Yale

Applications of Hyperbolic Representation Learning

CPSC483: Deep Learning on Graph-Structured Data

Rex Ying

Readings

- Readings are updated on the website (syllabus page)
- **Lecture 19 readings:**
  - HGCN
  - Hyperbolic GNN survey
- **Lecture 20 readings:**
  - Neural Distance Embeddings
  - Hyperbolic Cone Embedding

Content

- **Poincaré Ball Model and Hyperbolic Embedding**
- **Hierarchical Sequence Embeddings**
  - **Genomic Sequences**
- **Cone Embeddings**
  - **Knowledge Graphs**

Content

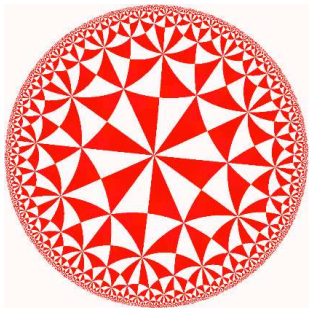
- **Poincaré Ball Model and Hyperbolic Embedding**

- **Hierarchical Sequence Embeddings**
  - **Genomic Sequences**

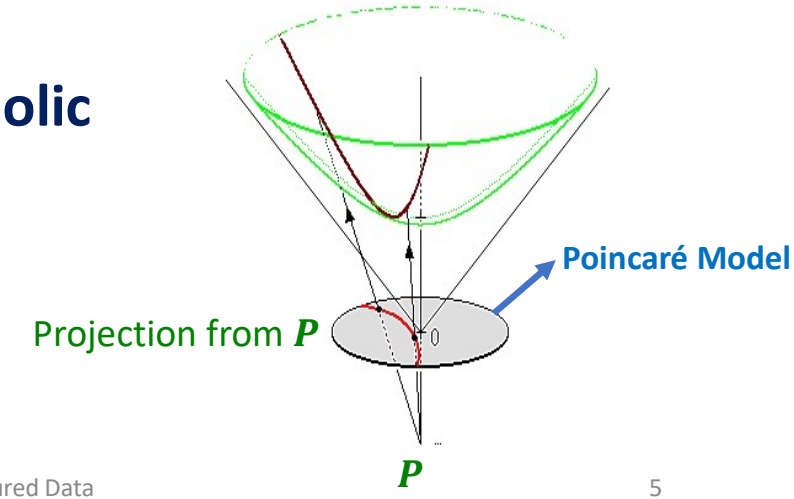
- **Cone Embeddings**
  - **Knowledge Graphs**

Recap: Poincaré Ball Model

- **Poincaré Ball Model** $\mathbb{D}^n = \{\boldsymbol{x} \in \mathbb{R}^n : \|\boldsymbol{x}\| < 1\}$
  - $\|\cdot\|$ is the Euclidean norm
  - open ball without boundary
  - It is a projection of hyperboloid in Minkowski space
- The **metric** of Poincaré Ball Model is different from the Euclidean metric!
- **n-dimensional** Poincaré Ball Model is a **hyperbolic space** of dimension n



2-D Poincaré Model Visualization



Recap: Metric and Distance

- **Poincaré Metric:**

$$g_{\boldsymbol{x}}^{B} = \left(\frac{2}{1-\|\boldsymbol{x}\|^2}\right)^2 g^E,$$

where g^E is the Euclidean metric.

- **Hyperbolic distance with Poincaré Model:**

$$d_{\mathbb{D}}(\boldsymbol{x}, \boldsymbol{y}) = \operatorname{arcosh}\left(1 + \frac{2\|\boldsymbol{x}-\boldsymbol{y}\|^2}{(1-\|\boldsymbol{x}\|^2)(1-\|\boldsymbol{y}\|^2)}\right)$$

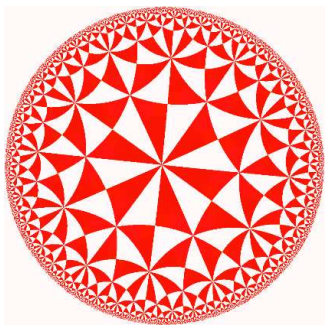
- **Poincaré Norm:**

$$\|\boldsymbol{x}\|_{\mathbb{D}} \coloneqq d_{\mathbb{D}}(0, \boldsymbol{x}) = 2\operatorname{artanh}(\|\boldsymbol{x}\|)$$

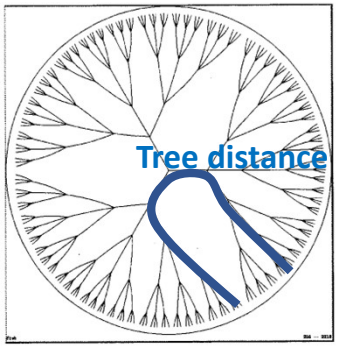
$$\operatorname{arcosh}(x) = \ln(x + \sqrt{x^2+1})$$
$$\operatorname{artanh}(x) = \frac{1}{2}\ln(\frac{1+x}{1-x})$$

Recap: Hyperbolic Embedding

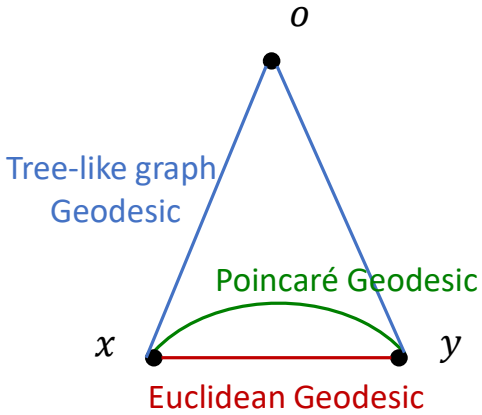
- **Hierarchical and Tree-like** graphs are **best** with the **hyperbolic space**!
  - Lower Distance Distortion
  - Exponential Volume Growth



Poincaré Visualization



Distance of tree-like graphs



Today: we show how **hyperbolic space (Poincaré Ball Model)** helps to better capture hierarchical relations in tree-like graphs!

- Biological Sequences
- Heterogenous Knowledge Graph

Content

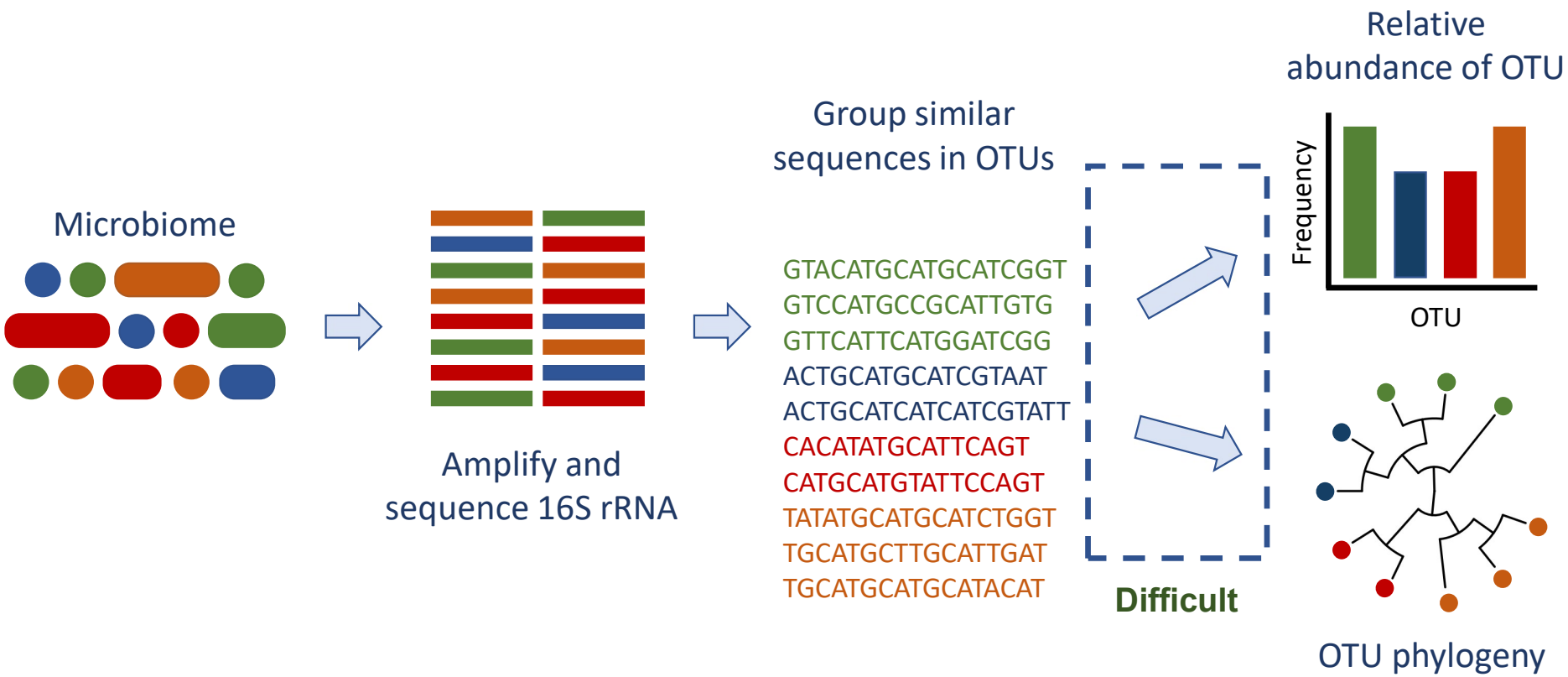
- Poincaré Ball Model and Hyperbolic Embedding
- **Hierarchical Sequence Embeddings**
  - **Genomic Sequences**
- Cone Embeddings
  - Knowledge Graphs

Motivation

- **Human Microbiome:** the "forgotten organ", wide variety of bacteria, fungi and viruses on the surfaces of our body, with its disfunctions linked to diseases and phenotypes:
  - Diabetes
  - Antibiotic-resistance
  - Inflammatory bowel disease
  - Allergies
  - Fibromyalgia
  - Depression
  - Various cancers

→ great potential as biomarker for **diagnosis** and target for **treatments**

Existing Approaches



Inspired by Morgan & Huttenhower (2012)

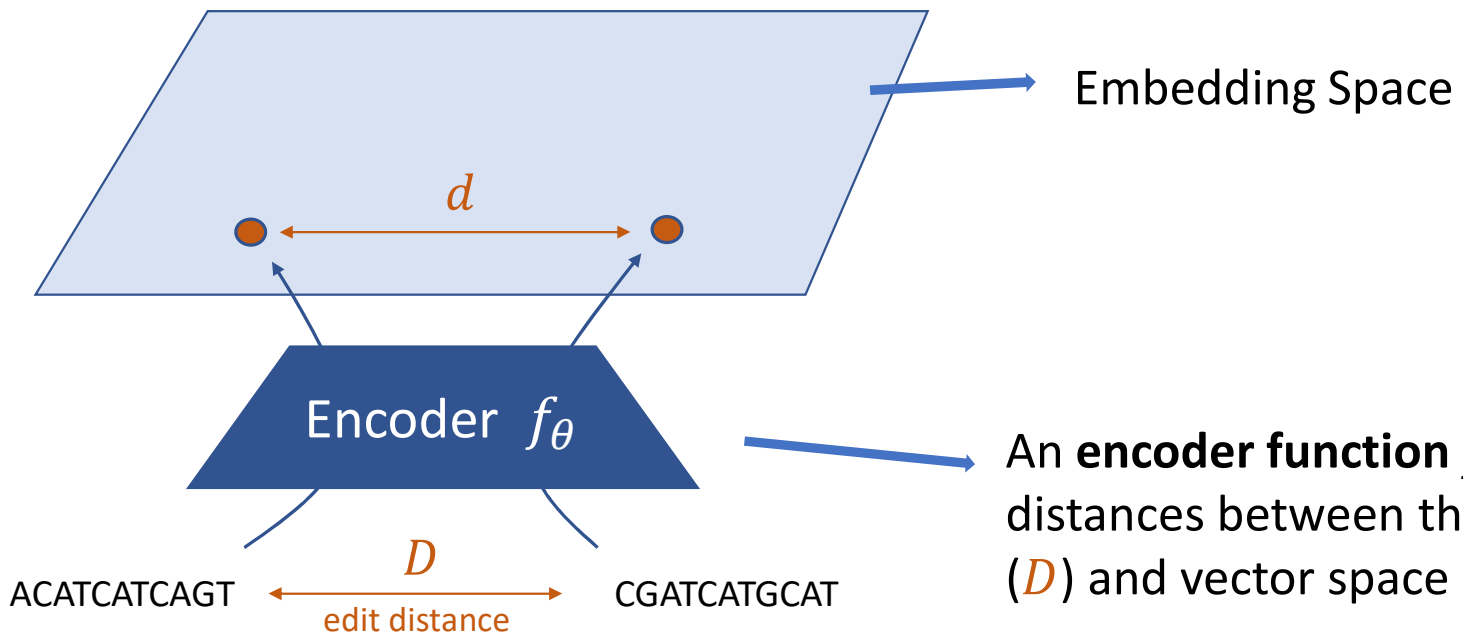
Problem

- **Exponentially increasing amount** of genomic data available but often this is not used in bioinformatics:
  - Bioinformatics algorithms cannot exploit the underlying **manifold assumption** to generate effective representations and heuristics.
  - Existing machine learning frameworks do not fit well with tasks formalized as **combinatorial optimization problems**.

Need for a new framework for **data-dependent** approaches to learn the **representation** of biological sequences for bioinformatics tasks.

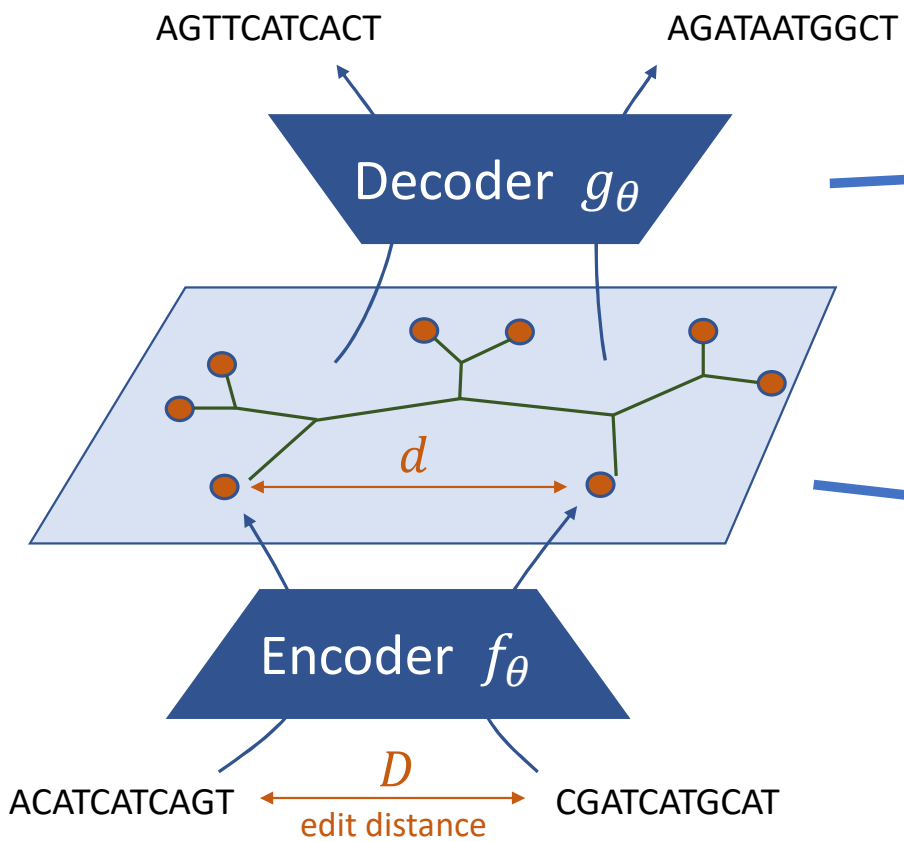
Neural Distance Embeddings (1)

Neural Distance Embeddings (NeuroSEED):



An **encoder function f_θ** preserves distances between the sequence (D) and vector space (d)

Neural Distance Embeddings (2)

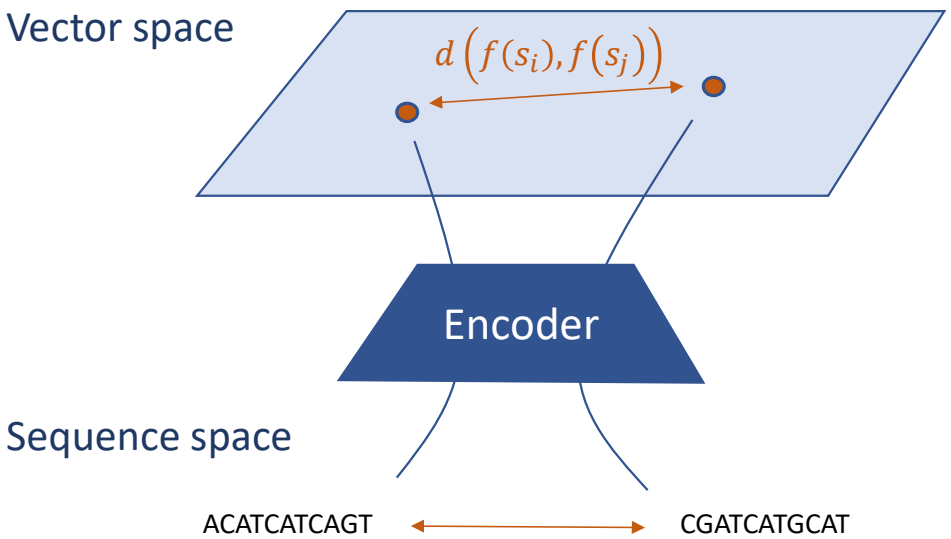


Decoder function g_θ decodes sequences from the embedding space.

The embedding space can be used to study the relationships between sequences

Edit Distance Approximation

Method: learn the encoding function f that **minimizes** the approximation error given a distance function d :



$ED(s_i, s_j)$: edit distance between s_i and s_j

$$\min_f \mathbb{E}\left(ED(s_i, s_j) - d\left(f(s_i), f(s_j)\right)\right)^2$$

How to choose the appropriate distance function d to preserve the distance?

Distance Functions

Consider a pair of vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ of dimension n,

- **Manhattan (L1)**: $d(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_1 = \sum_{i=1}^{n} |x_i - y_i|$
- **Euclidean (L2)**: $d(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_2 = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$
- **Squared distance**: $d(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_2^2 = \sum_{i=1}^{n} (x_i - y_i)^2$
- **Cosine similarity**: $d(\boldsymbol{x}, \boldsymbol{y}) = 1 - \frac{\boldsymbol{x} \cdot \boldsymbol{y}}{\|\boldsymbol{x}\| \|\boldsymbol{y}\|} = 1 - \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}}$
- **Hyperbolic** (use n-dimensional Poincaré Ball Model):

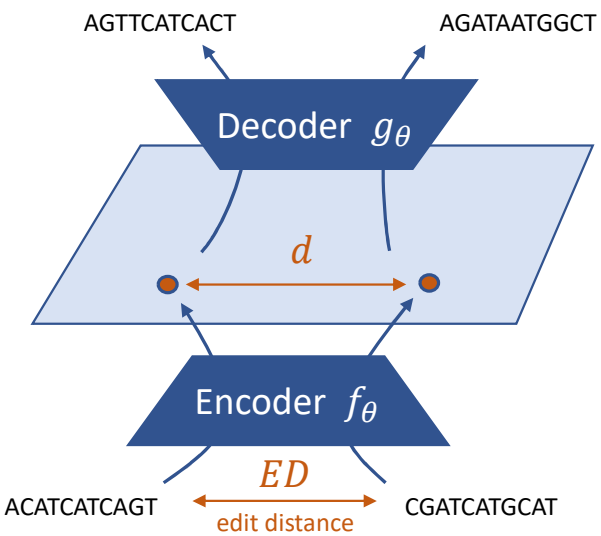
$$d(\boldsymbol{x}, \boldsymbol{y}) = \operatorname{arcosh}\left(1 + \frac{2\|\boldsymbol{x} - \boldsymbol{y}\|^2}{(1 - \|\boldsymbol{x}\|^2)(1 - \|\boldsymbol{y}\|^2)}\right)$$

Pipeline Overview

- **Step 1:** Collect training sequences. Generate ground-truth (edit distance)
- **Step 2:** Train the encoder and decoder
  - Minibatches of sequence pairs
  - Loss function:

$$L(\theta, S) = \sum_{s_1, s_2 \in S} (ED(s_1, s_2) - \alpha d(f_\theta(s_1), f_\theta(s_2)))^2$$

    - ED is the ground-truth edit distance, f_θ is the encoder, $d(\cdot,\cdot)$ is the distance in the embedding space, α is the learnable scalar.
- **Step 3:** Inference step. Generate sequence from the embedding space
- **Step 4:** Downstream tasks



Alignment-free and Alignment-based Methods

- **Alignment-free:** compare the biological sequences by counting the frequencies of substrings with a specific length in respective sequences.
- **Alignment-free Baselines:**
  - k-mer: represents sequences by the frequency vector of subsequences of a certain length k
  - FFP: looks at the Jensen-Shannon divergence between distributions of k-mer.
- **Embedding-based methods:**
  - Use an encoder to map sequences into the embedding space and estimate distance using embeddings

EDA: Experimental Results

- Performance of **Edit Distance Approximation (%RMSE)** on real-world datasets (lower,
- Baseline geometries were chosen by best average performance

		RT988		Qiita		Greengenes		
	Model	**Baseline**	**Hyperbolic**	**Baseline**	**Hyperbolic**	**Baseline**	**Hyperbolic**	**Training/Inference**
	NW alignment	-	-	-	-	-	-	- / 17.5h
Alignment-free methods	**4-mer**	1.79	-	6.01	-	5.93	-	7s / 7s
	5-mer	1.41	-	5.03	-	3.60	-	29s / 29s
	6-mer	1.47	-	5.72	-	3.15	-	118s / 118s
	FFP 8	12.03	-	20.42	-	10.26	-	360s / 360s
	FFP 9	11.86	-	17.53	-	8.63	-	679s / 679s
	FFP 10	10.80	-	16.16	-	14.13	-	1274s / 1274s
	Linear	21.3±7.0	0.51±0.01	4.39±0.09	2.50±0.01	1155±18	2.70±0.01	1.1h / 3s
Embedding approaches	**MLP**	1.10±0.05	0.59±0.20	4.36±0.19	1.85±0.02	4.38±0.13	2.53±0.03	0.9h / 3s
	CNN	0.58±0.05	0.59±0.01	2.68±0.05	**1.56±0.01**	1.37±0.04	**1.00±0.01**	2.1h / 6s
	GRU	1.10±0.11	2.56±3.33	3.30±0.06	2.60±0.16	1.61±0.02	1.18±0.16	7.4h / 65s
	Global T.	0.52±0.01	**0.46±0.01**	2.10±0.05	1.83±0.03	2.09±0.03	1.91±0.07	2.2h / 3s
	Local T.	0.57±0.00	**0.45±0.01**	2.42±0.02	1.86±0.02	1.85±0.04	1.89±0.05	2.0h / 3s

Best

Worst

EDA: Experimental Results

- Performance of **Edit Distance Approximation (%RMSE)** on real-world datasets (lower,
- Baseline geometries were chosen by best average performance

		RT988		Qiita		Greengene		
	Model	**Baseline**	**Hyperbolic**	**Baseline**	**Hyperbolic**	**Baseline**	**Hyperbolic**	**Training/Inference**
	NW alignment	-	-	-	-	-	-	- / 17.5h
Alignment-free methods	**4-mer**	1.79	-	6.01	-	5.93	-	7s / 7s
	5-mer							29s / 29s
	6-mer							118s / 118s
	FFP 8							360s / 360s
	FFP 9							679s / 679s
	FFP 10							1274s / 1274s
	Linear	2					1	1.1h / 3s
Alignment-based methods	**MLP**	1.					3	0.9h / 3s
	CNN	0.58±0.05	0.59±0.01	2.68±0.05	**1.56±0.01**	1.37±0.04	**1.00±0.01**	2.1h / 6s
	GRU	1.10±0.11	2.56±3.33	3.30±0.06	2.60±0.16	1.61±0.02	1.18±0.16	7.4h / 65s
	Global T.	0.52±0.01	**0.46±0.01**	2.10±0.05	1.83±0.03	2.09±0.03	1.91±0.07	2.2h / 3s
	Local T.	0.57±0.00	**0.45±0.01**	2.42±0.02	1.86±0.02	1.85±0.04	1.89±0.05	2.0h / 3s

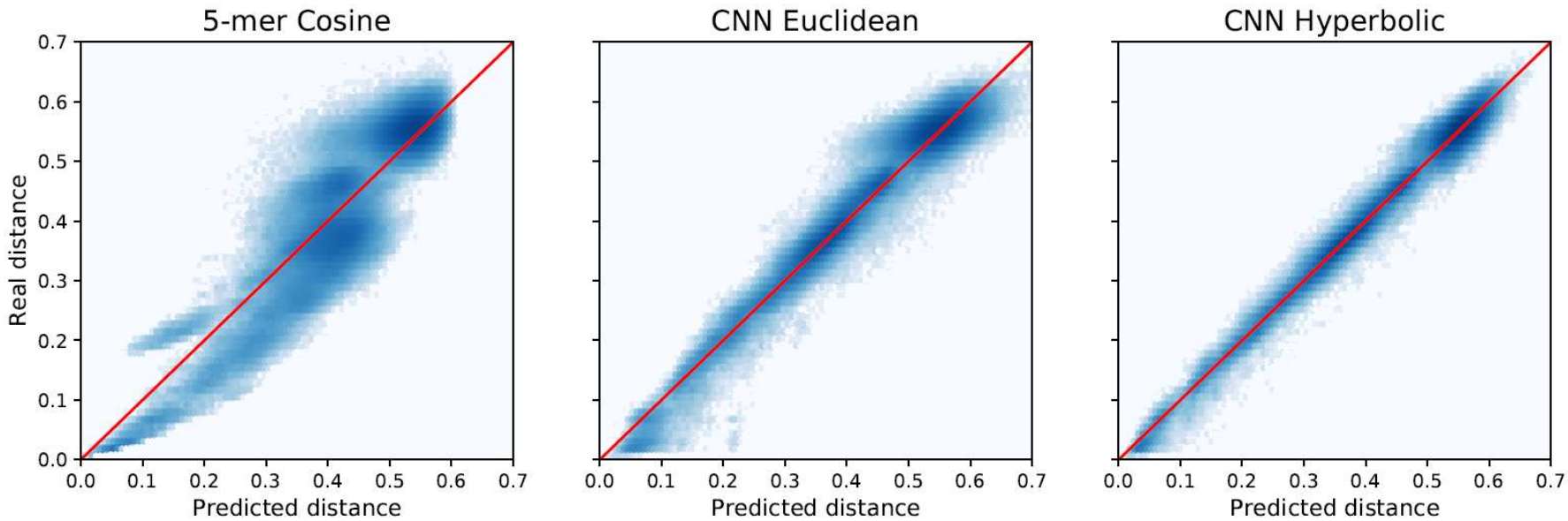
Training and inference time for 5k sequences:

- The distance function does not significantly impact the runtime.
- Alignment-based methods have faster inference time!

Best

Worst

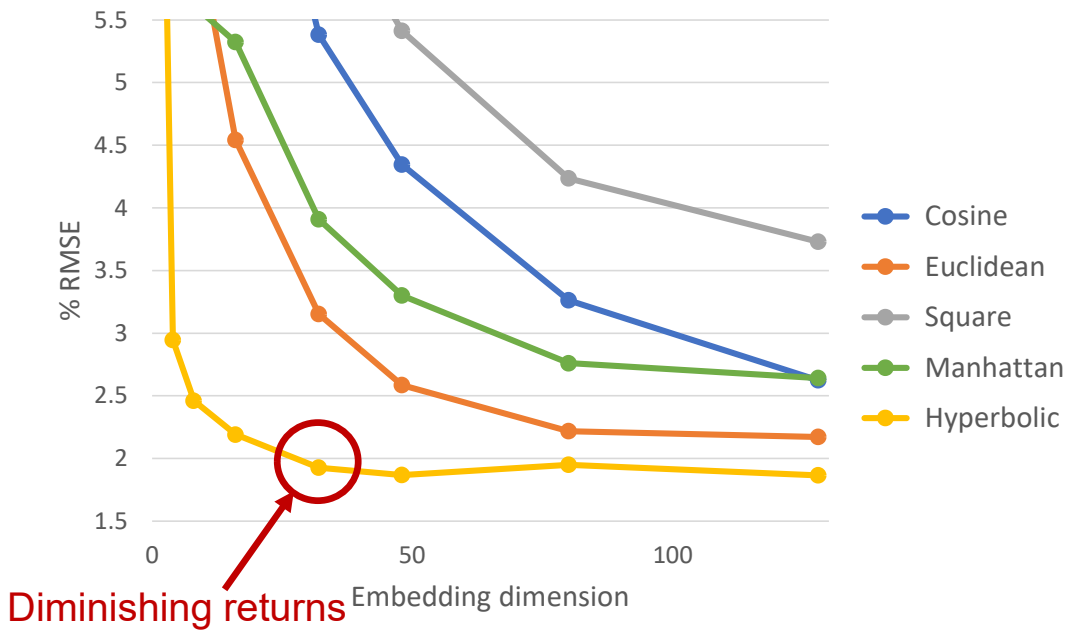
EDA: Error Analysis



- The blue shade represents the density of points
- The CNN model follows much more tighly the red line **in the hyperbolic space**

EDA: Dimensionality

- The hyperbolic space is **more efficient** than other geometries in capturing hierarchies
- The model with the hyperbolic space reaches the "elbow" at dimension 32
- Hyperbolic embeddings with only 8 dimension achieves the performance of other geometries with 128 dimensions

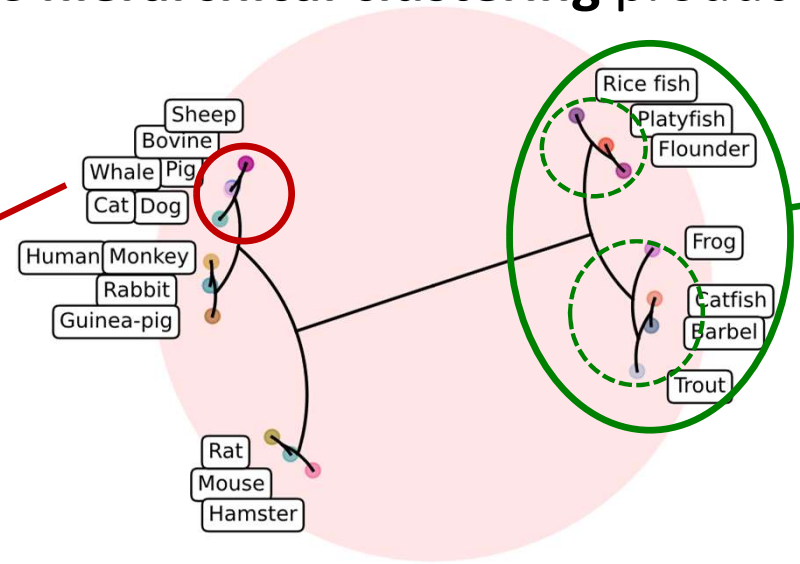


Edit distance approximation %RMSE on Qiita dataset for a global transformer **with different distance functions**

Task: Hierarchical Clustering (HC)

- **Hierarchical clustering(HC):**
  given a pairwise distance function, define a tree with **internal points corresponding to clusters** and **leaves to datapoints.**
- An example of the **hierarchical clustering** produced on the **Poincaré disk**:

Closely related embeddings (gene sequences)



2 related high-level clusters

Unsupervised Method for HC

Unsupervised Method:

- **Agglomerative Clustering** is the most commonly used method for Hierarchical Clustering task.
- NeuroSEED embeddings **reduces the complexity** to generate the pairwise distance matrix from $\mathcal{O}(N^2M^2/\log M)$ to $\mathcal{O}(N(M+N))$ for N sequences of length M

HC Experimental Results (1)

- Average Linkage: the best performing clustering heuristic across all models.
- **No statistical difference** in the quality of the hierarchical clustering produced with ground truth distances compared to that with NeuroSEED embeddings.

Baselines	
Single Linkage	0.628
Complete Linkage	0.479
Average Linkage	**0.000**

Average Linkage % increase in Dasgupta's cost of NeuroSEED models compared to hierarchical clustering using ground truth distance

- Single Linkage, complete Linkage: alternative hierarchical clustering methods
- **Dasgupta cost:**

$$C(T, w) = \sum_{ij} w_{ij} |\text{leaves}(T[i \vee j])|$$

Hierarchical Clustering can be described as a rooted tree T, for two datapoints i and j. w_{ij} is their pairwise similarity. $i \vee j$ is their lowest common ancestor in T. $\text{leaves}(T[i \vee j])$ is the set of leaves of the subtree rooted at $i \vee j$.

- **Dasgupta cost** measures how well the tree generated respects the similarities between datapoints **(the lower, the better)**

HC Experimental Results (2)

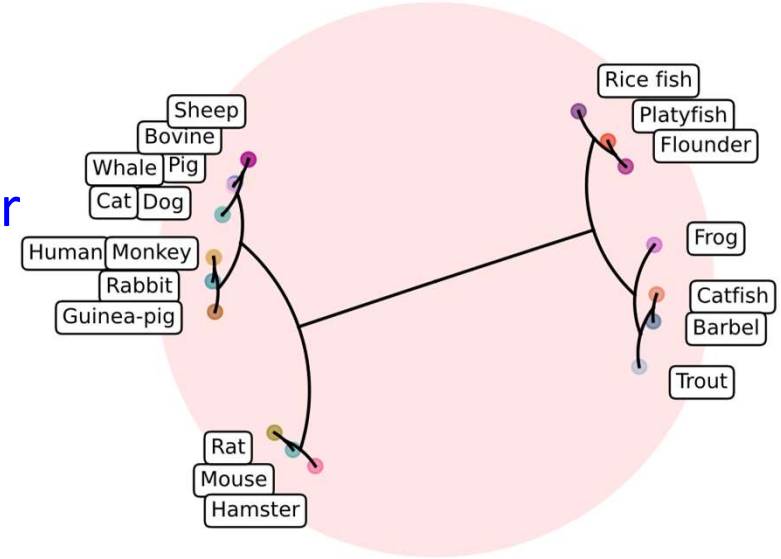
- Average Linkage: the best performing clustering heuristic across all models.
- There is statistical difference between the different architectures and geometries.
- NeuroSEED embedding produces lower Dasgupta's cost!

Model	Cosine	Euclidean	Square	Manhattan	Hyperbolic
4-mer	0.261	0.260	0.242	0.191	0.299
Linear	0.062±0.007	0.172±0.036	0.153±0.037	0.177±0.026	0.028±0.005
MLP	0.169±0.054	0.095±0.021	0.289±0.094	0.178±0.029	0.035±0.004
CNN	0.028±0.003	0.030±0.004	0.067±0.022	0.081±0.047	**-0.004±0.015**
GRU	-	0.042±0.006	0.068±0.010	0.069±0.015	0.066±0.043
Global T.	0.032±0.014	**0.003±0.008**	0.038±0.005	**0.002±0.003**	**0.000±0.006**
Local T.	0.035±0.003	0.022±0.008	0.034±0.005	0.022±0.003	**0.000±0.007**

Average Linkage % increase in Dasgupta's cost between different architectures and geometries

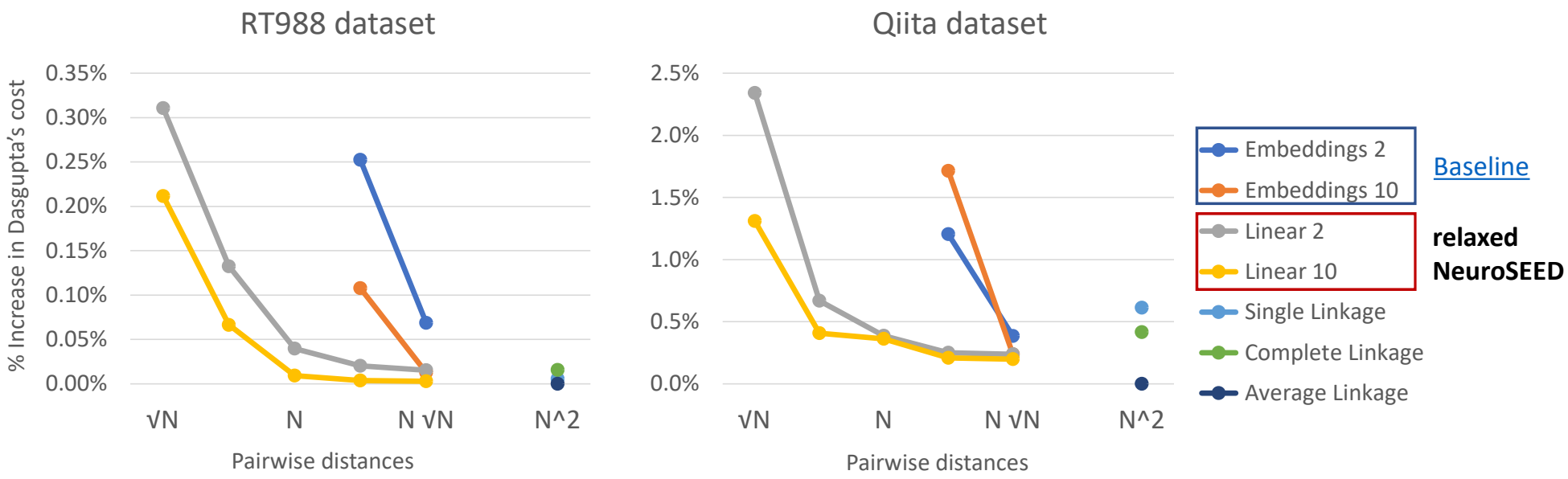
Supervised Heuristics for HC

- Use the continuous relaxation[1] of Dasgupta's discrete cost as loss function to embed sequences in the hyperbolic space. No pretraining required.
- NeuroSEED significantly decreases the number of pairwise distances required, by directly mapping the sequences into the space.



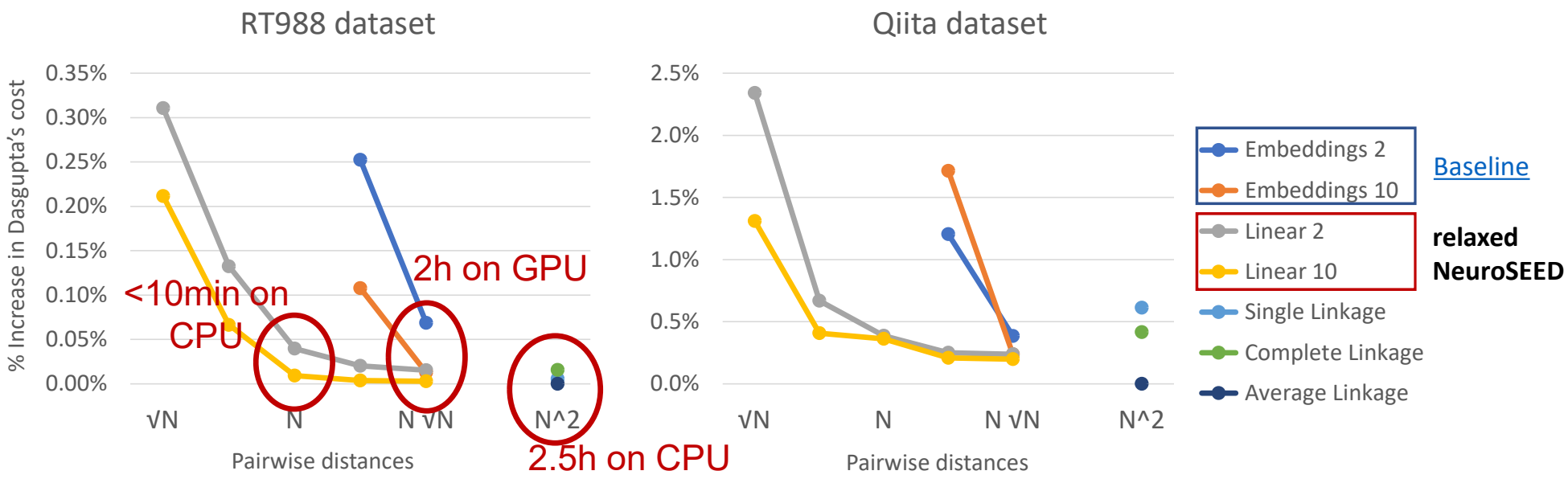
HC Experimental Results (3)

- The performances are reported as the percentage increase in cost compared to the one of the Average Linkage (best performing)



HC Experimental Results (4)

- A simple linear layer **with only N** pairwise distances obtains very similar results to agglomerative clustering (N^2 distances) and hyperbolic embedding baselines ($N\sqrt{N}$ distances)



Task: Multiple Sequence Alignment

Multiple Sequence Alignment: process of sequence alignment of three or more biological sequences

Two approaches :

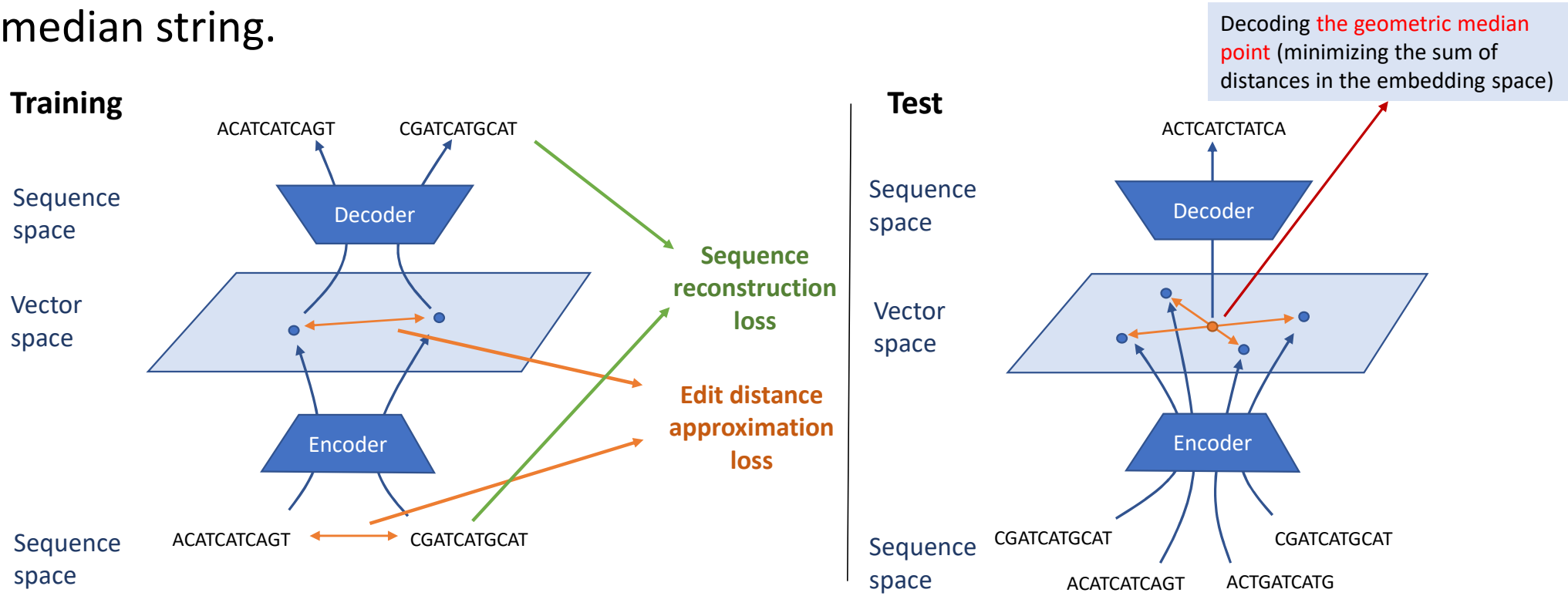
1. Unsupervised: uses the models pretrained for edit distance approximation followed by the **Clustal** method (a popular MSA heuristic).

Model	Cosine	Euclidean	Hyperbolic
Linear	60.6±35.1	111.3±3.6	57.5±22.0
MLP	72.3±11.8	53.6±3.1	-11.7±18.9
CNN	31.0±16.2	4.7±9.7	**-16.3±16.1**
Global T.	39.4±74.3	1.9±3.8	31.1±21.8
Local T.	31.9±30.5	8.6±14.1	**-20.1±7.3**

Percentage improvement in the alignment cost (the lower the better) returned by Clustal

Supervised Heuristics for MSA

<u>2. Steiner String</u>: autoencoder with distance preserving latent space to predict median string.

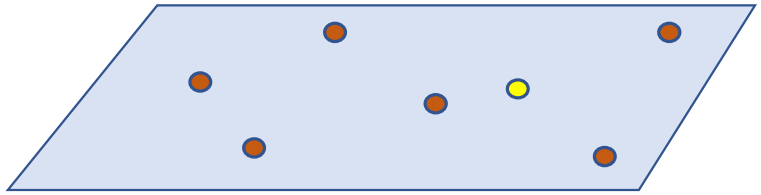


Summary: Applications of Sequence Embeddings

- Unsupervised embeddings, expensive string problems

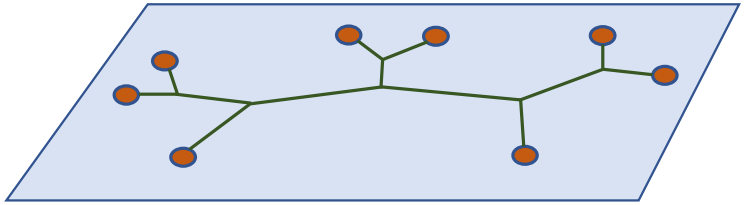
Closest sequence retrieval:

- database searching
- sequence classification



Understand relations between groups of sequences:

- sequence clustering
- hierarchical clustering



Summary: Applications of Sequence Embeddings

- Unsupervised embeddings, expensive string problems

Closest sequence retrieval:

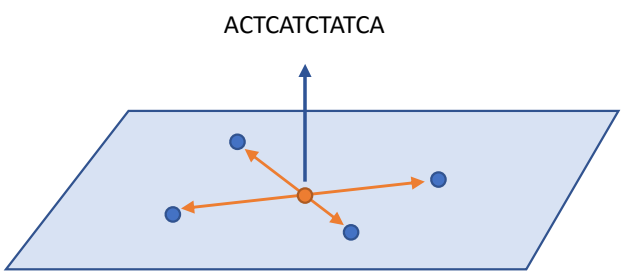
- database searching
- sequence classification

Decode from embedding space:

- compression of sequences
- retrieving alignment
- decoding new sequences

Understand relations between groups of sequences:

- sequence clustering
- hierarchical clustering



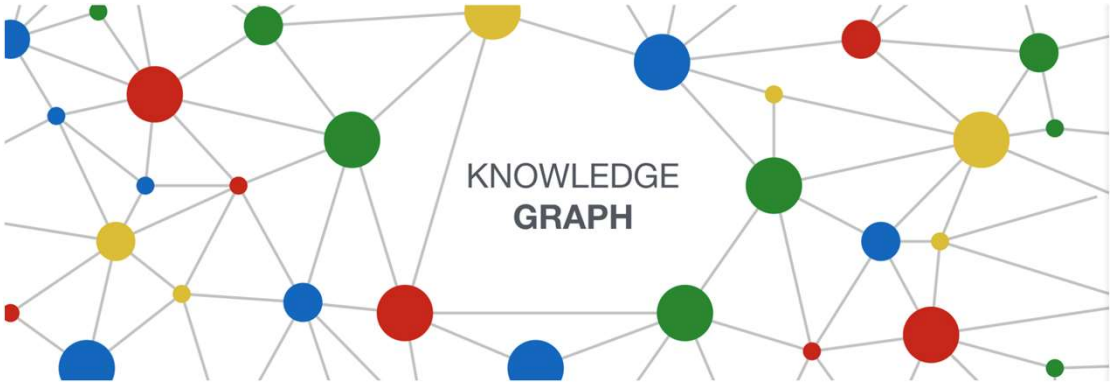
Content

- Poincaré Ball Model and Hyperbolic Embedding
- Hierarchical Sequence Embeddings
  - Genomic Sequences
- **Cone Embeddings**
  - **Knowledge Graphs**

Background: Knowledge Graph

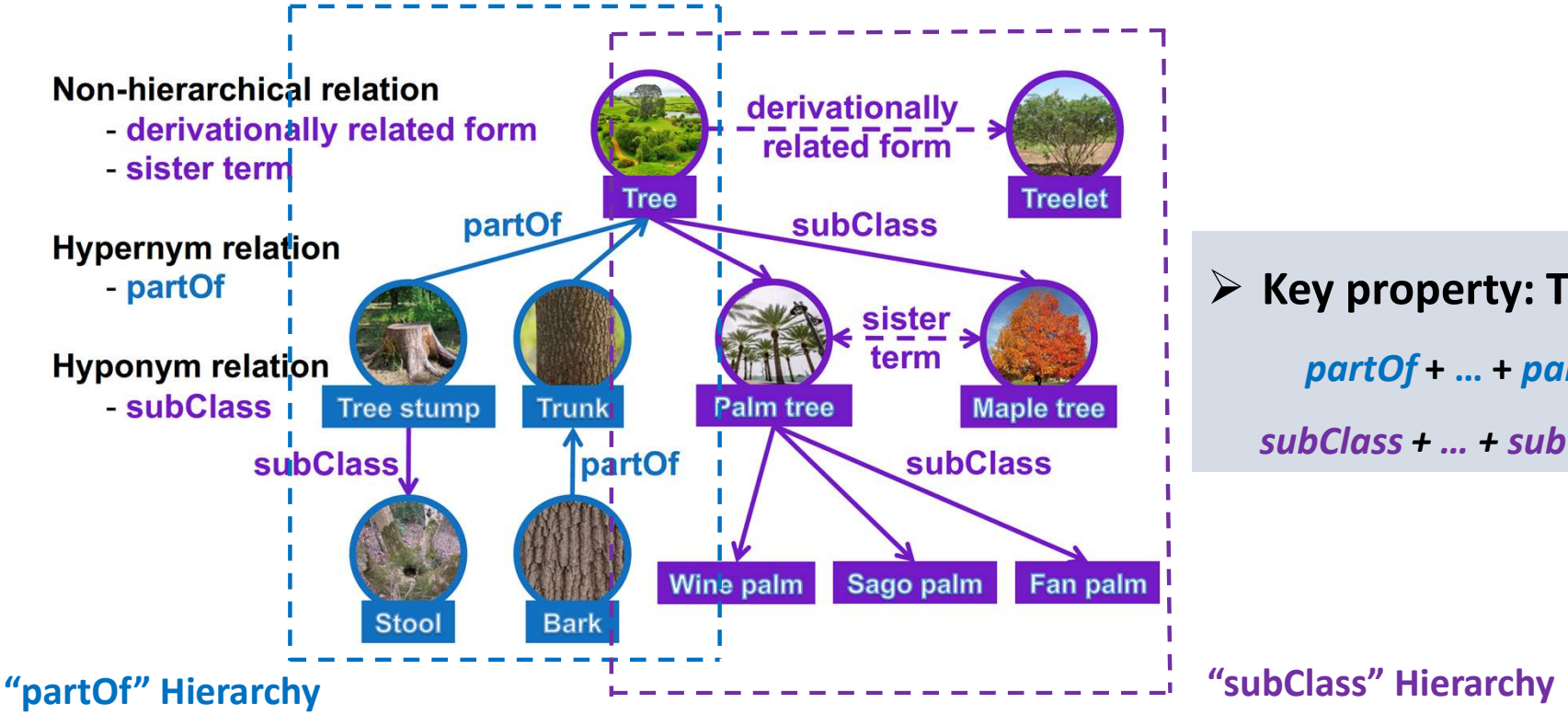
- **Knowledge graph contains**:
  - Hierarchical relations
  - Non-hierarchical relations

ConceptNet
An open, multilingual knowledge graph



WIKIDATA

Background: Heterogeneous Hierarchies



- **Key property: Transitivity**

  partOf + … + *partOf* = *partOf*

  subClass + … + *subClass* = *subclass*

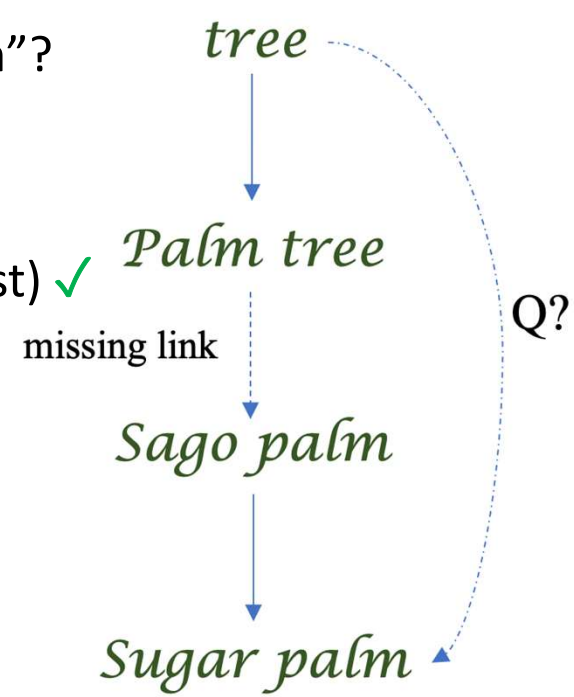
Why Modeling Transitivity?

Ancestor-Descendant prediction task:

- Do **hierarchical reasoning** that involves cross-layer connection
- **Query:** is there a relationship between "tree" and "Sugar palm"?

Such tasks require:

- Model non-hierarchical properties (to infer the missing link first) ✓
- Model transitivity of hierarchical relations ✓



Previous Methods

- **Knowledge graph embedding model**
  - TransE, ComplEx, RotatE, TuckER, etc.
  - Model transitivity: ✗

- **Geometry-based modeling on hierarchy**
  - Order, Box, Cone
  - Model heterogeneous hierarchies: ✗
  - Model non-hierarchical relations: ✗

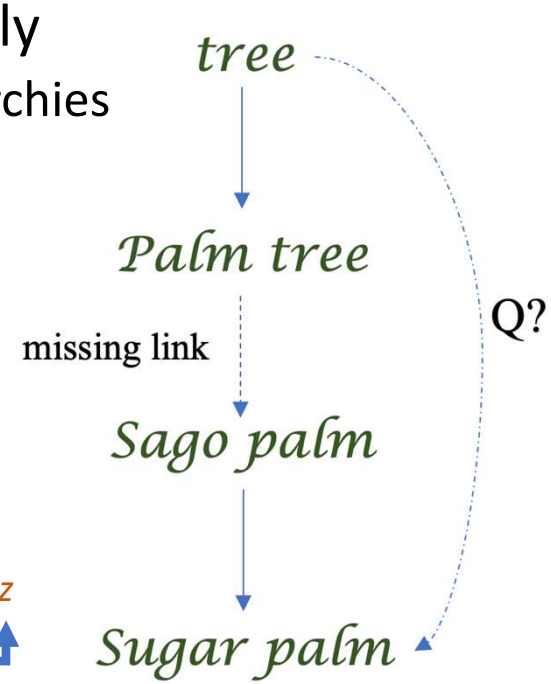
➢ **Transitivity:**

partOf* + … + *partOf* = *partOf

subClass* + … + *subClass* = *subclass

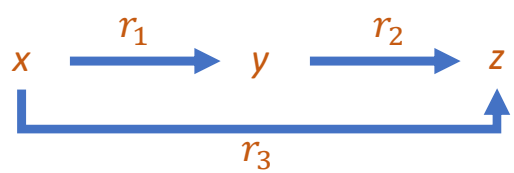
Cone Embedding

- **ConE** (**Con**e **E**mbedding)
- Knowledge graph embedding model that simultaneously
  - Capture **transitive closure property** of heterogeneous hierarchies
    - **Partial Ordering**
      - ***partOf; subClass …***



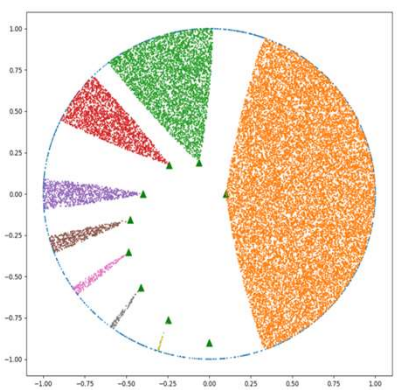
  - Model other **non-hierarchical relations**
    - **Symmetry**
      - $r(x, y) \Rightarrow r(y, x)$

    - **Composition**
      - $r_1(x, y) \wedge r_2(y, z) \Rightarrow r_3(x, z) \quad \forall x, y, z$



Idea: Nested Cones for Hierarchy

- Hyperbolic space better preserves **tree-like structure**
- Each hierarchical relation induces a **partial ordering**
- Use **hyperbolic entailment cones**[1] to capture the partial ordering

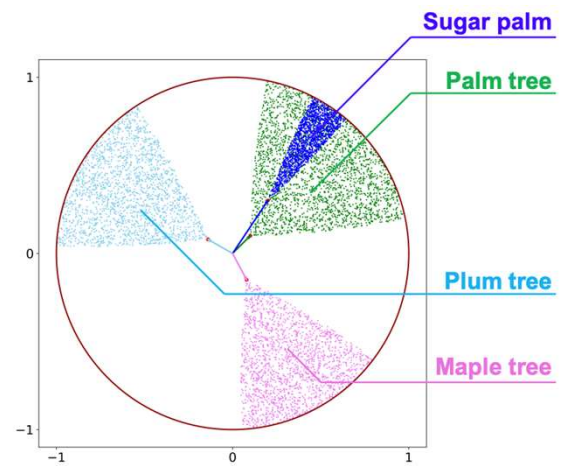


Poincaré angular cones [1]

subClass

Sugar Palm → **Palm Tree**

Cone of PalmTree contains the cone of SugarPalm



Hyperbolic entailment cones in hyperbolic plane

[1] Hyperbolic Entailment Cones for Learning Hierarchical Embeddings

Hyperbolic Cones

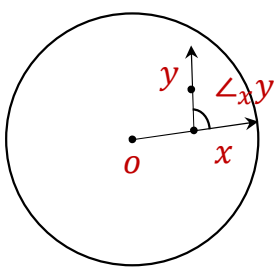
- Let C_x denote the cone at apex x. The entailment cones satisfy transitivity:

$$\forall x, y \in B^d : y \in C_x \Rightarrow C_y \subseteq C_x$$

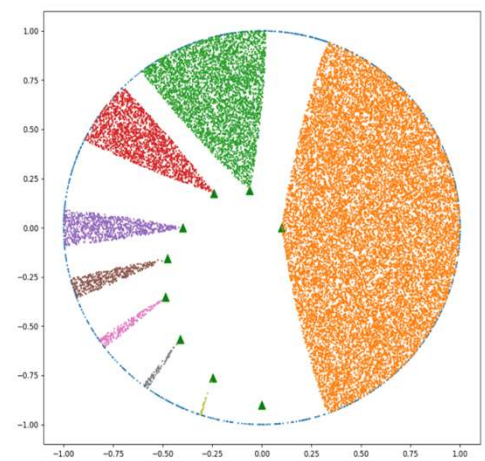
- To **ensure the transitivity of nested cones** [see proof in Paper (Appendix F)]:

$$C_x = \{y \in B^d | \angle_x y \leq \sin^{-1}(K \frac{1-\|x\|^2}{\|x\|})\}, \quad K \text{ is a hyper-parameter}$$

- **Half Aperture** $\phi_x = \sin^{-1}(K \frac{1-\|x\|^2}{\|x\|})$



Note: Half Aperture ϕ_x decreases as $\|x\|$ increases
$\Rightarrow$ guarantee the **transitive closure**



Cone Embedding

Entity ➡ Hyperbolic Cone

Relation

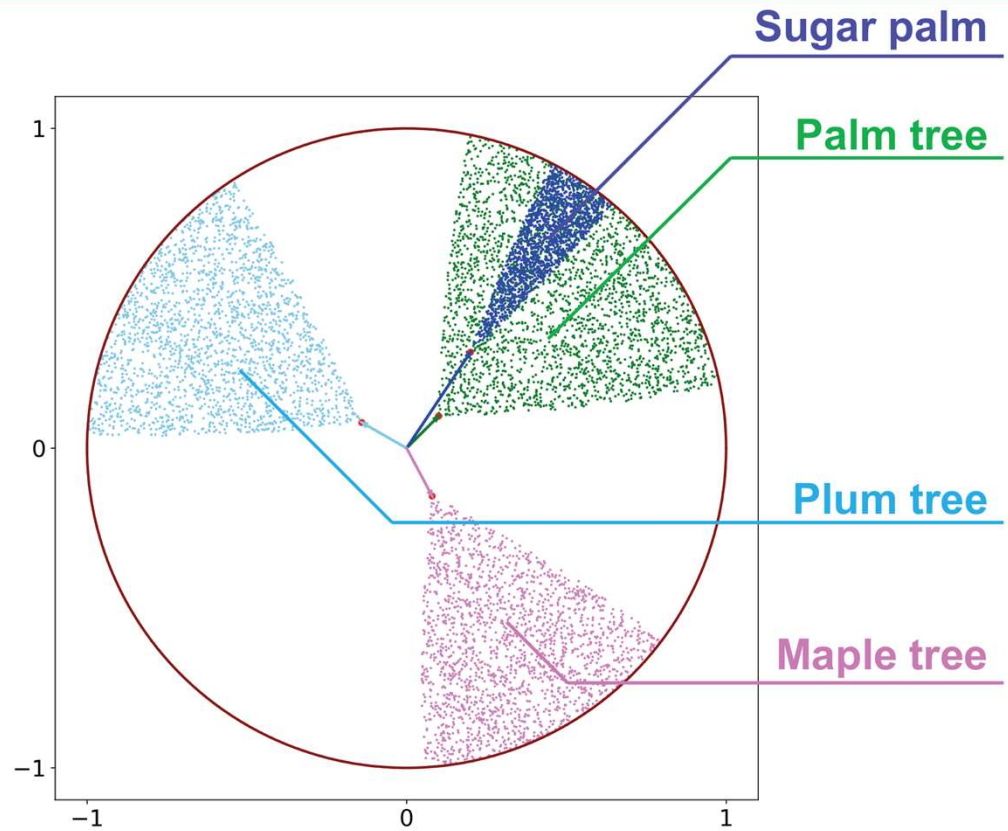
⬇

Cone Transformation

Partial Ordering

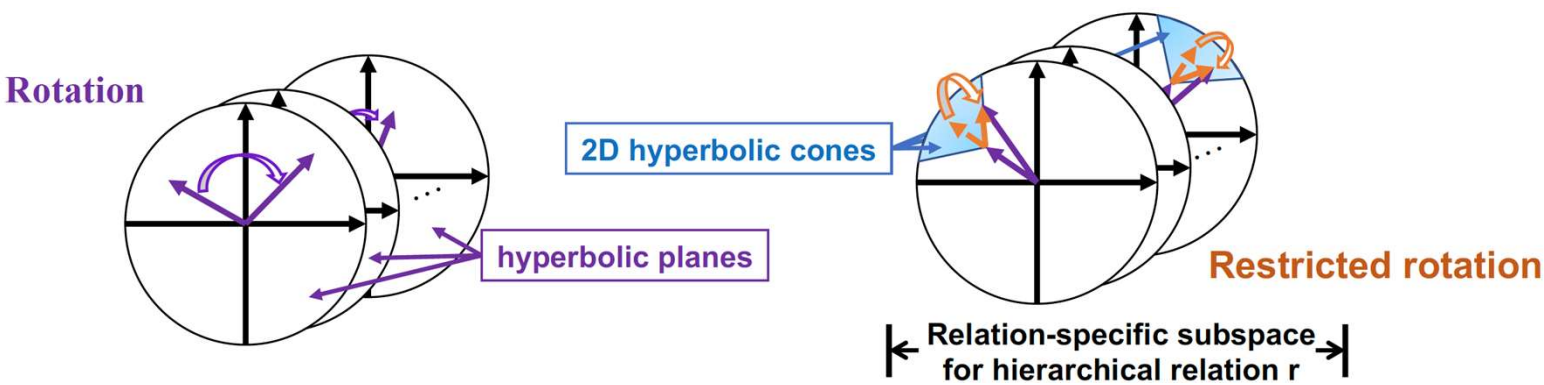
⬇

Cone Containment



Model Overview

- We use the **product of 2d hyperbolic planes** as the embedding space
  - $S = B^2 \times B^2 \times \cdots \times B^2$ is a product space of d 2-dimensional Poincaré disk, resulting in an embedding dimension of $2d$.
- For each relation r, the planes are separated into 2 types
  - Use **restricted rotation** in a relation-specific subspace to **model hierarchical relation**
  - Use **a general rotation** to **model non-hierarchical relation**



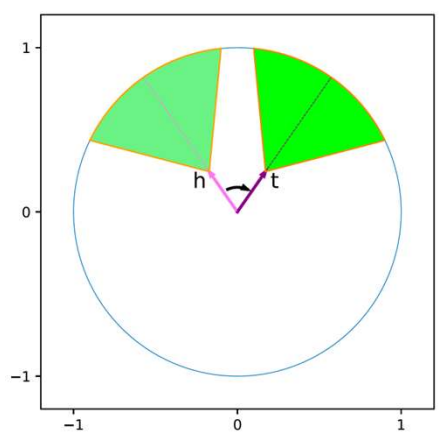
Embedding Space

- Entity h: $\boldsymbol{h} = (\boldsymbol{h_1}, \boldsymbol{h_2}, \dots, \boldsymbol{h_d})$, $\boldsymbol{h_i}$ is the apex of the i-th 2D hyperbolic cone.
- Relation r: $\boldsymbol{r} = (\boldsymbol{r_1}, \boldsymbol{r_2}, \dots, \boldsymbol{r_d})$ where $\boldsymbol{r_i} = (s_i, \theta_i)$ parameterizes transformation for the i-th hyperbolic plane.
  - s_i is the **scaling factor** indicating how far to go in radial direction
  - $(\theta_i \phi_{h_i}/\pi)$ is the **rotation angle** restricted by **half aperture $\boldsymbol{\phi_{h_i}}$** of the cone that apexs at h_i

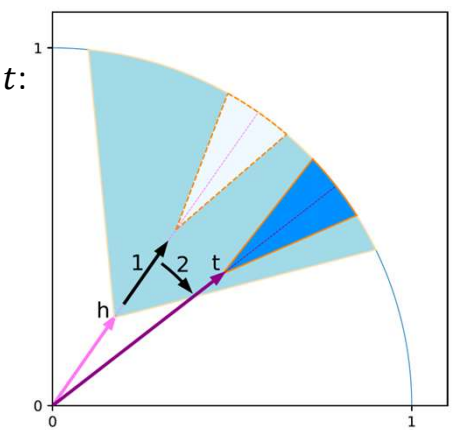
Half aperture:

$$\phi_{h_i} = \sin^{-1}(K \frac{1 - \|h_i\|^2}{\|h_i\|})$$

Cone Rotation from $\boldsymbol{h}$ to $\boldsymbol{t}$:



Restricted Rotation from h to t:
1: **scaling**
2: **rotation** (s_i, θ_i)



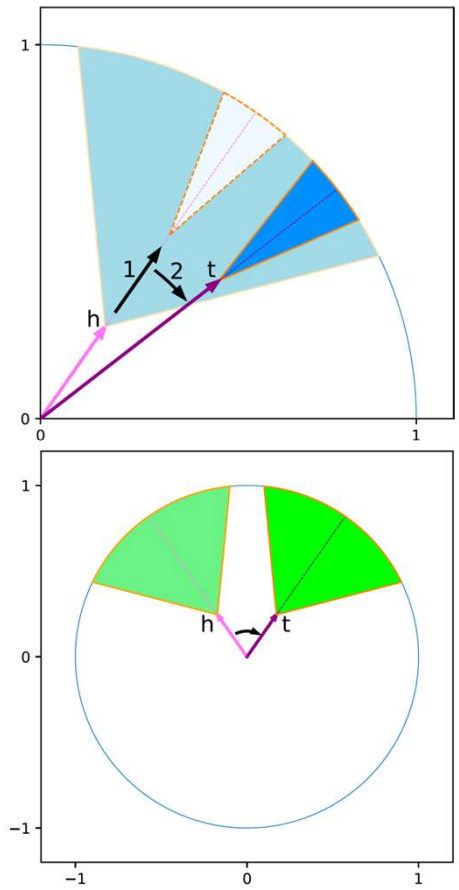
Embedding Subspaces

- For each relation: we assign a **subspace X** of hyperbolic space to capture hierarchical properties of the relation

Resulting cone is guaranteed to be contained by the original cone

- Use **restricted rotation** from the parent cone to the child cone in the **assigned relation-specific subspace X**

- For the complement subspace $E \backslash X$, we use a **general rotation** to model the relation.



Restricted rotation

Rotation

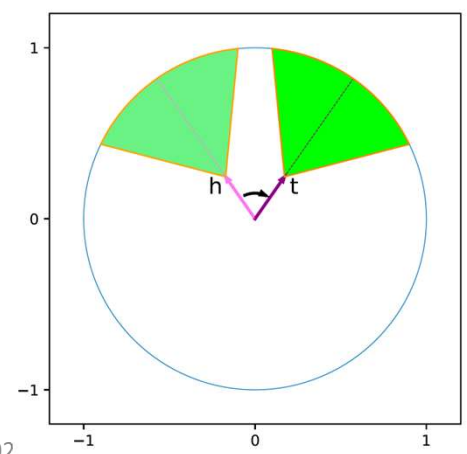
Transformation (1)

f_1 is a function of **head** and **relation**, and will be combined with **tail** t to compute the scores (for the i-th hyperbolic plane)

- **Non-hierarchical transformation**

$$f_1(\boldsymbol{h}_i, \boldsymbol{r}_i) = \exp_o(\boldsymbol{G}(\theta_i) \log_o(\boldsymbol{h}_i))$$

- $\boldsymbol{G}(\theta_i) = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{bmatrix}$
- $\exp_o$ and $\log_o$ are exponential and logrithmic mapping at origin o
- Apply rotation in the tangent space of origin o



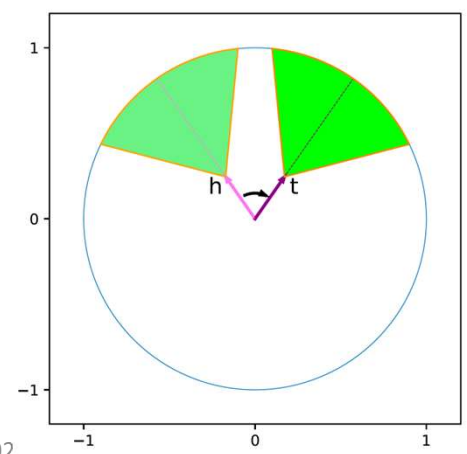
Cone rotation from h to t used for non-hierarchical relations

Transformation (1)

- **Non-hierarchical transformation**

$$f_1(\boldsymbol{h}_i, \boldsymbol{r}_i) = \exp_o(\boldsymbol{G}(\theta_i) \log_o(\boldsymbol{h}_i))$$

- $\boldsymbol{G}(\theta_i) = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{bmatrix}$
- $\exp_o$ and $\log_o$ are exponential and logrithmic mapping at origin o
- Apply rotation in the tangent space of origin o



f_1 cannot model hierarchical relations!
Rotation does not obey transitive property:
rotation by θ_i twice result in a rotation of $2\theta_i$
$r(h_1, h_3)$ and $r(h_1, h_2) \Rightarrow r(h_2, h_3)$ ✗

Use Restricted Rotation Transformation to model hierarchical relations!

Transformation (2)

- **Hierarchical transformation**

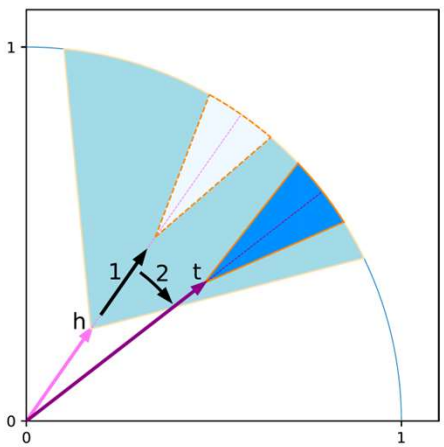
Restricted rotation

$$f_2(\boldsymbol{h}_i, \boldsymbol{r}_i) = \exp_{\boldsymbol{h}_i}(s_i \boldsymbol{G}\left(\frac{\theta_i \phi_{h_i}}{\pi}\right)\overline{\boldsymbol{h}_i}), \text{ where } \boldsymbol{r}_i = (s_i, \theta_i)$$

- Where $\overline{\boldsymbol{h}_i}$ is the unit vector of $\boldsymbol{h}_i$ in the tangent space of $\boldsymbol{h}_i$
- $\overline{\boldsymbol{h}_i} = \widehat{\boldsymbol{h}_i}/\|\widehat{\boldsymbol{h}_i}\|$, where $\widehat{\boldsymbol{h}_i} = \log_{\boldsymbol{h}_i}(\frac{1+\|\boldsymbol{h}_i\|}{2\|\boldsymbol{h}_i\|}\boldsymbol{h}_i)$

Half aperture:

$$\phi_{h_i} = \sin^{-1}(K\frac{1-\|h_i\|^2}{\|h_i\|})$$



Restricted Rotation from h to t for hierarchical relations:

1: **scaling** s_i

2: **rotation** θ_i

Scoring Function

- **Cone containment constraint** serves to preserve **partial ordering** of a hierarchical relations
- To enforce the cone containment constraint, the **distance scoring function** for triple (h, r, t) is defined as

Mask

Restricted rotation

$$f_r(h,t) = -\frac{1}{d}\left[\sum_{i=1}^{d} m(r)_i \left(d_{\mathbb{D}}\left(f_2(\boldsymbol{h}_i, \boldsymbol{r}_i), \boldsymbol{t}_i\right)\right)\right.$$

Hyperbolic distance

$$\left. + \sum_{i=1}^{d} \left(1 - m(r)_i\right) \left(d_{\mathbb{D}}\left(f_1(\boldsymbol{h}_i, \boldsymbol{r}_i), \boldsymbol{t}_i\right)\right) \quad + b_h + b_t\right]$$

Rotation

  - Where $d_{\mathbb{D}}$ is the distance in the hyperbolic plane (2D Poincaré model)
  - b_h and b_t are the learnt radius parameters of h and t
  - $m(r) \in \{0,1\}^d$ is a d-dimensional mask for each relation r. $m(r)_i = 1$ indicates restricted rotation is used for relation r in the i-th hyperbolic plane

Loss Function

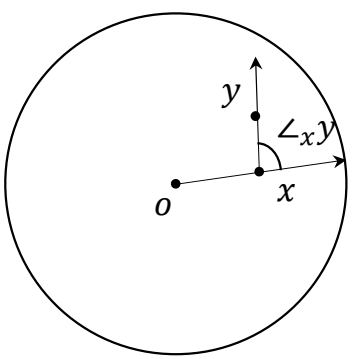
- **Distance Loss:** for a triple (h, r, t), we generate negative samples (h, r, t') by sdubstituting the tail with a random entity $t' \in \mathcal{T}$, $\mathcal{T}$ is the entity set.

Squashing

$$L_d(h, r, t) = -\log \sigma\big(f_r(h, t)\big) + \sum_{t' \in \mathcal{T}} \frac{1}{|\mathcal{T}|} \log \sigma\big(-f_r(h, t')\big)$$

Negative log likelihood

- **Angle Loss:** encourages cone of h to contain cone of t in relation-specific subspaces, by constraining the angle between the cones



$$L_a(h, r, t) = \sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{d}} m(r)_i \max\Big(0, \angle_{\boldsymbol{h}_i} \boldsymbol{t}_i - \phi(\boldsymbol{h}_i)\Big)$$

Half aperture:

$$\phi_{h_i} = \sin^{-1}\left(K \frac{1 - \|h_i\|^2}{\|h_i\|}\right)$$

- **The final loss:** $L = L_d + \boxed{w} L_a$

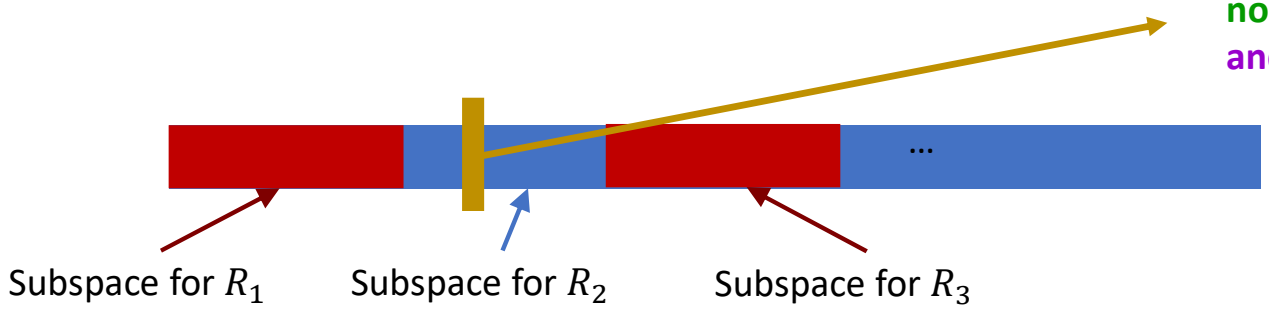
Hyper-parameter

Embedding Subspaces (1)

How to choose the subspace for each relation?

- **Option 1:** Sequential assignment (**Orthogonal**)

The same dimension can capture non-hierarchical property for R1, and hierarchical property for R2



Example:

Consider $S = B^2 \times B^2 \times B^2 \times B^2 \times B^2$(10-dimensional embedding space) and 5 hierarchical relations. We use dimension 1 to 2 for relation R_1, dimension 3 to 4 for relation R_2, etc.

- The subspace dimension can be at most $\frac{d}{n}$, where n is the number of hierarchical relations, d is the embedding dimension.

Hierarchical Reasoning

Example of Hierarchical Reasoning:

- B ***studiesAt*** A***;*** B ***classmateOf*** C; missing link between C and A
  - ***studiesAt:*** hierarchical relation, satisfies partial ordering
  - ***classmateOf:*** non-hierarchical relation

Query: is there a relationship between "C" and "A"?

- Consider the case where the embedding dimension $S = B^2$, we use **restricted rotation** to model hierarchical relation ***studiesAt*** in the single hyperbolic plane. At the same time, we use **general rotation** to model non-hierarchical relation ***classmateOf.***
- With a single hyperbolic plane, we can predict the missing relation between C and A: ***studiesAt.***

Embedding Subspaces (2)

- **Option 2:** Independent random sampling (**Overlapping**)
- Assigning **overlapping subspaces** is better for knowledge graph completion, and scalable to number of hierarchical relations

Randomly sampled dimensions for each relation's subspace (overlapping)

	MRR	H@1	H@3	H@10
Orthogonal	.493	.449	.512	.577
Overlapping	**.495**	**.451**	**.513**	**.582**

Comparison between orthogonal subspaces and overlapping subspaces for knowledge graph completion

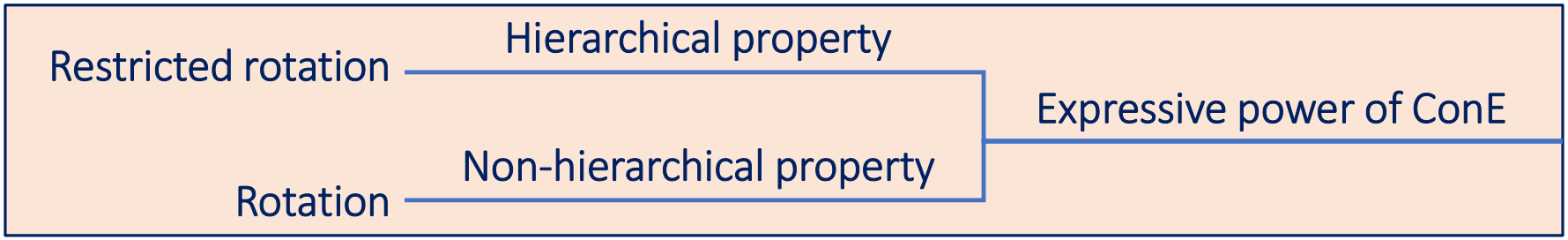
Experiments: Hierarchical Reasoning

- **Ancestor-descendant prediction**
  - **Is "Tree" and ancestor of "Wine Palm", and through which relation type?**
  - **Ancestor-descendant relationship**: If there exists a path from h_1 to h_2 containing only one type of hierarchical relation
  - **Inferred descendant pairs**: harder samples, there is missing edge in the path that require the model to infer

	WN18RR			DDB14			GO21		
	Fraction of inferred descendant pairs among all true descendant pairs in the test set								
Model	0%	50%	100%	0%	50%	100%	0%	50%	100%
Order [19]	.889	.739	.498	.731	.633	.513	.642	.592	.534
Poincaré [10]	.810	.685	.508	.976	.832	.571	.525	.519	.516
HypCone [12]	.799	.677	.504	.973	.823	.594	.554	.539	.519
RotatE [7]	.601	.593	.582	.615	.590	.565	.546	.534	.526
RotH [16]	.601	.608	.611	.609	.596	.578	.596	.583	.564
ConE	**.895**	**.801**	**.679**	**.981**	**.909**	**.818**	**.789**	**.744**	**.693**
Improvement (%)	+1.9%	+9.6%	+11.1%	+0.5%	+10.3%	+38.4%	+22.9%	+25.7%	+22.9%

Table 2: Ancestor-descendant prediction results in mAP (mean average precision). Best score in **bold** and second best underlined. We create different test sets that get harder as they contain more and more test cases (0%, 50%, 100%) of inferred descendant pairs.

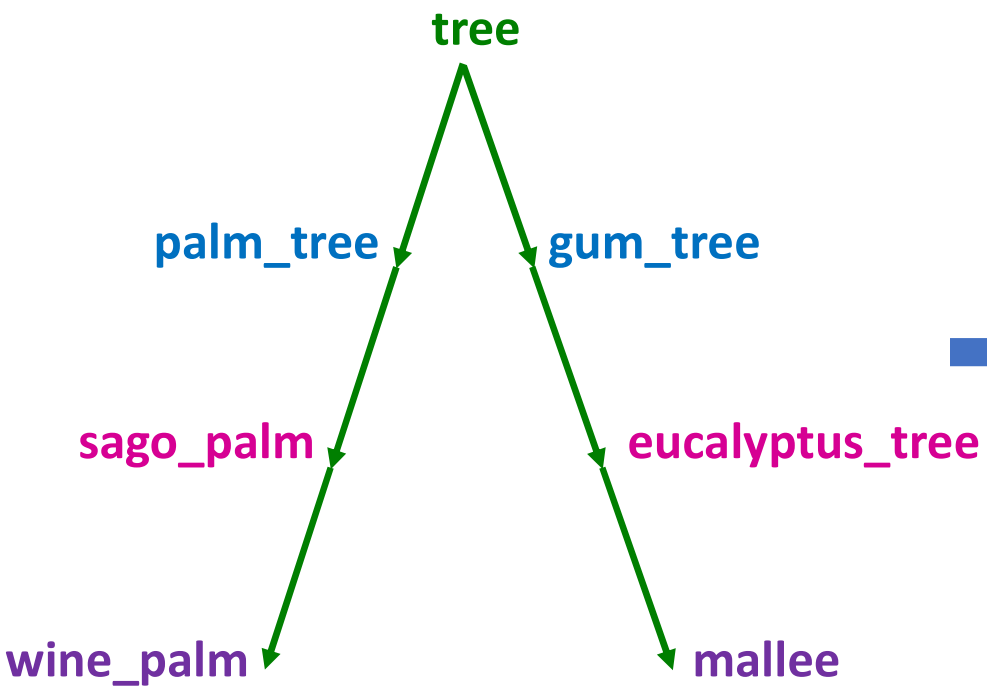
Experiments: Knowledge Graph Completion



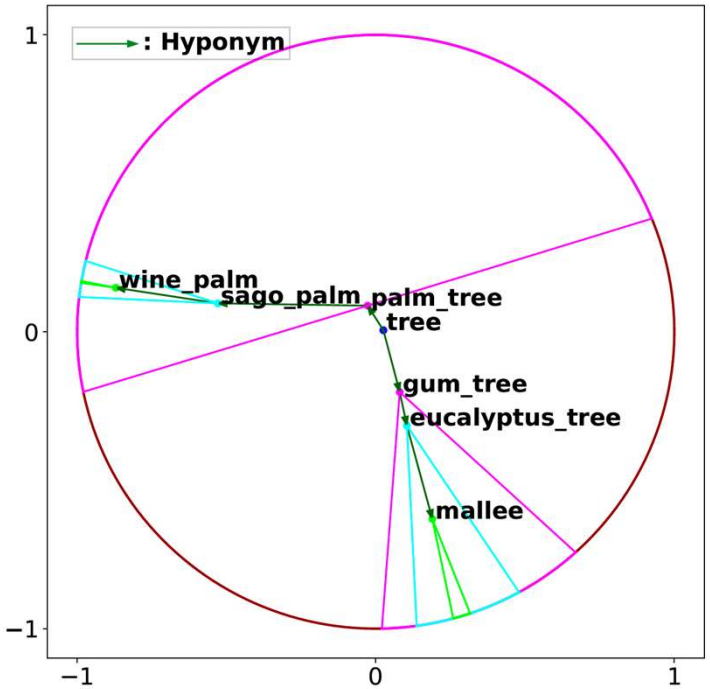
	WN18RR $\kappa = (1.00, 0.61, 0.99, 0.50)$				DDB14 $\kappa = (1.00, 0.84, 0.78, 0.18)$				GO21 $\kappa = (1.00, 0.65, 0.96, 0.22)$				FB15k-237 $\kappa = (1.00, 0.18, 0.36, 0.06)$			
Model	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
TransE [5]	.226	.017	.403	.532	.183	.103	.212	.337	.149	.066	.179	.310	.294	-	-	.465
RotatE [7]	.476	.428	.429	.571	.225	.154	.245	.362	.203	.123	.234	**.357**	.338	.241	.375	.533
TuckER [14]	.470	.443	.482	.526	.198	.137	.219	.314	.205	.136	.222	.342	**.358**	**.266**	**.394**	**.544**
HAKE [33]	.496	.451	.513	.582	.217	.146	.237	.361	.169	.104	.185	.295	.341	.243	.378	.535
MuRP [15]	.481	.440	.495	.566	.214	.146	.231	.349	.166	.100	.181	.301	.335	.243	.367	.518
RotH [16]	.495	.449	.514	**.586**	.223	.152	.245	.357	.151	.079	.171	.289	.344	.246	.380	.535
ConE	.496	**.453**	**.515**	.579	**.231**	**.161**	**.252**	**.364**	**.211**	**.140**	**.237**	.347	.345	.247	.381	.540

Knowledge Graph Completion results, best out of dimension $d \in \{100, 250, 500\}$. Best score in **bold** and second best underlined. κ is a tuple denoting the 4 Krackhardt scores that measure how hierarchical a graph is, higher scores mean more hierarchical.

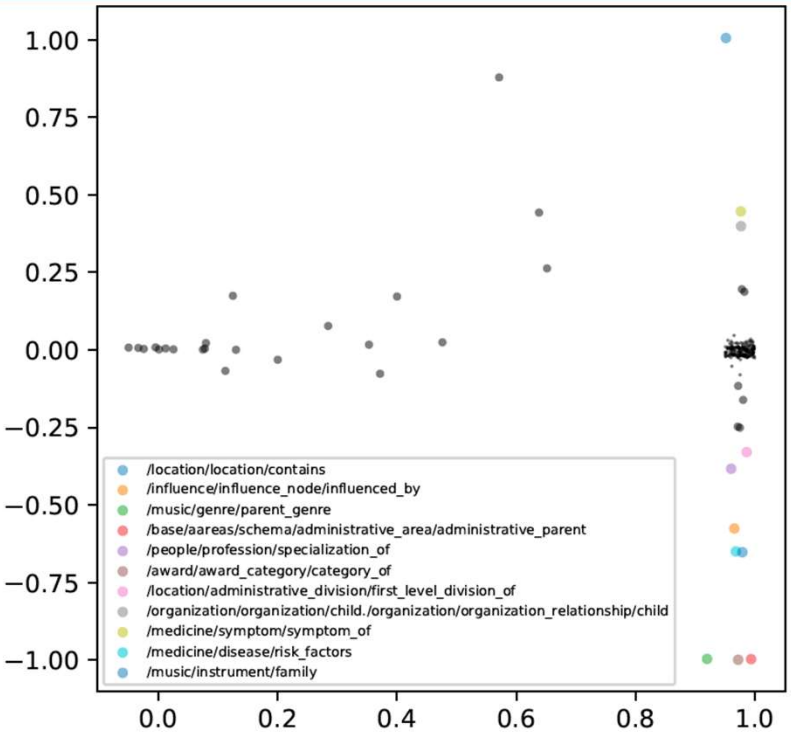
Cone Containment is Well-preserved



embed



Hierarchical-ness scores



Relation	Score	Hierarchical
administrative_area/administrative_parent	2.0	true
award_category/category_of	2.0	true
music/genre/parent_genre	2.0	true
location/contains	2.0	true
music/instrument/family	1.7	true
medicine/disease/risk_factors	1.7	unknown
influence/influence_node/influenced_by	1.6	unknown
medicine/symptom/symptom_of	1.4	unknown
organization_relationship/child	1.4	true
administrative_division/first_level_division_of	1.3	true
rest of the relations	< 1.1	false

Above the line (> 1.1) are predicted to be hierarchical relations

Ground-truth relation (manually labeled)

Hierarchical-ness Scores visualization

x: asymmetry, the same as *hierarchy* metric in Krackhardt scores

y: Tree_likeness. Adapted from the LUBedness metric in Krackhardt scores

Summary of Hyperbolic Embedding

- Hyperbolic space reflects **the hierarchical structure** when embedding biological sequences.
  - **Alignment-based and data-dependent** methods greatly **accelerate** large-scale analyses in bioinformatics.
  - NeuroSEED provides **fast approximations** of the distance, low-dimensional **hyperbolic** representations for biological sequences.
- **Hierarchical KG embedding: ConE** models **entities as hyperbolic cones** to simultaneously capture hierarchical and non-hierarchical relation patterns in heterogeneous knowledge graphs.
  - **Assign subspace** to each relation to tackle heterogeneity in hierarchies.
  - General rotation models non-hierarchical relations, while **restricted rotation** models partial ordering.