A Load Balancing Regularization

To avoid some spaces to be overselected and balance the number of samples accepted by each selected space. The following load balancing regularization technique (Shazeer et al., 2017) is employed.

The first additional loss is used to encourage all spaces to have equal importance. For a batch of X inputs,

$$\ell_1 = \text{CV}(\sum_{x \in X} g(\mathbf{x}))^2 \tag{14}$$

where $\text{CV}(v) = \frac{variance(v)}{mean(v)}$ is the coefficient of variation.

The second additional loss ensures all active spaces to have a roughly equal number of training examples.

$$\ell_2 = \text{CV}(Load(X))^2 \tag{15}$$

$$Load(X)_i = \sum_{x \in X} \Phi(\frac{f_1(\mathbf{x}) - kthexcluding(f(\mathbf{x}), k, i)}{\ln(1 + \exp(f_2(\mathbf{x}))})) \tag{16}$$

where $kthexcluding(v, k, i)$ is the k^{th} highest component of v excluding component i. Φ is the CDF of the standard normal distribution.

Two additional losses are added to the task specific loss and the final loss is defined as:

$$\ell = \ell_{\text{task}} + \mu(\ell_1 + \ell_2) \tag{17}$$

where μ is a scaling factor which we set it to 0.001 by default without further tuning.

B Spherical Space for Hierarchical Structures

Suppose we have a simple tree with three vertices x, y, z and z is the parent of x and y, our goal is to embed it into a sphere $\mathbb{S}$ while preserving the graph distance (the shortest path between a pair of vertices).

The graph distance between x and y is denoted as $d(x, y)$, which is also equal to $d(x, z) + d(z, y)$. Thus we have:

$$\frac{d(x, y)}{d(x, z) + d(z, y)} = 1 \tag{18}$$

We would like the distance between them on the sphere $\mathbb{S}$, denoted as $d_{\mathbb{S}}(x, y)$, to be close to $d(x, y)$. Let z be on the pole of the sphere and x, y be on the great circle. We have:

$$\frac{d_{\mathbb{S}}(x, y)}{d_{\mathbb{S}}(x, z) + d_{\mathbb{S}}(z, y)} \to 1, \textit{given that } \angle yzx \to \pi. \tag{19}$$

As such, the spherical space also has the capability of modeling the hierarchical structures.

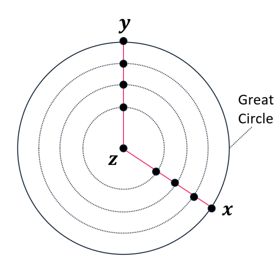

Figure 8: Embed trees on spheres.

C More Details and Results

C.1 More Properties of Switch Spaces

Our scoring function is not a standard distance metric as it does not satisfy the triangle inequality since we cannot guarantee that two pairs of points have the same active component spaces.

In product space, product of $\mathbb{E}^{b_1}$, ..., $\mathbb{E}^{b_N}$ is identical to the single space $\mathbb{E}^{b_1+\cdots+b_N}$. However, this is not the case in switch space when $K < N$ because the active Euclidean spaces may vary.

C.2 Additional Experimental Results

Params	WN18RR	FB15K-237
Learning Rate	0.01	0.005
Total Dimension	500	500
Batch Size	500	500
Max Epochs	200	200
# Negative Sample	50	50
N: # of total spaces	5	5
K: # of active spaces	2	4

Table 5: Hyper-parameters configuration for knowledge graph completion.

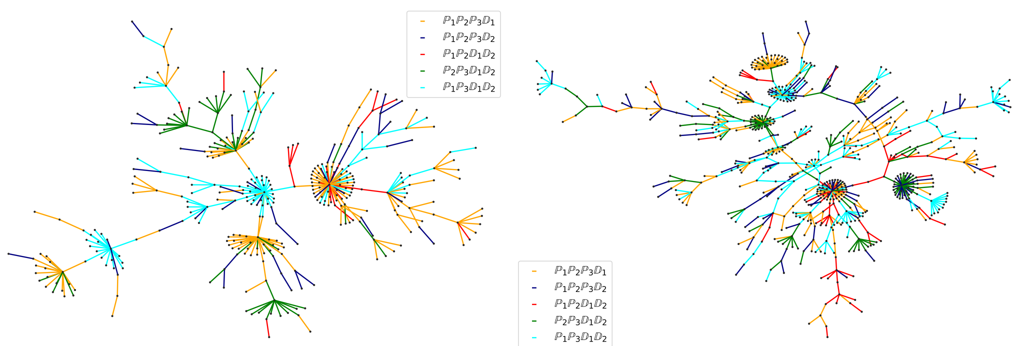

Figure 9: More examples of the connected component on FB15K-237.

Multiplying by the gating probability: We mentioned that the gating probability in equation (12) can be removed and we did not use it in our experiments. We aslo conducted experiments without removing it on WN18RR. The MRR and HR@3 are shown in Table 4 row A. As such, it is fair to say that multiplying the gating probability is optional.

Table 4: More results.

	WN18RR	
	MRR	HR@3
A	0.526	0.546
B	0.521	0.544
C	0.525	0.547

Impact of input x of the gating network : We find that the performance of using relation embeddings only (Table 4 row C) as the input of the gating network is slightly better than that of using entity embedding only (Table 4 row B). Intuitively, the relation is also a better indicator. For example, the relation "is-a-part-of" has a hierarchical property. The optimal solution is obtained by combining both.

More examples: We present more case studies on FB15K-237 in Figures 9 and 10.

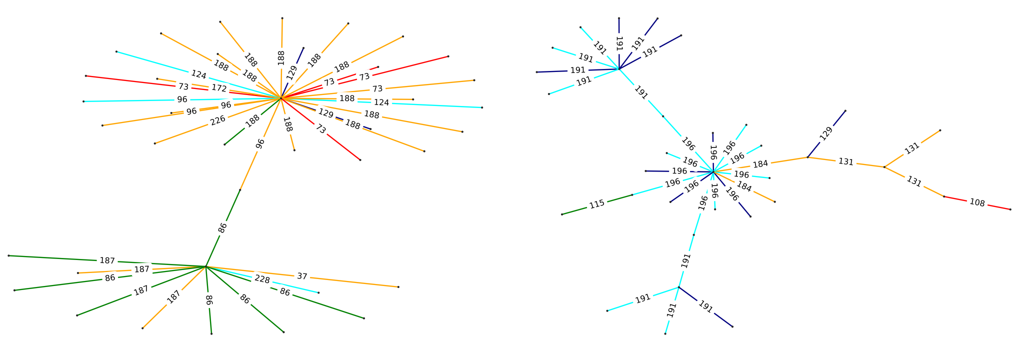

Figure 10: More examples of the connected component on FB15K-237.

C.3 EXPERIMENTAL DETAILS ON KNOWLEDGE GRAPH COMPLETION

Datasets We use two standard datasets including WN18RR (Bordes et al., 2013; Dettmers et al., 2018) and FB15K-237 (Bordes et al., 2013; Dettmers et al., 2018). WN18RR is taken from WordNet, a lexical database of semantic relations between words. It has $40,943$ entities, 11 relations, and $86,835/3,034/3,134$ training/validation/test triples. FB15K-237 is a subset of the Freebase knowl-

Model	WN18RR		FB15K-237	
	embedding dim	Model Size	Embedding Dim	Model Size
TransE/DistMult/ConvE	tuned from {128, 256, 512}	max 20.97M	tuned from {128, 256, 512}	max 7.57M
BoxE/MurP/TuckER	500	20.48M	500	7.39M
RotE/RotH	500	20.49M	500	7.63M
QuatE	4000	16.38M	4000	58.64M
RotatE	1000	40.95M	2000	29.32M
ComplEx-N3	1000	40.95M	1000	14.78M
SwisE	500	20.49M	500	7.63M

Table 6: Model size comparison on the knowledge graph completion task.

Dataset	C_1	C_2	#Interactions	density
MovieLens 100K	943	1682	100,000	0.063
MovieLens 1M	6040	3706	1,000,209	0.045

Table 7: Statistics of MovieLens 100K and MovieLens 1M.

edge graph, which is a global resource consisting of common and general information. It has $14,541$ entities, 237 relations, and $272,115/17,535/20,466$ training/validation/test triples.

Implementation Details The total dimension is fixed to 500 for fair comparison. The model size comparison is shown in Table 6. Learning rate is tuned among $\{0.01, 0.005, 0.001\}$. For all experiments, we reports the average over 5 runs. We set the kernel size to 5 and stride to 3 for convolution operation in the gating network. N is set to 5 and K is tuned among $\{1, 2, 3, 4\}$. The number of negative samples (uniformly sampled) per factual triple is set to 50. Optimizer Adam is used for model learning. Hyper-parameters are determined based on validation sets. We perform early stopping if the validation MRR stops increasing after 10 epochs. The key hyper-parameters are shown in Table 5.

Related work on knowledge graph completion A number of embedding techniques have been explored for knowledge graphs. Representative Euclidean models are RESCAL (Nickel et al., 2011), DistMult (Yang et al., 2015), TransE (Bordes et al., 2013), TuckER (Balazevic et al., 2019b), ConvE (Dettmers et al., 2018), RotE (Chami et al., 2020b), R-GCN (Schlichtkrull et al., 2018), and BoxE (Abboud et al., 2020). Complex/Hypercomplex number models such as ComplEx (Trouillon et al., 2016; Lacroix et al., 2018), RotatE (Sun et al., 2019), QuatE (Zhang et al., 2019) have shown better capability in modeling asymmetric relations. Recently, learning KG embeddings in hyperbolic spaces has gain increasing popularity. Hyperbolic models such as MurP (Balazevic et al., 2019a) and RotH (Chami et al., 2020b) can effectively capture the hierarchical relational patterns in KGs. As can be concluded from the literature, it is important for the KGE models to have the capability in capturing the relational and structural patterns in real-world KGs. However, current models usually focus on specific patterns and lose sight of the big picture. Our model SwisE is capable of modeling not only different relational patterns (symmetric, antisymmetric, and inversive, etc.) but also various structural patterns (hierarchical, cyclical, etc) of KGs.

C.4 Experimental Details on Recommender Systems

We conduct our experiments on two datasets: MovieLens 100K and MovieLens 1M (Harper & Konstan, 2015). We hold 70% actions in each user's interactions as the training set, 10% actions as the validation set for model tuning and the remaining 20% actions as the test set. All interactions (e.g., ratings) are binarized following the implicit feedback setting Rendle et al. (2009). We estimate the global average curvature with the algorithm described in (Gu et al., 2019) for the two datasets and obtain 0.190 for MovieLens 100K and 0.695 for MovieLens 1M, which suggests that they lean towards Euclidean/cyclical structures. Statistics of them are in Table 7.

For all models, the total dimension is fixed to 100 for fair comparison.As such, the model sizes are the same. The curvatures for spherical and hyperbolic models are set to 1 and -1, respectively. N is set to 5 and K is tuned among $\{1, 2, 3, 4\}$. Regularization rate is chosen from $\{0.1, 0.01, 0.001\}$. m is fixed to 0.5. Adam is also adopted as the optimizer.