Ivana Balazevic, Carl Allen, and Timothy Hospedales. Tucker: Tensor factorization for knowledge graph completion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 5188–5197, 2019b.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

Silvere Bonnabel. Stochastic gradient descent on riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229, 2013.

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*, 26:2787–2795, 2013.

Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. Hyperbolic graph convolutional neural networks. In *Advances in neural information processing systems*, pp. 4868–4879, 2019.

Ines Chami, Albert Gu, Vaggos Chatziafratis, and Christopher Ré. From trees to continuous embeddings and back: Hyperbolic hierarchical clustering. *Advances in Neural Information Processing Systems*, 2020a.

Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. Low-dimensional hyperbolic knowledge graph embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 6901–6914, Online, July 2020b. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.617. URL https://www.aclweb.org/anthology/2020.acl-main.617.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. Learning factored representations in a deep mixture of experts. *arXiv preprint arXiv:1312.4314*, 2013.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*, 2021.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. *arXiv preprint arXiv:1805.09112*, 2018.

Albert Gu, Frederic Sala, Beliz Gunel, and Christopher Ré. Learning mixed-curvature representations in product spaces. In *International Conference on Learning Representations*, 2019.

F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge Belongie, and Deborah Estrin. Collaborative metric learning. In *Proceedings of the 26th international conference on world wide web*, pp. 193–201, 2017.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Timothee Lacroix, Nicolas Usunier, and Guillaume Obozinski. Canonical tensor decomposition for knowledge base completion. In *International Conference on Machine Learning*, pp. 2863–2872, 2018.

Qi Liu, Maximilian Nickel, and Douwe Kiela. Hyperbolic graph neural networks. In *Advances in Neural Information Processing Systems*, pp. 8230–8241, 2019.

Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1930–1939, 2018.

Yu Meng, Jiaxin Huang, Guangyuan Wang, Chao Zhang, Honglei Zhuang, Lance Kaplan, and Jiawei Han. Spherical text embedding. In *Advances in Neural Information Processing Systems*, pp. 8208–8217, 2019.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Icml*, 2011.

Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In *Advances in neural information processing systems*, pp. 6338–6347, 2017.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pp. 452–461, 2009.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pp. 593–607. Springer, 2018.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Ondrej Skopek, Octavian-Eugen Ganea, and Gary Bécigneul. Mixed-curvature variational autoencoders. In *International Conference on Learning Representations*, 2020.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*, 2019.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. Hyperbolic representation learning for fast and efficient neural question answering. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pp. 583–591, 2018.

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

Abraham A Ungar. Thomas precession and its associated grouplike structure. *American Journal of Physics*, 59(9):824–834, 1991.

Lucas Vinh Tran, Yi Tay, Shuai Zhang, Gao Cong, and Xiaoli Li. Hyperml: A boosting metric learning approach in hyperbolic space for recommender systems. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pp. 609–617, 2020.

Shen Wang, Xiaokai Wei, Cícero Nogueira dos Santos, Zhiguo Wang, Ramesh Nallapati, Andrew Arnold, Bing Xiang, S Yu Philip, and Isabel F Cruz. Mixed-curvature multi-relational graph neural network for knowledge graph completion. In *The Web Conference*, 2021.

Richard C Wilson, Edwin R Hancock, Elżbieta Pekalska, and Robert PW Duin. Spherical and hyperbolic embeddings of data. *IEEE transactions on pattern analysis and machine intelligence*, 36(11):2255–2269, 2014.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *ICLR*, 2015.

Bangpeng Yao, Dirk Walther, Diane Beck, and Li Fei-Fei. Hierarchical mixture of classification experts uncovers interactions between brain regions. *Advances in Neural Information Processing Systems*, 22:2178–2186, 2009.

Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. Quaternion knowledge graph embeddings. In *Advances in Neural Information Processing Systems*, pp. 2735–2745, 2019.