arXiv:1805.09786v1 [cs.NE] 24 May 2018

Hyperbolic Attention Networks

Caglar Gulcehre Misha Denil Mateusz Malinowski Ali Razavi
Razvan Pascanu Karl Moritz Hermann Peter Battaglia Victor Bapst
David Raposo Adam Santoro Nando de Freitas
caglarg@google.com
Deepmind

Abstract

We introduce hyperbolic attention networks to endow neural networks with enough capacity to match the complexity of data with hierarchical and power-law structure. A few recent approaches have successfully demonstrated the benefits of imposing hyperbolic geometry on the parameters of shallow networks. We extend this line of work by imposing hyperbolic geometry on the activations of neural networks. This allows us to exploit hyperbolic geometry to reason about embeddings produced by deep networks. We achieve this by re-expressing the ubiquitous mechanism of soft attention in terms of operations defined for hyperboloid and Klein models. Our method shows improvements in terms of generalization on neural machine translation, learning on graphs and visual question answering tasks while keeping the neural representations compact.

1 Introduction

The focus of this work is to endow neural network representations with suitable geometry to capture fundamental properties of data, including hierarchy and clustering behaviour. These properties emerge in many real-world scenarios that approximately follow power-law distributions [27, 9]. This includes a wide variety of natural phenomena in physics [22], biology [25], and even human-made structures such as metabolic-mass relationships [5], social networks [20, 30], and frequencies of words [32, 31, 37].

Complex networks [20], which connect distinguishable heterogeneous sets of elements represented as nodes, provide us with an intuitive way of understanding these structures. They will also serve as our starting point for introducing hyperbolic geometry, which is by itself difficult to visualize. Nodes in complex networks are referred to as heterogeneous, in the sense that they can be divided into sub-nodes which are themselves distinguishable from each other. The scale-free structure of natural data manifests itself as a power law distribution on the node degrees of the complex network that describes it.

Complex networks can be approximated with tree-like structures, such as taxonomies and dendrograms. Let us begin by recalling a simple property of n-ary trees that will help us understand hyperbolic space and why Euclidean geometry is perhaps inadequate to model relational data.

In an n-ary tree, the number of nodes at distance r from the root and the number of nodes at distance no more than r from the root both grow as n^r. Just like trees, the volume of hyperbolic space also expands exponentially. For example, in a two-dimensional hyperbolic space with curvature $-\zeta^2, \zeta > 0$, the length and area of the disc of radius r grow as $2\pi \sinh(\zeta r)$ and $2\pi(\cosh(\zeta r) - 1)$, respectively, both of which grow exponentially in r [20, 19].

The growth of volume in hyperbolic space should be contrasted with Euclidean space where the corresponding quantities expand only polynomially, length as $2\pi r$ and area as πr^2 in the two-dimensional example. Figure 1 is an attempt at visualizing this. For hierarchical data with scale-free

Preprint. Work in progress.

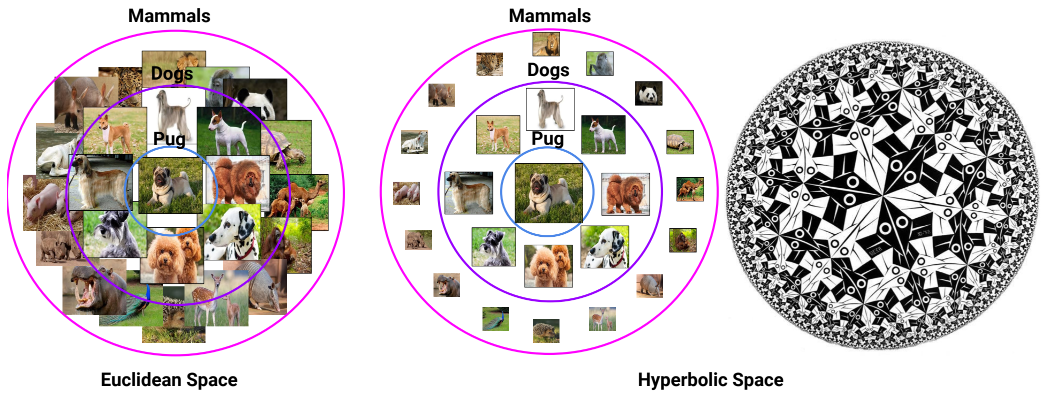

Figure 1: An intuitive depiction of how images might be embedded in 2D. The location of the embeddings reflects the similarity between each image and that of a pug. Since the number of instances within a given semantic distance from the central object grows exponentially, the Euclidean space is not able to compactly represent such structure (left). In hyperbolic space (right) the volume grows exponentially, allowing for sufficient room to embed the images. For visualization, we have shrunk the images in this Euclidean diagram, a trick also used by Escher.

structure, the polynomially expanding Euclidean space cannot capture the exponential complexity in the data (and so the images overlap). On the other hand, the exponentially expanding hyperbolic space is able to match the complexity of the data (here we have used the visualization trick of the famous artist Escher of making the images progressively smaller toward the boundary to illustrate the exponential expansion).

The intimate connection between hyperbolic space scale free networks (where node degree follows a power law) is made more precise in Krioukov et al. [20]. In particular, there it is shown that the heterogeneous topology implies hyperbolic geometry, and conversely hyperbolic geometry yields heterogeneous topology.

Moreover, Sarkar [35] describes a construction that embeds trees in two-dimensional hyperbolic space with arbitrarily low distortion, which is not possible in Euclidean space of any dimension [23]. Following this exciting line of research, recently the machine learning community has gained interest in learning non-Euclidean embeddings directly from data [28, 8, 33, 29, 38, 6].

Fuelled by the desire of increasing the capacity of neural networks so as to match the complexity of data, we propose *hyperbolic attention networks*. As opposed to previous approaches, which impose hyperbolic geometry on the parameters of shallow networks [28, 8], we impose hyperbolic geometry on their activations. This allows us to exploit hyperbolic geometry to reason about embeddings produced by *deep* networks. We achieve this by introducing efficient hyperbolic operations to express the popular, ubiquitous mechanism of attention [2, 11, 41, 46]. Our method shows improvements in terms of generalization on neural machine translation [41], learning on graphs and visual question answering [1, 24, 15] tasks while keeping the representations compact.

2 Models of hyperbolic space

Hyperbolic space cannot be isometrically embedded into Euclidean space [20]. There are several ways to endow different *subsets* of Euclidean space with a hyperbolic metric, leading to different *models* of hyperbolic space. This leads to the well known Poincaré ball model [14] and many others.

The different models of hyperbolic space are all necessarily the same, but different models define different coordinate systems, which offer different affordances for computation. In this paper, we primarily make use of the hyperboloid, whose status as the only commonly used unbounded model makes it a convenient target for projecting into hyperbolic space. We also make use of the Klein model, because it admits an efficient expression for the hyperbolic aggregation operation we define in Section 4.2.

We briefly review the definitions of the hyperboloid and Klein models and the relationship between them, in just enough detail to support the presentation in the remainder of the paper. A more thorough treatment can be found in Iversen [14]. The geometric relationship between the the two models is diagrammed in Figure 5 of the supplementary material.

Hyperboloid model: This model of n dimensional hyperbolic space is a manifold in the $n+1$ dimensional Minkowski space. The Minkowski space is $\mathbb{R}^{n+1}$ endowed with the indefinite Minkowski bilinear form

$$\langle \mathbf{q}, \mathbf{k} \rangle_M = \sum_{i=1}^{n} q_i k_i - q_{n+1} k_{n+1} \ .$$

With this definition in had, the hyperboloid model consists of the set

$$\mathbb{H}^n = \{\mathbf{x} \in \mathbb{R}^{n+1} \mid \langle \mathbf{x}, \mathbf{x} \rangle_M = -1,\ x_{n+1} > 0\}$$

endowed with the distance metric $d_{\mathbb{H}}(\mathbf{q},\ \mathbf{k}) = \operatorname{arccosh}(-\langle \mathbf{q},\ \mathbf{k} \rangle_M)$.

Klein model: This model of hyperbolic space is a subset of $\mathbb{R}^n$ given by $\mathbb{K}^n = \{\mathbf{x} \in \mathbb{R}^n \mid \|\mathbf{x}\| < 1\}$, and a point in the Klein model can be obtained from the corresponding point in the hyperboloid model by projection

$$\pi_{\mathbb{H}\to\mathbb{K}}(\mathbf{x})_i = \frac{x_i}{x_{n+1}} \ ,$$

with its inverse given by

$$\pi_{\mathbb{K}\to\mathbb{H}}(\mathbf{x}) = \frac{1}{\sqrt{1 - \|\mathbf{x}\|}}(\mathbf{x},\ 1) \ .$$

Distance computations in the Klein model can be inherited from the hyperboloid, in the sense that $d_{\mathbb{K}}(\mathbf{q},\ \mathbf{k}) = d_{\mathbb{H}}(\pi_{\mathbb{K}\to\mathbb{H}}(\mathbf{k}),\ \pi_{\mathbb{K}\to\mathbb{H}}(\mathbf{q}))$.

3 Attention as a building block for relational reasoning

Learning relations in a graph by using neural networks to model the interactions or relations has shown promising results in visual question answering [34], modelling physical dynamics [3], and reasoning over graphs [21, 43, 16, 18]. Graph neural networks [21, 3] incorporate a message passing as part of the architecture in order to capture the intrinsic relations between entities. Graph convolution networks [7, 17, 10] use convolutions to efficiently learn a continuous-space representation for a graph of interest.

Many of these relational reasoning models can be expressed in terms of an attentive read operation. In the following we give a general description of the attentive read, and then discuss its specific instantiations in two relational reasoning models from the literature.

3.1 Attentive read

First introduced for translation in Bahdanau et al. [2], attention has seen widespread use in deep learning, not only for applications in NLP but also for image processing [46] imitation learning [11] and memory [13]. The core computation is the *attentive read* operation, which has the following form:

$$\mathbf{r}(\mathbf{q}_i,\ \{\mathbf{k}_j\}_j) = \frac{1}{Z}\sum_j \alpha(\mathbf{q}_i,\ \mathbf{k}_j)\mathbf{v}_{ij} \ . \tag{1}$$

Here $\mathbf{q}_i$ is a vector called the *query* and the $\mathbf{k}_j$ are *keys* for the memory locations being read from. The pairwise function $\alpha(\cdot,\cdot)$ computes a scalar matching score between a query and a key, and the vector $\mathbf{v}_{ij}$ is a *value* to be read from location j by query i. $Z > 0$ is a normalization factor for the full sum. Both $\mathbf{v}_{ij}$ and Z are free to depend on arbitrary information, but we leave any dependencies here implicit.

It will be useful in the discussion to break this operation down into two parts. The first is the **matching**, which computes attention weights $\alpha_{ij} = \alpha(\mathbf{q}_i, \mathbf{k}_j)$ and the second is the **aggregation**, which takes a weighted average of the values using these weights,

$$m_i(\{\alpha_{ij}\}_j,\ \{\mathbf{v}_{ij}\}_j) = \frac{1}{Z}\sum_j \alpha_{ij}\mathbf{v}_{ij} \ .$$

Instantiating a particular attentive read operation involves specifying both $\alpha(\cdot,\cdot)$ and $\mathbf{v}_{ij}$ along with the normalization constant Z.

If one performs an attentive read for each element of the set j then the resulting operation corresponds in a natural way to message passing on a graph, where each node i aggregates messages $\{\mathbf{v}_{ij}\}_j$ from its neighbours along edges of weight $\alpha(\mathbf{q}_i, \mathbf{k}_j)/Z$.

We can express many (although not all) message passing neural network architectures [12] using the attentive read operation of Equation 1 as a primitive. In the following sections we do this for two architectures and then discuss how we can replace both the matching and aggregation steps with versions that leverage hyperbolic geometry.

3.2 Relation networks

Relation Networks (RNs) [34] are a neural network architecture designed for reasoning about the relationships between objects. An RN operates on a set of objects O by applying a shared operator to each pair of objects $(\mathbf{o}_i,\ \mathbf{o}_j) \in O \times O$. The pairs can be augmented by a global information, and the result of each relational operation is passed through a further global transformation.

Using the notation of the previous section, we can write the RN as

$$RN(O,\ \mathbf{c}) = f\left(\sum_i \mathbf{r}(\mathbf{o}_i,\ \{\mathbf{o}_j\}_j)\right) \ ,$$

where $\alpha(\mathbf{o}_i,\ \mathbf{o}_j) = 1$, $\mathbf{v}_{ij} = g(\mathbf{o}_i,\ \mathbf{o}_j,\ \mathbf{c})$, $Z = 1$. f is the global transformation g is the global transformation, the local transformation and $\mathbf{c}$ is the global context, as described in Santoro et al. [34]. We augment the basic RN to allow $\alpha(\mathbf{o}_i, \mathbf{o}_j) \in [0, 1]$ to be a general learnable function.

Interpreting the RN as learned message passing on a graph over objects, the attention weights take on the semantics of edge weights, where α_{ij} can be thought of as the probability of the the (directed) edge $\mathbf{o}_j \to \mathbf{o}_i$ appearing in the underlying reasoning graph.

3.3 Scaled dot-product attention

In the Transformer model of Vaswani et al. [41] the authors define an all-to-all message passing operation on a set of vectors which they call scaled dot-product attention. In the language of Section 3.1 the scaled dot-product attention operation performs several attentive reads in parallel, one for each element of the input set.

Vaswani et al. [41] write scaled dot-product attention as $\mathbf{R} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right)\mathbf{V}$, where $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$ are referred to as the *queries*, *keys*, and *values* respectively, and d is the shared dimensionality of the queries and keys. Using lowercase letters to denote rows of the corresponding matrices, we can write each row of $\mathbf{R}$ as the result of an attentive read with

$$\alpha(\mathbf{q}_i, \mathbf{k}_j) = \exp\left(\frac{1}{\sqrt{d}}\langle \mathbf{q}_i, \mathbf{k}_j\rangle\right) \ , \qquad \mathbf{v}_{ij} = \mathbf{v}_j \ , \qquad Z = \sum_j \alpha(\mathbf{q}_i, \mathbf{k}_j) \ .$$

We experiment with both softmax and sigmoid operations for computing the attention weights in our hyperbolic models. The motivation for considering sigmoid attention weights is that in some applications (e.g. visual question answering) it makes sense for the attention weights over different entities to not compete with each other.

3.4 The path forward

At this point, we have discussed how many natural hierarchies posses scale-free structures, and also how the geometry of hyperbolic space naturally encodes this structure. We have considered two models of hyperbolic space (hyperboloid and Klein), and have seen how points in the two models correspond.

We described the attentive read operation, and how it can be broken into two steps which we call matching and aggregation. We then showed how the relational operations from Relation Networks and the Transformer could be expressed using the attentive read as a primitive.

The following section is devoted to explaining how we can redefine the attentive read as an operation on points in hyperbolic space. This will allow us to create hyperbolic versions of Relation Networks and the Transformer by replacing their attentive read with our hyperbolic version.

4 Hyperbolic attention networks

In this section we show how to redefine the attentive read operation of Section 3.1 as an operation on points in hyperbolic space. The key to doing this is to define new matching and aggregation functions that operate on hyperbolic points and take advantage of the metric structure of the manifold they live on. However, in order to apply these operations inside of a network we first we need a way to interpret network activations as points in hyperbolic space.

We describe how to map an arbitrary point in $\mathbb{R}^n$ onto the hyperboloid, where we can interpret the result as a point in hyperbolic space. The choice of mapping is important since we must ensure that the rapid scaling behavior of hyperbolic space is maintained. Armed with an appropriate mapping we proceed to describe the hyperbolic matching and aggregation operations that operate on these points.

4.1 Hyperbolic network activations

Mapping neural network activations into hyperbolic space requires care, since network activations might live anywhere in $\mathbb{R}^n$, but hyperbolic structure can only be imposed on special subsets of Euclidean space [20]. This means we need a way to map activations into an appropriate manifold. We choose to map into the hyperboloid, which is convenient since it is the only unbounded model of hyperbolic space in common use.

Pseudo-polar coordinates: In polar coordinates, we express an n-dimensional point as a scalar radius, and $n-1$ angles. Pseudo-polar coordinates consist of a radius r, as in ordinary polar coordinates, and an n-dimensional vector $\mathbf{d}$ representing the direction of the point from the origin. In the following discussion we assume that the coordinates are normalized, i.e. that $\|\mathbf{d}\| = 1$.

If $(\mathbf{d},\ r) \in \mathbb{R}^{n+1}$ are the activations of a layer in the network, we map them onto the hyperbolid in $\mathbb{R}^{n+1}$ using $\pi((\mathbf{d},\ r)) = (\sinh(r)\mathbf{d},\ \cosh(r))$, which increases the scale by an exponential factor.

It is easily verified that the resulting point lies in the hyperboloid,

and to verify that we maintain the appropriate scaling properties we compute the distance between a point and the origin using this projection:

$$d_{\mathbb{H}}(\mathbf{0},\ (\mathbf{d},\ r)) = \operatorname{arccosh}(-\langle\pi(\mathbf{0}),\ \pi((\mathbf{d},\ r))\rangle_M) = \operatorname{arccosh}(\frac{1}{2}(\cosh(2r)\ +\ 1))\ \sim r\ ,$$

which shows that this projection preserves exponential growth in volume for a linear increase in r. Without the exponential scaling factor the effective distance of $\pi((\mathbf{d},\ r))$ from the origin grows logarithmically in hyperbolic space.[1]

4.2 Hyperbolic attention

In this section, we show how to build an attentive read operation that operates on points in hyperbolic space. We consider how to exploit hyperbolic geometry in both the *matching* and the *aggregation* steps of the attentive read operation separately.

Hyperbolic matching: The most natural way to exploit hyperbolic geometry for matching pairs of points is to use the hyperbolic distance between them. Given a query $\mathbf{q}_i$ and a key $\mathbf{k}_j$ both lying in

[1]Alternatively we can treat $\mathbf{d}$ as a vector in the tangent space of $\mathbb{H}^n$ at the origin defining a geodesic and compute distances between points using the law of cosines, this leads to similar scaling properties.

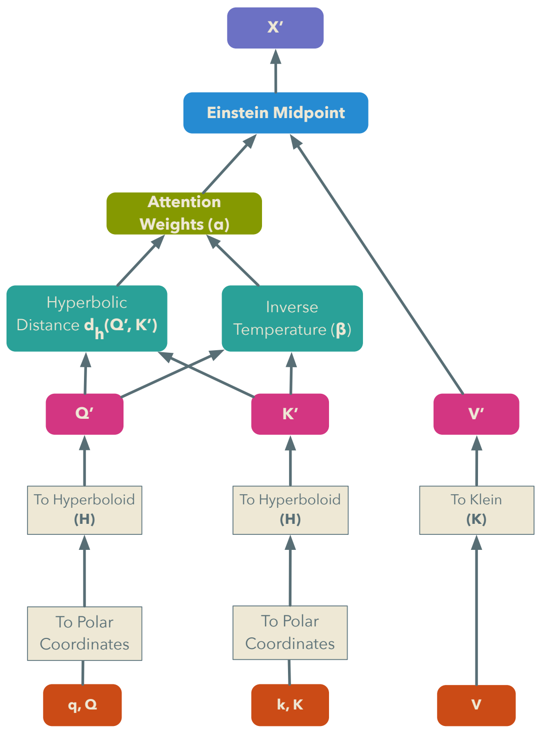

Figure 2: The computational graph for the self-attention mechanism of the hyperbolic Transformer. We show the different operations in the blocks and their interactions are represented by the arrows.

hyperbolic space we take,

$$\alpha(\mathbf{q}_i, \mathbf{k}_j) = f(-\beta d_{\mathbb{H}}(\mathbf{q}_i, \mathbf{k}_j) - c) \quad , \tag{2}$$

where $d_{\mathbb{H}}(\cdot, \cdot)$ is the hyperbolic distance, and β and c are parameters that can be set manually or learned along with the rest of the network. Having the bias parameter c is useful because distances are non-negative. We take the function $f(\cdot)$ to be either $\exp(\cdot)$, in which case we set the normalization appropriately to obtain a softmax, or $\mathrm{sigmoid}(\cdot)$.

Hyperbolic aggregation: The path to extend the weighted midpoint to hyperbolic space is much less obvious, but fortunately such a extension already exists as the Einstein midpoint. The Einstein midpoint is straightforward to compute by adjusting the aggregation weights appropriately (see Ungar [39, Definition 4.21])

$$m_i(\{\alpha_{ij}\}_j, \{\mathbf{v}_{ij}\}_j) = \sum_j \left[\frac{\alpha_{ij}\gamma(\mathbf{v}_{ij})}{\sum_\ell \alpha_{i\ell}\gamma(\mathbf{v}_{i\ell})} \right] \mathbf{v}_{ij} \quad , \tag{3}$$

where the $\gamma(\mathbf{v}_{ij})$ are the Lorentz factors, that are given by $\gamma(\mathbf{v}_{ij}) = \frac{1}{\sqrt{1-\|\mathbf{v}_{ij}\|^2}}$. The norm in the denominator of the Lorentz factor is the Euclidean norm of the Klein coordinates of the point $\mathbf{v}_{ij}$, and the correctness of Equation 3 also relies on the points $\mathbf{v}_{ij}$ being represented by their Klein coordinates. Fortunately the various models of hyperbolic space in common use are all isomorphic, so we can work in an arbitrary hyperbolic model and simply project to and from the Klein model to execute midpoint computations, as we discuss in the following section.

The reason for using the Einstein midpoint for hyperbolic aggregation is that it obeys many of the properties that we expect from a weighted average in Euclidean space. In particular, it is invariant to translating the $\mathbf{v}_{ij}$'s by a fixed distance in a common direction, and also by rotations of the constellation of points about the origin. The derivation of this operation is quite involved, and beyond the scope of this paper. We point the interested reader to Ungar [39, 40] for a full exposition.

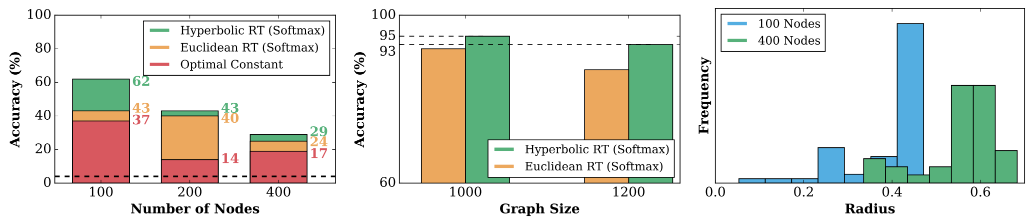

Figure 3: **Left:** Performance of the Recursive Transformer models on the Shortest Path Length Prediction task on graphs of various sizes. The black dashed line indicates chance performance. **Center:** Results on Link Prediction Tasks. **Right:** The histogram of the radiuses for a model trained on a graph with 100 and 400 nodes.

5 Experiments

We evaluate our models on synthetic and real-world tasks. Experiments where the underlying graph structure is explicitly known clearly show the benefits of using hyperbolic geometry as an inductive bias. At the same time, we show that real-world tasks withi implicit graph strucutre such as a diagostic visual question answering task [15], and neural machine translation, equally benefit from relying on hyperbolic geometry.

We provide experiments with feedforward networks, the Transformer [41] and Relation Networks [34] endowed with hyperbolic attention.

Our results show the effectiveness of our approach on diverse tasks and architectures. The benefit of our approach is particularly prominent in relatively small models, which supports our hypothesis that hyperbolic geometry induces compact representations and is therefore better able to represent complex functions in limited space.

5.1 Modeling scale-free graphs

We use the algorithm of von Looz et al. [45] to efficiently generate large scale-free graphs, and define two predictive tasks that test our model's ability to represent different aspects of the structure of these networks. For both tasks in this section, we train Recursive Transformer (RT) models, using hyperbolic and Euclidean attention. A Recursive Transformer is identical to the original transformer, except that the weights of each self-attention layer are tied across depth. We use models with 3 recursive self-attention layers, each of which has 4 heads with 4 units each for each of $\mathbf{q}$, $\mathbf{k}$, and $\mathbf{v}$. This model has similarities to Graph Attention Networks [42, 18].

Link prediction (LP): Link prediction is a classical graph problem, where the task is to predict if an edge exists between two nodes in the graph. We experimented with graphs of 1000 and 1200 nodes and observed that the hyperbolic RT performs better than the Euclidean RT on both tasks. We report the results in Figure 3 (middle). In general, we observed that for graphs of size 1000 and 1200 the hyperbolic transformer performs better than the Euclidean transformer given the same amount of capacity.

Shortest path length prediction (SPLP): In this task, the goal is to predict the length of the shortest path between a pair of nodes in the graph. We treat this as a classification problem with a maximum path-length of 25 which becomes naturally an unbalanced classification problem. We use rejection sampling during training to ensure the network is trained on an approximately uniform distribution of path lengths. At test time we sample paths uniformly at random, so the length distribution follows that of the underlying graphs. We report the results in Figure 3 (left). In Figure 3 (right), we visualize the distribution of the scale of the learned activations (r in the projection of Section 4.1) when training on graphs of size 100 and 400. We observe that our model tends to use larger scales for the larger graphs. As a baseline, we compare to the optimal constant predictor, which always predicts the most common expected path length. This baseline does quite well since the path length distribution on the test set is quite skewed.

For both tasks, we generate training data online. Each example is a new graph in which we query the connectivity of a randomly chosen pair of nodes. To make training easier, we use a curriculum,

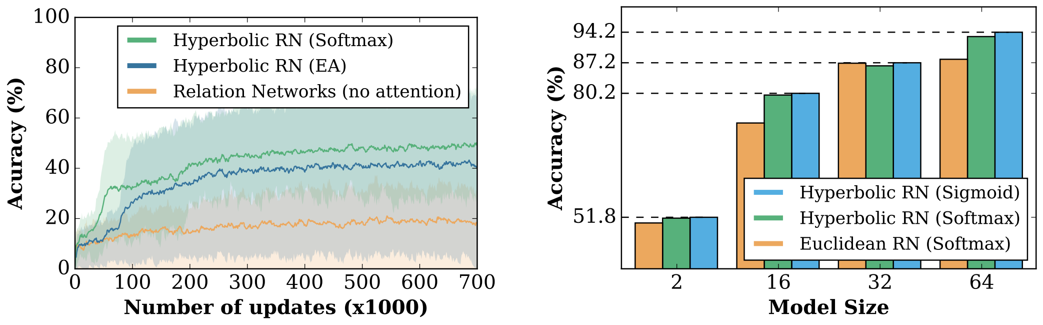

Figure 4: **Left:** Comparison of our models with low-capacity on the Sort-of-CLEVR dataset. The "EA" refers to the model that uses hyperbolic attention weights with Euclidean aggregation. **Right:** Performance of Relation Network extended by attention mechanism in either Euclidean or hyperbolic space on the CLEVR dataset.

whereby we start training on smaller graphs and gradually increase the number of vertices towards the final number. More details on the dataset generation procedure and the curriculum scheme are found in the supplementary material.

5.2 Sort-of-CLEVR

Since we expect hyperbolic attention to be particularly well suited to relational modelling, we investigate our models on the relational variant of the Sort-of-CLEVR dataset [34]. This dataset consists of simple visual scenes allowing us to solely focus on the relational aspect of the problem. Our models extend Relation Nets (RNs) with the attention mechanism in hyperbolic space (with the Euclidean or Einstein midpoint aggregation), but otherwise we follow the standard setup-up [34]. Our best method yields accuracy of 99.2% that significantly exceeds the accuracy of the original RN (96%).

However, we are more interested in evaluating models on the low-capacity regime. Indeed, as Figure 4 (left) shows, the attention mechanism computed in the hyperbolic space improves around 20 percent points over the standard RN, where all the models use only two units of the relational MLP.

5.3 CLEVR

We train our Relation Network with various attention mechanisms on the CLEVR dataset [15]. CLEVR is a synthetic visual question answering datasets consisting of 3D rendered objects like spheres, cubes, or cylinders of various size, material, or color. In contrast to other visual question answering datasets [1, 24, 47], the focus of CLEVR is on relational reasoning.

In our experiments, we closely follow the procedure established in [34], both in terms of the model architecture, capacity, or the choice of the hyperparameters, and only differ by the attention mechanism (Euclidean or hyperbolic attention), or sigmoid activations.

Results are shown in Figure 4 (Right). For each model, we vary the capacity of the relational part of the network and report the resulting test accuracy. We find that hyperbolic attention with sigmoid consistently outperforms other models.

Our RN with hyperbolic attention and sigmoid achieves 95.7% accuracy on the test set at the same capacity level as RN, whereas the latter reportedly achieves 95.5% accuracy [34].

5.4 Neural machine translation

The Transformer [41] is a recently introduced state of the art model for neural machine translation. It relies heavily on attention as its core operation. As described in Section 3.3, we have extended the Transformer[2] by replacing its scaled dot-product attention operation with its hyperbolic counterpart. We evaluate all the models on the WMT14 En-De dataset [4].

[2]We use a publicly available version: `https://github.com/tensorflow/tensor2tensor`

	WMT 2014 En-De BLEU Scores	
	Tiny	**Base**
Transformer (Vaswani et al. [41])	-	27.3
Transformer (Shaw et al. [36])	-	26.5
Transformer (Latest)	17.3	27.1
Hyperbolic Transformer (+Sigmoid)	17.3	27.4
Hyperbolic Transformer (+Softmax, +Polar)	17.5	27.0
Hyperbolic Transformer (+Sigmoid, +Polar)	**18.0**	**27.5**

Table 1: Results for the WMT14 English to German translation task. Results are computed following the procedure in Vaswani et al. [41]. Citations indicate results taken from the literature. *Latest* is the result of training a new model using an unmodified version of the same code where we added hyperbolic attention (we have observed that the exact performance of the transformer on this task varies as the Tensor2tensor codebase evolves).

We train several versions of the Transformer model with hyperbolic attention. They use different coordinate systems (Weierstrass or polar), or different attention functions (softmax or sigmoid). We consider two model sizes, referred to here as *tiny* and *base*. The tiny model has two layers of encoders and decoders, each with 128 units and 4 attention heads. The base model has 6 layers of encoders and decoders, each with 512 units and 8 attention heads. All hyperparameter configurations for the Euclidean versions of these models are available in the Tensor2tensor repository.

Results are shown in Table 1. We observe improvements over the Euclidean model by using hyperbolic attention, in particular when coupled with the sigmoid activation function for the attention weights. The improvements are more significant when the model capacity is restricted. In addition, our best model (with sigmoid activation function and without pseudo-polar coordinates) using the *big* architecture from Tensor2tensor, achieves 28.45 BLEU score, whereas Vaswani et al. [41] report 28.4 BLEU score with the original version of this model.[3].

6 Conclusion

We have presented a novel way to impose the inductive biases from hyperbolic geometry on the activations of deep neural networks. Our proposed hyperbolic attention operation makes use of hyperbolic geometry in both the computation of the attention weights, and in the aggregation operation over values. We implemented our proposed hyperbolic attention mechanism in both Relation Networks and the Transformer and showed that we achieve improved performance on a diverse set of tasks. We have shown improved performance on link prediction and shortest path length prediction in scale free graphs, on two visual question answering datasets, and finally on English to German machine translation. The gains are particularly prominent in relatively small models, which confirms our hypothesis that hyperbolic geometry induces more compact representations.

Acknowledgements

We would like to thank Neil Rabinowitz, Chris Dyer and Serkan Cabi for constructive comments on earlier versions of this draft. We thank Yannis Assael for helping us with the styles of the plots in this draft. We would like to thank Thomas Paine for the discussions.

References

[1] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2425–2433, 2015.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*, 2014.

[3]We achieve 28.3 BLEU score using the *big* Transformer with the publicly available framework.

[3] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. In *Advances in neural information processing systems*, pages 4502–4510, 2016.

[4] Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/W/W14/W14-3302`.

[5] Gunnar A Borg. Psychophysical bases of perceived exertion. *Med sci sports exerc*, 14(5):377–381, 1982.

[6] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[7] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[8] Benjamin Paul Chamberlain, James Clough, and Marc Peter Deisenroth. Neural embeddings of graphs in hyperbolic space. *arXiv preprint arXiv:1705.10359*, 2017.

[9] Aaron Clauset, Cosma Rohilla Shalizi, and Mark EJ Newman. Power-law distributions in empirical data. *SIAM review*, 51(4):661–703, 2009.

[10] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3844–3852, 2016.

[11] Yan Duan, Marcin Andrychowicz, Bradly Stadie, OpenAI Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. One-shot imitation learning. In *Advances in neural information processing systems*, pages 1087–1098, 2017.

[12] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. *arXiv preprint arXiv:1704.01212*, 2017.

[13] Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471, 2016.

[14] Birger Iversen. *Hyperbolic geometry*, volume 25. Cambridge University Press, 1992.

[15] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 1988–1997. IEEE, 2017.

[16] Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. *arXiv preprint arXiv:1802.04687*, 2018.

[17] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[18] WWM Kool and M Welling. Attention solves your tsp. *arXiv preprint arXiv:1803.08475*, 2018.

[19] Dmitri Krioukov, Fragkiskos Papadopoulos, Amin Vahdat, and Marián Boguná. Curvature and temperature of complex networks. *Physical Review E*, 80(3):035101, 2009.

[20] Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguná. Hyperbolic geometry of complex networks. *Physical Review E*, 82(3):036106, 2010.

[21] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015.

[22] Henry W Lin and Max Tegmark. Critical behavior in physics and probabilistic formal languages. *Entropy*, 19(7):299, 2017.

[23] Nathan Linial, Avner Magen, and Michael E Saks. Low distortion euclidean embeddings of trees. *Israel Journal of Mathematics*, 106(1):339–348, 1998.

[24] Mateusz Malinowski and Mario Fritz. A multi-world approach to question answering about real-world scenes based on uncertain input. In *Advances in neural information processing systems*, pages 1682–1690, 2014.

[25] Brian J McGill, Brian J Enquist, Evan Weiher, and Mark Westoby. Rebuilding community ecology from functional traits. *Trends in ecology & evolution*, 21(4):178–185, 2006.

[26] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[27] Mark EJ Newman. Power laws, pareto distributions and zipf's law. *Contemporary physics*, 46(5):323–351, 2005.

[28] Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems*, pages 6341–6350, 2017.

[29] Jorg Ontrup and Helge Ritter. Hyperbolic self-organizing maps for semantic navigation. In *Advances in neural information processing systems*, pages 1417–1424, 2002.

[30] Fragkiskos Papadopoulos, Dmitri Krioukov, Marián Boguñá, and Amin Vahdat. Greedy forwarding in dynamic scale-free networks embedded in hyperbolic metric spaces. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9. IEEE, 2010.

[31] Steven T Piantadosi. Zipf's word frequency law in natural language: A critical review and future directions. *Psychonomic bulletin & review*, 21(5):1112–1130, 2014.

[32] David MW Powers. Applications and explanations of zipf's law. In *Proceedings of the joint conferences on new methods in language processing and computational natural language learning*, pages 151–160. Association for Computational Linguistics, 1998.

[33] Helge Ritter. Self-organizing maps on non-euclidean spaces. In *Kohonen maps*, pages 97–109. Elsevier, 1999.

[34] Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Tim Lillicrap. A simple neural network module for relational reasoning. In *Advances in neural information processing systems*, pages 4974–4983, 2017.

[35] Rik Sarkar. Low distortion delaunay embedding of trees in hyperbolic plane. In *International Symposium on Graph Drawing*, pages 355–366. Springer, 2011.

[36] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*, 2018.

[37] Shuntaro Takahashi and Kumiko Tanaka-Ishii. Do neural nets learn statistical laws behind natural language? *PloS one*, 12(12):e0189326, 2017.

[38] 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*, pages 583–591. ACM, 2018.

[39] Abraham Albert Ungar. *Analytic hyperbolic geometry: Mathematical foundations and applications*. World Scientific, 2005.

[40] Abraham Albert Ungar. A gyrovector space approach to hyperbolic geometry. *Synthesis Lectures on Mathematics and Statistics*, 1(1):1–194, 2008.

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

[42] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[43] Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. Order-embeddings of images and language. In *ICLR*, 2016.

[44] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700, 2015.

[45] Moritz von Looz, Henning Meyerhenke, and Roman Prutkin. Generating random hyperbolic graphs in subquadratic time. In *International Symposium on Algorithms and Computation*, pages 467–478. Springer, 2015.

[46] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. *arXiv preprint arXiv:1711.07971*, 2017.

[47] Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. Visual7w: Grounded question answering in images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4995–5004, 2016.

A Appendix

A.1 More on models of hyperbolic space

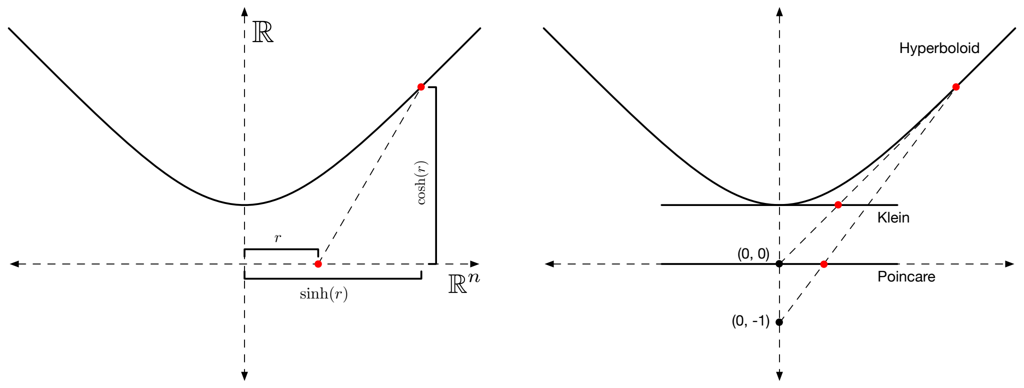

Figure 5: Relationships between different representations of points used in the paper. **Left:** The relationship between pseudo-polar coordinates in $\mathbb{R}^n$ and the hyperboloid in $\mathbb{R}^{n+1}$. **Right:** Projections relating the hyperboloid, Klein and Poincaré models of hyperbolic space.

A.2 Scale-free graph generation

We use the algorithm described by von Looz et al. [45]. In our experiments, we set the α to 0.95 and *edge_radius_R_factor* to 0.35.

A.3 Scale-free graph curriculum

Curriculum was an essential part of our training on the scale-free graph tasks. On LP and SPLP tasks, we use a curriculum where we extract the connected components from the graph by cutting the disk that the graphs generated on into slices by starting from a 30 degree angle and gradually increasing the size of the slice from the disk by increasing the angle during the training according to the number of lessons that are involved in the curriculum. This process is also visualized in Figure 6.

A.4 Travelling salesman problem (TSP)

We train an off-policy DQN-like agent [26] with the HRT. The graphs for the TSP is generated following the procedure introduced in [44].

On this task, as an ablation we just compared the hyperbolic networks with and The results are provided in Figure 4 (Right) with and without implicit coordinates. Overall, we found that the hyperbolic transformer networks performs better when using the implicit polar coordinates.

A.5 Hyperbolic Recursive Transformer

As shown in Figure 9, the hyperbolic RT is an extension of transformer that ties the parameters of the self-attention layers. The self-attention layer gets the representations of the nodes of the graph coming from the encoder and the decoder decodes that representation from the recursive self-attention layers for the prediction.

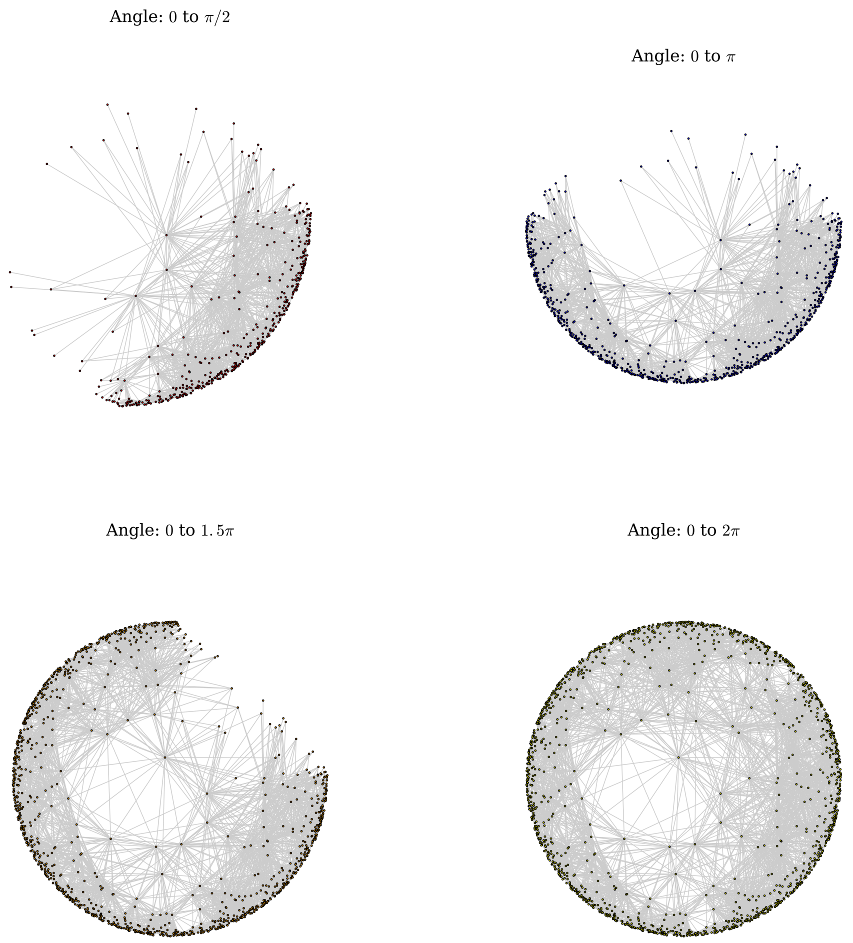

Figure 6: We show an example of a curriculum on the hyperbolic disk. In the first lesson, we take slices from the graph only between angle 0 and $\pi/2$. In the second lesson we will have to take the slice from 0 to π.

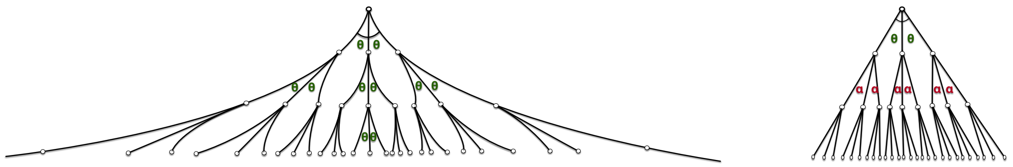

Figure 7: An illustration of how trees can be represented in hyperbolic (left) and Euclidean geometry (right) in a cone. In hyperbolic space, as the tree grows the angles between the edges (θ) can be preserved from one level to the next. In Euclidean space, since the number of nodes in the tree grows faster than the rate that the volume grows, angles may not be preserved (θ to α). Lines in the left diagram are straight in hyperbolic space, but appear curved in this Euclidean diagram.

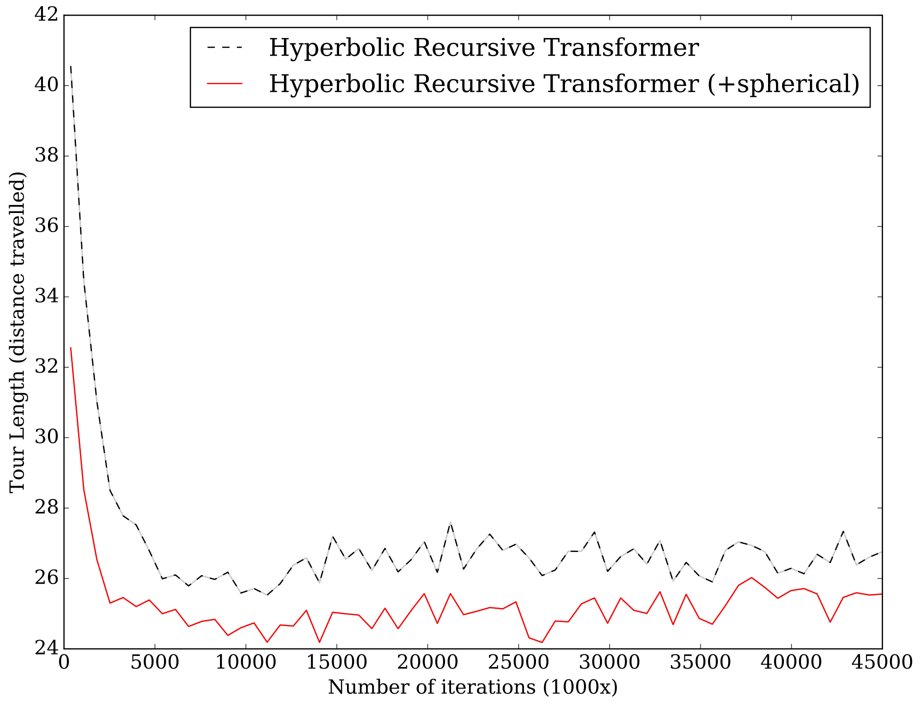

Figure 8: The comparisons between a hyperbolic recursive transformer with and without spherical coordinates on the travelling salesman problem.

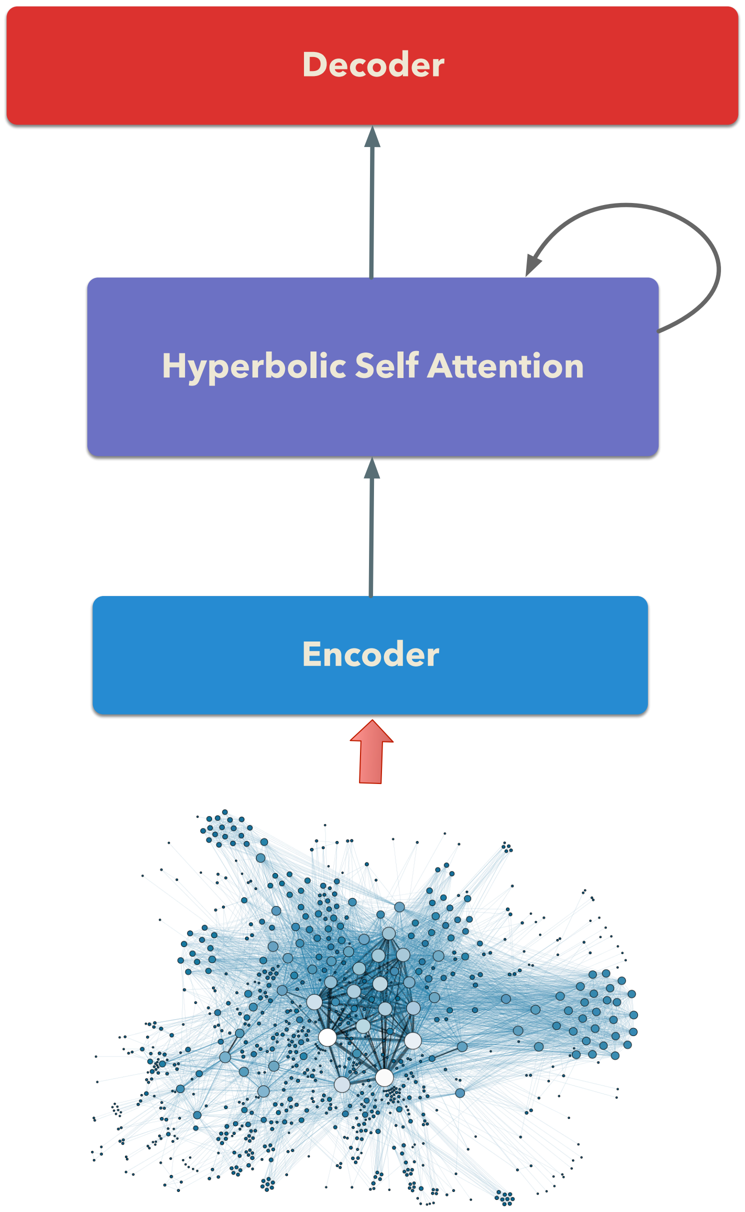

Figure 9: A depiction of a hyperbolic recursive transformer on the graph structured data. The model takes in nodes of the graph as an input and the encoder maps those inputs and provides them to the recursive hyperbolic self-attention block as an input.