Poincaré Embeddings for Learning Hierarchical Representations

Geometric Methods in Machine Learning

Léa BRESSON

ENSAE ParisTech

Paris-Saclay University

2018

Contents

1 Introduction

With the explosion of text data, one of the biggest challenges in NLP is to enable computers to understand the meanings of words. In recent years, there has been an immense interest in word embeddings that is a method able to capture semantic similarities between words by representing discrete words in a continuous vector space.

Traditional methods learn and represent embeddings in a Euclidean space. As a consequence, embeddings are constrained by the linear property of the Euclidean space as very large dimensions are required to model complex relations. Today, there is a real need to increase the representation capacity of embedding methods.

In this line, Nickel and Kiela [1] proposed a method that computes embeddings not in Euclidean but in a hyperbolic space (more precisely a Poincaré unit ball) allowing to capture hierarchical properties of data that we can't capture directly in Euclidean space. Such embeddings are called Poincaré embeddings and are one of the latest trend in NLP. The aim of this report is to present and assess the empirical efficiency of such an approach. We focus on large datasets whose objects can be organized according to a latent hierarchy.

The present report is organized as follows. In **Section 2**, we briefly introduce the Poincaré embeddings and its properties. The algorithm to obtain Poincaré embedding is described in **Section 3**. In **Section 4**, we present the results of the authors' and our implementations. Finally, we discuss the method introduced by Nickel and Kiela in **Section 5**.

2 Beyond Euclidean embeddings

2.1 Motivation for hyperbolic geometry

One of the main challenges of representation learning is to preserve distance and more complex relationships when embedding a space into another. Traditional methods propose to embed objects into a **Euclidean space** where distances between two points are the same no matter how far one goes from the origin. The problem with Euclidean space is that it requires a large number of dimensions to reflect distances correctly leading to computational problems (runtime, memory complexity, overfitting). In this context, modeling a complex dataset, which can be represented with hierarchical structures such as trees, can become **computationally infeasible**.

Nickel and Kiela propose a method that exploits this structural property for learning more efficient embeddings. The main innovation is that these embeddings are learned in a **hyperbolic space**, as opposed to the commonly used Euclidean space. In the following, we assume that data exhibit a latent hierarchical structure in which the symbols can be organized and we do not have access to information about this hierarchy.

A hyperbolic space has a constant negative curvature allowing the distances to increase exponentially with respect to Euclidean distances and consequently, this space is more suitable for modeling **hierarchical data**. More precisely, hyperbolic spaces can be thought of as continuous versions of trees because hyperbolic disc area and circle length grow exponentially with their radius as does the distance from the root of the tree to its leaves (with every new child).

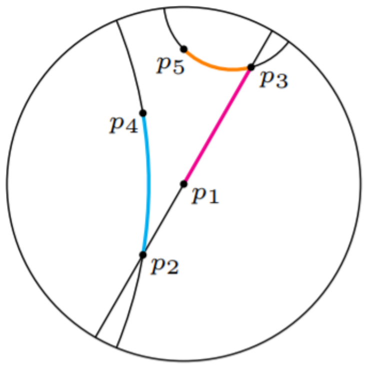


Figure 1: Geodesics of the Poincaré disk. The Figure is from [1].

Figure 1 represents a hyperbolic surface that has been projected onto a Euclidean surface. One can see that the distances of points increase exponentially as we move away from the center.

2.2 Poincaré ball model

Nickel and Kiela use a particular model of hyperbolic space, i.e. the **Poincare ball model**, since it is particularly suitable for gradient based optimization.

Let $\mathcal{B}^d = \Big\{x \in \mathbb{R}^d \mid ||x||< 1\Big\}$ the open d-dimensional unit ball. The hyperbolic distance between two points $u, v \in \mathcal{B}^d$ is given as

$$d(u,v) = \operatorname{arcosh}\Big(1 + 2\frac{||u-v||^2}{(1-||u||^2)(1-||v||^2)}\Big). \tag{1}$$

Equation 1 enables to learn embeddings that capture notions of both similarity (through the distance) and hierarchy (through the norm). More precisely:

- Similarity is encoded by the fact that connected nodes are close to each other and unconnected nodes are far.
- Hierarchical organization is determined by the distance to the origin: the root nodes are at the origin of $\mathcal{B}^d$ and the leaf nodes are placed close to the boundary of the Poincare ball (higher norm).

3 Learning Poincaré embeddings

Let $S = \{x_i\}_{i=1}^n$ be a set of symbols for which we want to compute Poincaré embeddings $\Theta = \{\theta_i\}_{i=1}^n$ where $\theta_i \in \mathcal{B}^d$. To estimate Θ, we solve the following optimization problem

$$\Theta' \leftarrow \operatorname{argmin}_\Theta\{\mathcal{L}(\Theta) \mid \theta_i \in \Theta : ||\theta_i||< 1\}$$

with $\mathcal{L}(\Theta)$ a **task-specific loss function**. Nickel and Kiela solve this problem using a stochastic gradient method appropriate for hyperbolic space.

3.1 Problem setting

For training the model, the authors define a loss function based on the hyperbolic distance defined in Equation 1: semantically similar objects are encouraged to be close in the embedding space. More precisely, Nickel and Kiela introduce two objective functions, depending on the task to be solved.

For embeddings taxonomies, which aims to embed words such that related words are close in the hyperbolic space, they minimize the following loss function with respect to Θ

$$\mathcal{L}(\Theta) = \sum_{(u,v)\in\mathcal{D}} \ln \frac{e^{-d(u,v)}}{\sum_{v'\in\mathcal{N}(u)} e^{-d(u,v')}} \tag{2}$$

where $\mathcal{N}(u) = \{v' | (u, v') \notin \mathcal{D}\} \cup \{v\}$ is the set of negative examples for u with $\mathcal{D} = \{(u, v)\}$ the set of all observed hypernymy relations and $\mathcal{N}(u)$ the set of negative examples for u.

For the graph link prediction, which aims to embed nodes in a given graph such that missing links are well constructed, they minimize the cross entropy of probability of having a link between co-authors

$$P((u, v) = 1|\Theta) = \frac{1}{e^{(d(u,v)-r)/t} + 1}. \tag{3}$$

A variant of SGD called **Stochastic Riemannian Gradient Descent** (SRGD) is carried out within the Poincaré ball for minimizing the loss functions.

3.2 Riemannian optimization

Stochastic Riemannian optimization techniques are easily parallelizable and scale to large datasets. The update for each embedding is made as follows:

$$\theta_{t+1} = \mathcal{R}_{\theta_t}\Big(- \eta_t \nabla_R \mathcal{L}(\theta_t)\Big)$$

with $\mathcal{R}_{\theta_t}$ the retraction onto $\mathcal{B}$ at θ_t and ∇_R the Riemannian gradient.

Which simplifies[1] to

$$\theta_{t+1} \leftarrow proj\Big(\theta_t - \eta_t \frac{(1 - ||\theta_t||^2)^2}{4} \nabla_E\Big). \tag{4}$$

With the unit sphere projection defined as $\text{proj}(\theta) = \begin{cases} \frac{\theta}{||\theta||} - \epsilon \text{ if } ||\theta|| \geq 1 \\ \theta \text{ otherwise} \end{cases}$ so that the embeddings remain within the Poincaré ball after update.

As a result, the computational and memory complexity of an update depends linearly on the embedding dimension.

[1]Details are given in the paper (page 4) to compute the Riemannian gradient from the Euclidean one.

For training, the embeddings are initialized randomly from the uniform distribution $\mathcal{U}(-10^{-3}, 10^{-3}))$, i.e. close to the origin of $\mathcal{B}^d$. In addition, the model is trained with a progressively increasing learning rate ("burn-in" phase during 10 epochs with reduced training rate) in order to reduce the impact of a bad random initialization.

4 Experimental results

4.1 Authors' experiments

Nickel and Kiela evaluate the quality of Poincaré embeddings on three applications: *(i)* embedding taxonomies, *(ii)* embedding a network and *(iii)* embedding lexical entailment. To do so, the Poincaré distance is compared with Euclidean distance - $d(u, v) = ||u\breve{}v||^2$ - and, for asymmetric data, Translational distance - $d(u, v) = ||u - v + r||^2$ - (where r is a learnable translation vector). Experimental results show that Poincaré distance outperforms both baseline distances significantly for each tasks.

More precisely, for embedding taxonomies (see Equation 2), the authors use the *WordNet* dataset (data that exhibit a clear latent hierarchical structure) and consider two tasks:

- Reconstruction to assess the **representation capacity** (measured by the Mean Average Precision of the ranked reconstructed list),
- Link prediction to assess the **generalization performance** (the dataset is randomly split into training and test sets; the performance is measured by the rank of each observed relationship among the ground truth negative examples which were not in the training set).

The experimental results indicate that Poincaré embeddings are the most efficient for both tasks, even compared with translational embeddings which have more information about the structure of the data. In addition, Nickel and Kiela point out that Poincaré embeddings enable **parsimonious representations**: the representation given by the 5 dimensional space is better than the result of the Euclidean model in a 200 dimensional space! Finally, the excellent link prediction results show that Poincaré embeddings are **robust** with regard to the embedding dimension (this illustrates the fact that hyperbolic geometry can introduce an important structural bias). Results are given in Appendix A.

For Network embedding (see Equation 3), Nickel and Kiela use 4 commonly used social network representing scientific collaborations (*ASTROPH*, *CONDMAT*, *GRQC*, and *HEPPH*) on the reconstruction and the link prediction tasks. Again, Poincaré embeddings outperform Euclidean embeddings. Results are given in Appendix B.

Finally, the authors test the ability of Poincaré embeddings to make graded assertions about hierarchical relationships on the dataset *HYPERLEX*. Using Spearman's rank correlation, they obtain that the ranking based on Poincaré embeddings clearly outperforms all state of the art results for predicting lexical entailment. Results are given in Appendix C.

4.2 My implementation

I implemented the Poincaré embeddings method on Python focusing on Taxonomies Embeddings (mammal dataset from WordNet.). Since *(i)* the official implementation of Poincaré embeddings[2] is done using Pytorch and *(ii)* the Gensim implementation uses Numpy[3], I decided to use **Keras**.

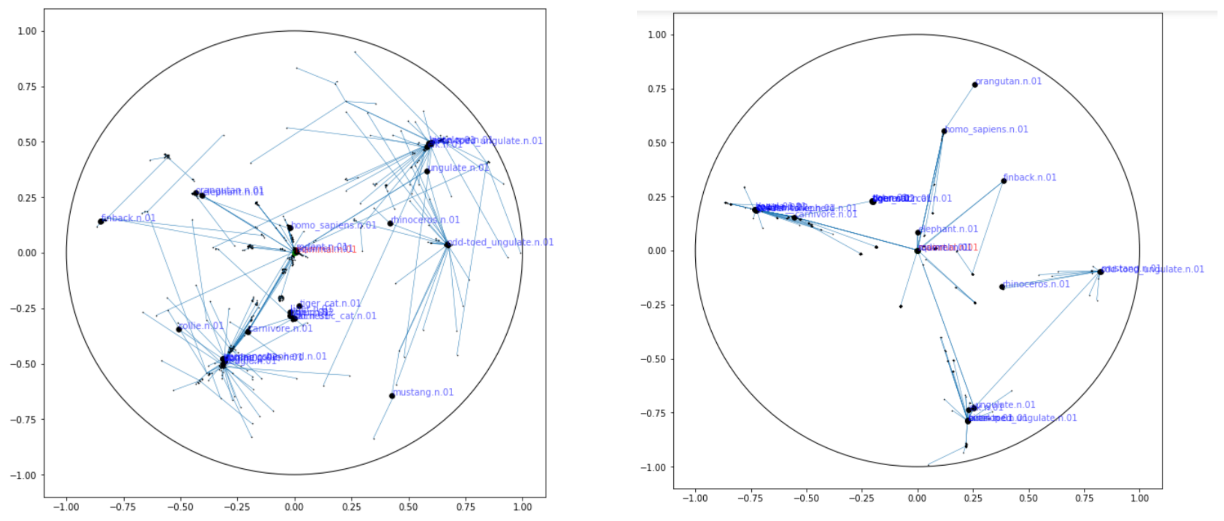
I focus on WordNet noun hypernym paires, which is the simplest hierarchical relationship to model: the Mammal dataset consists of 82,115 nouns and 6542 hypernymy relations. Here u is a target word, v is any hypernym for that word, v' is a set of randomly sampled words. As Nickel and Kiela, I randomly sampled 10 negative examples per positive example.

I made slight modifications in order to improve the algorithm based on the recommendations of the *RaRe Technologies* blog post[4]. For example, I implemented batch-wise training to speed up the training process I also introduced an option to apply (or not) L2 regularization to the embeddings. More details about the implementation are in the notebook.

I also implemented different optimization methods:

- the one introduced by the authors (customized SGD with update rule defined in Equation 4 and a burn-in period), with or without momentum[5],
- the traditional SGD, with L2 regularization or with momentum,
- the Adam algorithm[6], with and without regularization.

[2]https://github.com/facebookresearch/poincare-embeddings.
[3]https://github.com/RaRe-Technologies/gensim/blob/develop/gensim/models/poincare.py.
[4]https://rare-technologies.com/implementing-poincare-embeddings.
[5]Momentum helps accelerate SGD in the relevant direction and limits overfitting.
[6]Adam computes adaptive learning rates for each parameter.

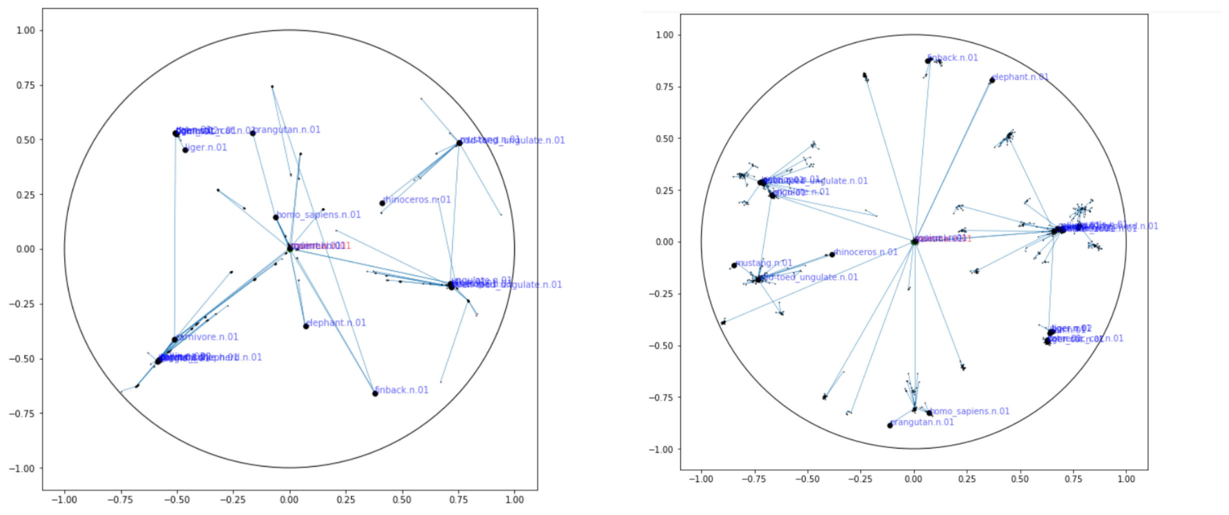


Without Momentum

With Momentum

Figure 2: Two-dimensional Poincaré embeddings (Riemannian optimization) - 100 iterations

I don't obtain the same results as the authors (see Appendix D), however, the learned Poincaré embeddings capture notions of **similarity and hierarchy**. Similar animals in the WordNet are close to each other in the embedding space. In addition, the nodes close to the boundary are lower in the hierarchy[7].



SGD with regularization

Adam (without regularization)

Figure 3: Two-dimensional Poincaré embeddings (alternative optimizer) - 50 iterations

[7]Note: the code for the plots is taken from the C++ implementation of the Poincarré embeddings (https://github.com/TatsuyaShirakawa/poincare-embedding) and from https://github.com/fornaxai/WDYL/blob/bd33698e60db762d315ead57215884809eec4dce/2017.09.05_Poincare_Embeddi

Surprisingly, I obtain similar results with SGD or Adam with less iterations (50 against 100 for Riemannian) and the same learning rate suggesting that a full Riemannian optimization is not necessary and traditional approaches, which are well documented and implemented, are relevant. In addition, we can achieve better results with a better choice of hyperparameters (the learning rate is fixed at 0.001 here). Below a nicer representation. More figures can be found in the notebook.

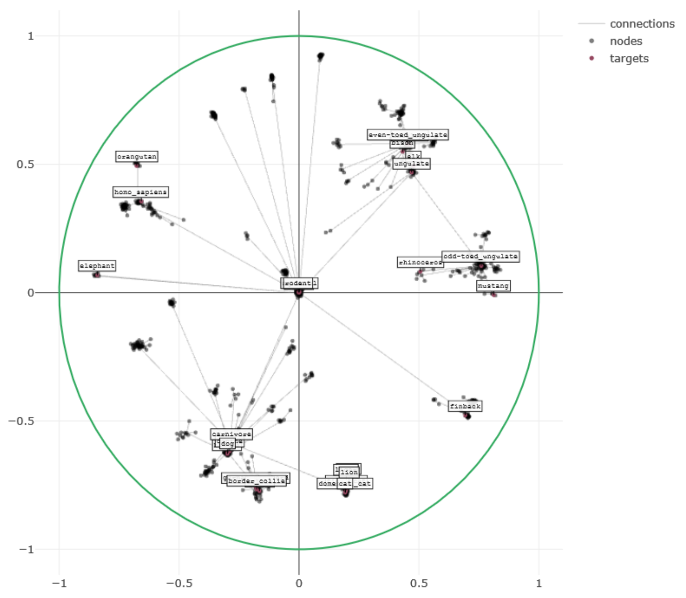


Figure 4: Two-dimensional Poincaré embeddings (with Adam)

I also implemented 3d-Poincaré embeddings:

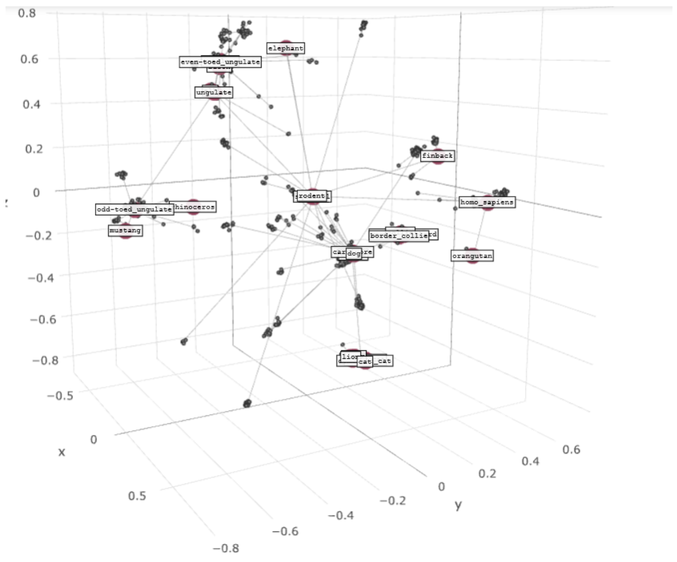


Figure 5: Three-dimensional Poincaré embeddings of transitive closure of the WordNet mammals (with Adam optimization algorithm and 50 iterations)

5 Conclusion and Discussion

To conclude learning embeddings in a hyperbolic plane, rather than Euclidan space, allows to capture not only semantic, but also hierarchical structure of the data. This improves the quality of the learned representations while also reducing the number of dimensions needed to obtain a good representation.
My implementations suggest that traditional optimization approaches such as SGD or Adam converge faster and give similar results as a full Riemannian optimization approach.

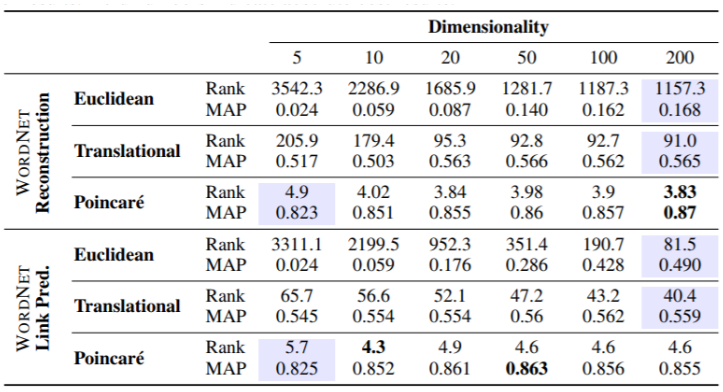
One of the limitation of the presented methods is that baseline distances may achieve better results than the ones depicted in the paper. Indeed, for each task, the authors define a loss (or a probability) function (see Equations 2 and 3) that could be advantageous to the Poincaré distance and disadvantageous to the baselines.

Also, one could ask: what happens if the hierarchy is less pronounced than in the dataset used by the authors ?

Bibliography

[1] Maximillian Nickel and Douwe Kiela. “Poincaré Embeddings for Learning Hierarchical Representations”. In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon et al. Curran Associates, Inc., 2017, pp. 6338–6347. URL: `http://papers.nips.cc/paper/7213-poincare-embeddings-for-learning-hierarchical-representations.pdf`.

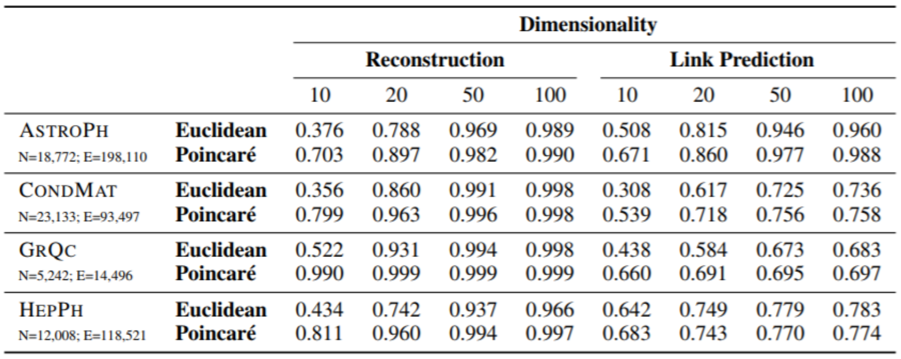
A Appendix A.



			Dimensionality					
			5	10	20	50	100	200
WordNet Reconstruction	Euclidean	Rank	3542.3	2286.9	1685.9	1281.7	1187.3	1157.3
		MAP	0.024	0.059	0.087	0.140	0.162	0.168
	Translational	Rank	205.9	179.4	95.3	92.8	92.7	91.0
		MAP	0.517	0.503	0.563	0.566	0.562	0.565
	Poincaré	Rank	4.9	4.02	3.84	3.98	3.9	**3.83**
		MAP	0.823	0.851	0.855	0.86	0.857	**0.87**
WordNet Link Pred.	Euclidean	Rank	3311.1	2199.5	952.3	351.4	190.7	81.5
		MAP	0.024	0.059	0.176	0.286	0.428	0.490
	Translational	Rank	65.7	56.6	52.1	47.2	43.2	40.4
		MAP	0.545	0.554	0.554	0.56	0.562	0.559
	Poincaré	Rank	5.7	**4.3**	4.9	4.6	4.6	4.6
		MAP	0.825	0.852	0.861	**0.863**	0.856	0.855

Experimental results on the transitive closure of the WordNet noun hierarchy. The Figure is from [1].

B Appendix B.

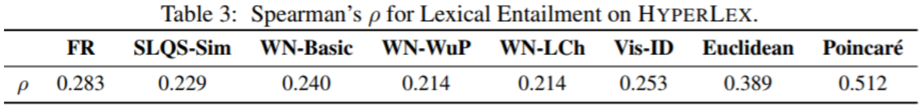


		Dimensionality							
		Reconstruction				Link Prediction			
		10	20	50	100	10	20	50	100
AstroPh N=18,772; E=198,110	Euclidean	0.376	0.788	0.969	0.989	0.508	0.815	0.946	0.960
	Poincaré	0.703	0.897	0.982	0.990	0.671	0.860	0.977	0.988
CondMat N=23,133; E=93,497	Euclidean	0.356	0.860	0.991	0.998	0.308	0.617	0.725	0.736
	Poincaré	0.799	0.963	0.996	0.998	0.539	0.718	0.756	0.758
GrQc N=5,242; E=14,496	Euclidean	0.522	0.931	0.994	0.998	0.438	0.584	0.673	0.683
	Poincaré	0.990	0.999	0.999	0.999	0.660	0.691	0.695	0.697
HepPh N=12,008; E=118,521	Euclidean	0.434	0.742	0.937	0.966	0.642	0.749	0.779	0.783
	Poincaré	0.811	0.960	0.994	0.997	0.683	0.743	0.770	0.774

Mean average precision on network data. The Figure is from [1].

C Appendix C.

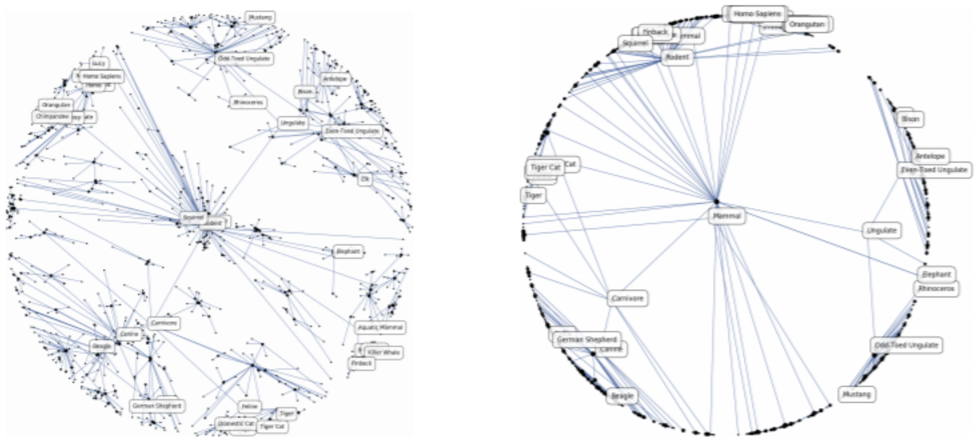
Table 3: Spearman's ρ for Lexical Entailment on HyperLex.



	FR	SLQS-Sim	WN-Basic	WN-WuP	WN-LCh	Vis-ID	Euclidean	Poincaré
ρ	0.283	0.229	0.240	0.214	0.214	0.253	0.389	0.512

Spearman's ρ for Lexical Entailment on HYPERLEX. The Figure is from [1].

D Appendix D.



(a) Intermediate embedding after 20 epochs (b) Embedding after convergence

Two-dimensional Poincaré embeddings of transitive closure of the WordNet mammals subtree. The Figure is from [1].