

# HVT: A COMPREHENSIVE VISION FRAMEWORK FOR LEARNING IN NON-EUCLIDEAN SPACE

**Anonymous authors**

Paper under double-blind review

## ABSTRACT

Data representation in non-Euclidean spaces has proven effective for capturing hierarchical and complex relationships in real-world datasets. Hyperbolic spaces, in particular, provide efficient embeddings for hierarchical structures. This paper introduces the Hyperbolic Vision Transformer (HVT), a novel extension of the Vision Transformer (ViT) that integrates hyperbolic geometry. While traditional ViTs operate in Euclidean space, our method enhances the self-attention mechanism by leveraging hyperbolic distance and Möbius transformations. This enables more effective modeling of hierarchical and relational dependencies in image data. We present rigorous mathematical formulations, showing how hyperbolic geometry can be incorporated into attention layers, feed-forward networks, and optimization. We offer improved performance for image classification using the ImageNet dataset.

## 1 INTRODUCTION

Representation learning is fundamental to modern machine learning, enabling models to extract meaningful features from raw data Bengio et al. (2014). While Euclidean spaces are traditionally used to model data relationships, many real-world datasets—including images—exhibit hierarchical structures better captured in non-Euclidean spaces Bronstein et al. (2017).

Images are inherently hierarchical, comprising structures at multiple scales: from pixels to edges, from shapes to objects, and ultimately to entire scenes Biederman (1987); Riesenhuber & Poggio (1999). This hierarchy can be conceptualized as:

- **Pixels**  $\{p_i\}$ : The basic units of an image.
- **Edges**  $\{e_j\}$ : Formed by grouping pixels with significant intensity gradients.
- **Shapes**  $\{s_k\}$ : Created by combining edges into simple motifs.
- **Objects**  $\{o_l\}$ : Recognizable entities composed of shapes.
- **Scenes**  $\mathcal{I}$ : Complete images where objects interact within a context.

The hierarchical nature of images implies that higher-level concepts are built upon lower-level features, reflecting a tree-like structure. Vision Transformers (ViTs) Dosovitskiy et al. (2021a) process images by dividing them into patches, treating each patch as a token. This patch-based approach introduces a hierarchical representation because:

1. **Local Features**: Each patch captures local patterns such as textures or edges.
2. **Global Context**: By attending over patches, the model aggregates local information to understand the overall structure.

This mirrors the hierarchical composition of images, from local to global features.

Hyperbolic spaces are well-suited for modeling hierarchical data due to their ability to embed tree-like structures with minimal distortion Nickel & Kiela (2017). By utilizing hyperbolic geometry and Möbius transformations, we can effectively capture the multi-scale dependencies inherent in visual data Ganea et al. (2018). Specifically, Möbius transformations allow for operations like addition and scalar multiplication in hyperbolic space, enabling neural networks to perform calculations while preserving hierarchical relationships.

In this paper, we propose the *Hyperbolic Vision Transformer*, which integrates hyperbolic geometry into the transformer architecture. Our contributions include:

- **Hyperbolic Neural Components:** Extending ViT to operate in hyperbolic space using hyperbolic versions of neural network components, such as attention mechanisms and linear layers.
- **Möbius Transformations in ViT:** Demonstrating how Möbius transformations enable operations in hyperbolic space, preserving hierarchical data structures.
- **Theoretical and Empirical Analysis:** Providing insights and evaluations showing improved modeling of hierarchical structures over traditional Euclidean approaches.

## 2 RELATED WORK

### 2.1 HYPERBOLIC GEOMETRY IN MACHINE LEARNING

Recent advances have leveraged hyperbolic geometry for machine learning, significantly impacting how hierarchical data is modeled. Nickel & Kiela (2017) successfully used Poincaré embeddings for such data, showing marked improvements over Euclidean embeddings. This approach was further refined by Ganea et al. (2018), who introduced hyperbolic embeddings for entailment cones, effectively capturing asymmetric relationships.

Khrulkov et al. (2020) and Liu et al. (2020) extended these concepts to visual data and zero-shot recognition, respectively, underscoring the versatility of hyperbolic embeddings in handling complex visual tasks. Most notably, Ermolov et al. (2022) developed Hyperbolic Vision Transformers (HVTs) that incorporate these embeddings within Vision Transformer architectures, enhancing metric learning. Our model extends this integration by incorporating hyperbolic geometry throughout the transformer operations, from Möbius transformations to hyperbolic self-attention.

### 2.2 HYPERBOLIC NEURAL NETWORKS AND ATTENTION

Ganea et al. (2018) initially formulated hyperbolic neural networks, introducing layers and activation functions suited for hyperbolic spaces. This foundation was expanded by Bachmann et al. (2020), who focused on optimizing these networks efficiently. Our method enriches this foundation by embedding hyperbolic layers directly within transformer architectures, enhancing the adaptability and depth of hyperbolic operations.

Another method presented in Hyperbolic Attention Networks Gulcehre et al. (2018) differs from ours primarily in how hyperbolic geometry is applied within the attention mechanism. Gulcehre et al. focus on embedding the activations into hyperbolic space using both the hyperboloid and Klein models, leveraging hyperbolic matching and aggregation operations. In contrast, our approach incorporates learnable curvature within positional embeddings, head-specific scaling in attention, and hyperbolic layer normalization, offering more flexibility and efficiency in capturing hierarchical data. Additionally, we utilize the Poincaré ball model for its computational suitability in vision tasks.

### 2.3 VISION TRANSFORMERS

Initially developed for NLP by Vaswani et al. (2017), Vision Transformers (ViT) have been adapted for visual tasks, as demonstrated by Dosovitskiy et al. (2021b). Enhancements such as those proposed by Caron et al. (2021) and El-Nouby et al. (2021) have refined ViTs for self-supervised learning. Unlike these Euclidean-based models, our Hyperbolic Vision Transformer employs hyperbolic geometry to model hierarchical and relational data structures more effectively.

### 2.4 COMPARISON TO KEY HYPERBOLIC METHODS

While Ermolov et al. (2022) introduced HVTs that focus on hyperbolic embeddings for metric learning, our approach fully integrates hyperbolic operations throughout the transformer, significantly enhancing its ability to manage hierarchical data beyond the objectives of metric learning. We extend

the application scope to include image classification by embedding hyperbolic geometry directly into the core components of the Vision Transformer.

Recently, Yang et al. (2024) proposed Hypformer, an efficient hyperbolic Transformer based on the Lorentz model of hyperbolic geometry. Hypformer introduces two foundational blocks—Hyperbolic Transformation with Curvatures (HTC) and Hyperbolic Readjustment and Refinement with Curvatures (HRC)—to define essential Transformer modules in hyperbolic space. They also develop a linear self-attention mechanism in hyperbolic space to handle large-scale graph data and long-sequence inputs efficiently.

While Hypformer makes significant contributions to the development of hyperbolic Transformers, particularly in processing large-scale graph data, our model differs in several key aspects:

- **Model Focus and Application Domain:** Hypformer is primarily designed for graph data and emphasizes scalability and efficiency in handling large-scale graphs and long sequences. In contrast, our model focuses on vision tasks, specifically image classification, integrating hyperbolic geometry throughout the Vision Transformer architecture to enhance the modeling of hierarchical and relational structures inherent in visual data.
- **Hyperbolic Model Used:** Hypformer operates in the Lorentz model of hyperbolic geometry, whereas we utilize the Poincaré ball model. The Poincaré ball model is advantageous for vision tasks due to its conformal properties, which preserve angles and better represent geometric structures in image data.
- **Innovative Components:** Our model introduces unique components such as learnable curvature in positional embeddings, head-specific scaling in the attention mechanism, hyperbolic layer normalization, gradient clipping, geodesic regularization, and layer scaling for training stability. These innovations are specifically tailored to enhance the performance of Vision Transformers in hyperbolic space.
- **Attention Mechanism:** While Hypformer develops a linear self-attention mechanism to address efficiency in handling large-scale data, our model extends the standard self-attention mechanism into hyperbolic space using Möbius operations and hyperbolic distance calculations. This approach allows us to capture complex relationships in visual data more effectively.

By fully integrating hyperbolic operations within the Vision Transformer framework and focusing on the unique challenges of vision tasks, our model offers a comprehensive solution that provides superior performance over previous hyperbolic Transformers, including Hypformer, in the domain of image classification.

### 3 PROPOSED METHOD

This section presents the mathematical foundations of the Hyperbolic Vision Transformer Network (HVT). We cover essential concepts from hyperbolic geometry and describe how we adapt Vision Transformer components to operate in hyperbolic space. Our main contributions include introducing learnable curvature in positional embeddings, head-specific scaling in attention mechanisms, hyperbolic layer normalization, gradient clipping, geodesic regularization, and layer scaling for training stability.

**Code Availability:** Code is available at <https://github.com/hyperbolicvit/hyperbolicvit>

#### 3.1 HYPERBOLIC GEOMETRY PRELIMINARIES

Hyperbolic space, characterized by constant negative curvature, embeds hierarchical and complex structures common in visual data. We adopt the Poincaré ball model due to its computational convenience and suitability for representing image data structures.

##### 3.1.1 POINCARÉ BALL MODEL

The  $n$ -dimensional Poincaré ball model is defined as the manifold:

$$\mathbb{D}^n = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}\| < 1\}, \quad (1)$$

where  $\|\cdot\|$  denotes the Euclidean norm. The Riemannian metric tensor  $g_{\mathbf{x}}$  of this manifold is given by:

$$g_{\mathbf{x}} = \lambda_{\mathbf{x}}^2 g^E, \quad \text{with} \quad \lambda_{\mathbf{x}} = \frac{2}{1 - \|\mathbf{x}\|^2}, \quad (2)$$

where  $g^E$  is the Euclidean metric tensor, and  $\lambda_{\mathbf{x}}$  is the conformal factor that scales the Euclidean metric to account for the curvature of hyperbolic space.

### 3.1.2 MÖBIUS OPERATIONS

To adapt Vision Transformer components to hyperbolic space, we utilize Möbius transformations, essential for processing vectors within the Poincaré ball model.

**Möbius Addition** For vectors  $\mathbf{x}, \mathbf{y} \in \mathbb{D}^n$ , the Möbius addition  $\mathbf{x} \oplus \mathbf{y}$  is defined as:

$$\mathbf{x} \oplus \mathbf{y} = \frac{(1 + 2\langle \mathbf{x}, \mathbf{y} \rangle + \|\mathbf{y}\|^2)\mathbf{x} + (1 - \|\mathbf{x}\|^2)\mathbf{y}}{1 + 2\langle \mathbf{x}, \mathbf{y} \rangle + \|\mathbf{x}\|^2\|\mathbf{y}\|^2}, \quad (3)$$

where  $\langle \cdot, \cdot \rangle$  denotes the Euclidean inner product. Möbius addition generalizes vector addition to hyperbolic space, ensuring the result remains within the manifold.

**Möbius Scalar Multiplication** For a scalar  $r \in \mathbb{R}$  and a vector  $\mathbf{x} \in \mathbb{D}^n$ , Möbius scalar multiplication  $r \otimes \mathbf{x}$  is defined as:

$$r \otimes \mathbf{x} = \tanh(r \tanh^{-1}(\|\mathbf{x}\|)) \frac{\mathbf{x}}{\|\mathbf{x}\|}. \quad (4)$$

This operation scales a vector while preserving its direction and ensures the scaled vector remains within the Poincaré ball.

**Möbius Matrix-Vector Multiplication** Given a matrix  $\mathbf{W} \in \mathbb{R}^{m \times n}$  and a vector  $\mathbf{x} \in \mathbb{D}^n$ , Möbius matrix-vector multiplication  $\mathbf{W} \otimes_M \mathbf{x}$  is defined as:

$$\mathbf{W} \otimes_M \mathbf{x} = \tanh\left(\frac{\|\mathbf{W}\mathbf{x}\|}{\|\mathbf{x}\|} \tanh^{-1}(\|\mathbf{x}\|)\right) \frac{\mathbf{W}\mathbf{x}}{\|\mathbf{W}\mathbf{x}\|}, \quad \text{if } \mathbf{x} \neq \mathbf{0}. \quad (5)$$

This operation extends Möbius scalar multiplication to linear transformations, allowing us to apply linear layers within hyperbolic space.

**Möbius Concatenation** To combine multiple vectors  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n \in \mathbb{D}^n$ , we use Möbius concatenation:

$$\bigoplus_{j=1}^n \mathbf{x}_j = \mathbf{x}_1 \oplus \mathbf{x}_2 \oplus \dots \oplus \mathbf{x}_n. \quad (6)$$

## 3.2 HYPERBOLIC NEURAL NETWORK COMPONENTS

We incorporate the hyperbolic operations into our neural network components to enable the Vision Transformer architecture to operate within hyperbolic space.

### 3.2.1 HYPERBOLIC LINEAR LAYER

Traditional linear layers are adapted to hyperbolic space using Möbius matrix-vector multiplication followed by Möbius addition with a bias:

$$\mathbf{h} = \mathbf{W} \otimes_M \mathbf{x} \oplus \mathbf{b}, \quad (7)$$

where  $\mathbf{x} \in \mathbb{D}^n$ ,  $\mathbf{W} \in \mathbb{R}^{m \times n}$ , and  $\mathbf{b} \in \mathbb{D}^m$ . This layer allows us to perform linear transformations while respecting the geometry of hyperbolic space.

### 3.2.2 HYPERBOLIC ACTIVATION AND NORMALIZATION

Activation functions and normalization are applied in the tangent space at the origin to leverage familiar Euclidean operations. The logarithmic map  $\log_0 : \mathbb{D}^n \rightarrow T_0\mathbb{D}^n$  maps points from the manifold to the tangent space:

$$\log_0(\mathbf{x}) = \frac{2 \tanh^{-1}(\|\mathbf{x}\|)}{\|\mathbf{x}\|} \mathbf{x}. \quad (8)$$

The exponential map  $\exp_0 : T_0\mathbb{D}^n \rightarrow \mathbb{D}^n$  brings points back to the manifold:

$$\exp_0(\mathbf{v}) = \tanh\left(\frac{\|\mathbf{v}\|}{2}\right) \frac{\mathbf{v}}{\|\mathbf{v}\|}. \quad (9)$$

Using these maps, we define hyperbolic versions of the ReLU activation and Layer Normalization:

$$\text{ReLU}^{\mathbb{D}}(\mathbf{x}) = \exp_0(\text{ReLU}(\log_0(\mathbf{x}))), \quad (10)$$

$$\text{LayerNorm}^{\mathbb{D}}(\mathbf{x}) = \exp_0(\text{LayerNorm}(\log_0(\mathbf{x}))). \quad (11)$$

This approach allows us to apply standard activation and normalization techniques within hyperbolic space.

### 3.2.3 HYPERBOLIC LAYER SCALING

To stabilize residual connections in hyperbolic space, we introduce a learnable scaling parameter  $\beta$ :

$$\mathbf{X}' = \mathbf{X} \oplus \beta \otimes \mathbf{O}, \quad (12)$$

where  $\mathbf{X}$  is the input,  $\mathbf{O}$  is the output from a layer, and  $\beta \in \mathbb{R}$  scales the residual contribution.

## 3.3 HYPERBOLIC VISION TRANSFORMER ARCHITECTURE

We integrate hyperbolic geometry into the Vision Transformer architecture by modifying key components to operate within hyperbolic space.

### 3.3.1 LEARNABLE HYPERBOLIC POSITIONAL EMBEDDINGS

Positional embeddings are adjusted using a learnable curvature parameter  $c$  to represent positional information in hyperbolic space better:

$$\mathbf{E}_{\text{pos}} = c \otimes \mathbf{E}_{\text{pos}}^0, \quad (13)$$

$$\mathbf{X} = \mathbf{X} \oplus \mathbf{E}_{\text{pos}}, \quad (14)$$

where  $\mathbf{E}_{\text{pos}}^0 \in \mathbb{D}^{1 \times N \times E}$  are the initial positional embeddings.

### 3.3.2 HYPERBOLIC SELF-ATTENTION MECHANISM

We extend the self-attention mechanism to hyperbolic space using Möbius operations and hyperbolic distance computations. Table 1 summarizes the symbols used.

Table 1: Symbols and their descriptions.

Symbol	Description
$B$	Batch size.
$H$	Number of attention heads.
$N$	Sequence length.
$D$	Head dimension, where $D = \frac{E}{H}$ and $E$ is the embedding dimension.
$c$	Curvature parameter of the Poincaré ball model.
$\epsilon$	Small constant for numerical stability ( $\epsilon = 1 \times 10^{-15}$ ).
$\delta$	Small constant for clamping ( $\delta = 1 \times 10^{-7}$ ).
$p$	DropConnect Wan et al. (2013) probability.

Given input embeddings  $\mathbf{X} \in \mathbb{D}^{B \times N \times E}$ , we compute the queries  $\mathbf{Q}$ , keys  $\mathbf{K}$ , and values  $\mathbf{V}$  using hyperbolic linear layers:

270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323

$$\mathbf{Q} = \mathbf{X} \otimes_M \mathbf{W}_Q \oplus \mathbf{b}_Q, \quad (15)$$

$$\mathbf{K} = \mathbf{X} \otimes_M \mathbf{W}_K \oplus \mathbf{b}_K, \quad (16)$$

$$\mathbf{V} = \mathbf{X} \otimes_M \mathbf{W}_V \oplus \mathbf{b}_V. \quad (17)$$

We reshape these tensors to accommodate multiple attention heads:

$$\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{D}^{B \times H \times N \times D}. \quad (18)$$

**Hyperbolic Distance Computation** The hyperbolic distance between a query  $\mathbf{Q}_{b,h,i}$  and a key  $\mathbf{K}_{b,h,j}$  is computed using the distance function in the Poincaré ball model:

$$d_{\mathbb{D}}(\mathbf{Q}_{b,h,i}, \mathbf{K}_{b,h,j}) = \cosh^{-1} \left( 1 + \frac{2c \|\mathbf{Q}_{b,h,i} \oplus (-\mathbf{K}_{b,h,j})\|^2}{(1 - c \|\mathbf{Q}_{b,h,i}\|^2)(1 - c \|\mathbf{K}_{b,h,j}\|^2) + \epsilon} \right). \quad (19)$$

To ensure numerical stability, we clamp the argument of  $\cosh^{-1}$ :

$$\mathcal{A}_{b,h,i,j} = \max \left( 1 + \frac{2c \|\mathbf{Q}_{b,h,i} \oplus (-\mathbf{K}_{b,h,j})\|^2}{(1 - c \|\mathbf{Q}_{b,h,i}\|^2)(1 - c \|\mathbf{K}_{b,h,j}\|^2) + \epsilon}, 1 + \delta \right). \quad (20)$$

We then compute the distance as:

$$d_{\mathbb{D}}(\mathbf{Q}_{b,h,i}, \mathbf{K}_{b,h,j}) = \log \left( \mathcal{A}_{b,h,i,j} + \sqrt{\mathcal{A}_{b,h,i,j}^2 - 1} \right). \quad (21)$$

**Attention Scores and Weights** Using the computed distances, we calculate the attention scores with a head-specific scaling factor  $\alpha_h$ :

$$\mathbf{A}_{b,h,i,j} = -\frac{d_{\mathbb{D}}(\mathbf{Q}_{b,h,i}, \mathbf{K}_{b,h,j})^2}{\alpha_h \sqrt{D}}. \quad (22)$$

The attention weights are obtained by normalizing the scores using the softmax function:

$$\alpha_{b,h,i,j} = \frac{\exp(\mathbf{A}_{b,h,i,j})}{\sum_{k=1}^N \exp(\mathbf{A}_{b,h,i,k})}. \quad (23)$$

**Output Computation** Each attention head produces an output by aggregating the value vectors, weighted by the attention weights:

$$\mathbf{O}_{b,h,i} = \bigoplus_{j=1}^N (\alpha_{b,h,i,j} \otimes \mathbf{V}_{b,h,j}). \quad (24)$$

The outputs from all heads are combined using Möbius concatenation and transformed with a final linear layer:

$$\mathbf{O}_{b,i} = \mathbf{W}_O \otimes_M \left( \bigoplus_{h=1}^H \mathbf{O}_{b,h,i} \right) \oplus \mathbf{b}_O. \quad (25)$$

DropConnect regularization is applied to the attention weights during training with probability  $p$ .

**Residual Connection with Layer Scaling** As before, we use a residual connection with a learnable scaling parameter  $\beta$ :

$$\mathbf{X}' = \mathbf{X} \oplus \beta \otimes \mathbf{O}. \quad (26)$$

### 3.3.3 HYPERBOLIC FEED-FORWARD NETWORK

The feed-forward network in hyperbolic space consists of two hyperbolic linear layers with a hyperbolic ReLU activation in between:

$$\mathbf{H}_1 = \mathbf{X}' \otimes_M \mathbf{W}_1 \oplus \mathbf{b}_1, \quad (27)$$

$$\mathbf{H}_2 = \text{ReLU}^{\mathbb{D}}(\mathbf{H}_1), \quad (28)$$

$$\mathbf{H}_3 = \mathbf{H}_2 \otimes_M \mathbf{W}_2 \oplus \mathbf{b}_2, \quad (29)$$

followed by a residual connection and hyperbolic layer normalization:

$$\mathbf{X}'' = \text{LayerNorm}^{\mathbb{D}}(\mathbf{X}' \oplus \beta \otimes \mathbf{H}_3). \quad (30)$$

### 3.4 OPTIMIZATION IN HYPERBOLIC SPACE

Optimizing neural networks in hyperbolic space presents unique challenges due to its non-Euclidean nature. We employ several techniques to ensure stable and effective training.

**Gradient Clipping** To prevent large gradients from destabilizing training, we clip gradients in the tangent space:

$$\text{grad}_{\text{clipped}} = \begin{cases} \text{grad}_{\theta} & \text{if } \|\text{grad}_{\theta}\| \leq v, \\ v \cdot \frac{\text{grad}_{\theta}}{\|\text{grad}_{\theta}\|} & \text{otherwise,} \end{cases} \quad (31)$$

where  $\|\cdot\|$  is the Euclidean norm in the tangent space, and  $v$  is the clipping threshold.

**Riemannian Adam Optimizer** We utilize the Riemannian Adam optimizer, which adapts the Adam optimization algorithm to Riemannian manifolds. The update rule is:

$$\theta_{t+1} = \exp_{\theta_t} \left( -\eta_t \cdot \frac{m_t^{\text{clipped}}}{\sqrt{v_t^{\text{clipped}} + \epsilon}} \right), \quad (32)$$

where  $\eta_t$  is the learning rate,  $m_t^{\text{clipped}}$  and  $v_t^{\text{clipped}}$  are the clipped first and second moment estimates, and  $\exp_{\theta_t}$  is the exponential map at  $\theta_t$ .

**Geodesic Distance Regularization** To enhance class separation and encourage meaningful representations, we introduce a regularization term based on geodesic distances:

$$\mathcal{L}_{\text{geo}} = \lambda_{\text{reg}} \cdot \mathbb{E}_{(i,j)} \left[ (1 - \delta_{y_i y_j}) \cdot d_{\mathbb{D}}(\mathbf{x}_i, \mathbf{x}_j) \right], \quad (33)$$

where  $\delta_{y_i y_j}$  is the Kronecker delta function indicating whether samples  $i$  and  $j$  belong to the same class. The total loss objective is  $\mathcal{L}_{\text{cross entropy}} + \mathcal{L}_{\text{geo}}$ .

**Layer Scaling and Attention Scaling** Introducing the learnable scaling parameter  $\beta$  in residual connections and head-specific scaling factors  $\alpha_h$  in the attention mechanism helps control the magnitude of updates. It allows each attention head to adapt its sensitivity to hyperbolic distances.

**Parameter Initialization** Weights are initialized using Xavier uniform initialization adapted for hyperbolic space Leimeister & Wilson (2019). All manifold parameters are initialized within the Poincaré ball to ensure valid representations and stable training.

### 3.5 LIMITATIONS

While the Hyperbolic Vision Transformer improves hierarchical data handling, it incurs higher computational demands due to the complexity of hyperbolic operations. Enhanced representation capabilities and scalability justify this trade-off. Future advancements in hardware and optimization algorithms are expected to mitigate these computational challenges. Additionally, Möbius operations can be approximated to reduce complexity.

## 4 EXPERIMENTS

In this section, we evaluate the performance of our proposed Hyperbolic Vision Transformer (HVT) on the ImageNet-1k dataset. We compare our model with standard Vision Transformers, other state-of-the-art convolutional neural networks, and models incorporating hyperbolic geometry to demonstrate the effectiveness of integrating hyperbolic geometry into vision architectures.

### 4.1 EXPERIMENTAL SETUP

**Dataset** The ImageNet data set Deng et al. (2009) is a large-scale hierarchical image database widely used to benchmark computer vision. It contains over 1.2 million training images and 50,000 validation images categorized into 1,000 classes. Each image is labeled with one of the 1,000 object categories, providing a challenging benchmark for image classification models.

378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431

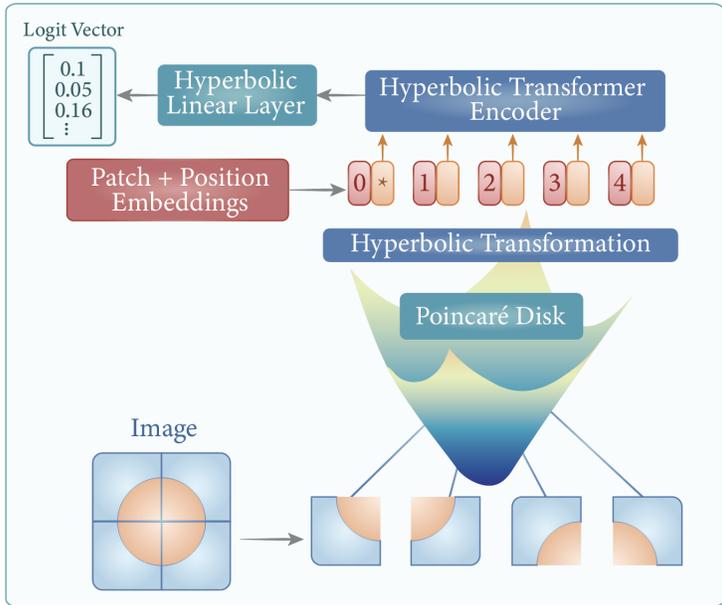


Figure 1: Overall Model Flow

**Implementation Details** We implement our HVT model using PyTorch and the `geoopt` library Kochurov et al. (2020) for operations in hyperbolic space. The architecture of HVT is based on the standard ViT-Base model Dosovitskiy et al. (2021a) with modifications to incorporate hyperbolic geometry in the attention mechanisms and positional encodings.

All models are trained on 8 NVIDIA A100 GPUs using Distributed Data Parallel (DDP). The training hyperparameters are as follows:

Table 2: Training Parameters for the Hyperbolic Vision Transformer (HVT).

Parameter	Value
<b>Optimizer</b>	Riemannian Adam Bécigneul & Ganea (2019) with an initial learning rate of $1 \times 10^{-3}$ .
<b>Batch Size</b>	32.
<b>Epochs</b>	300.
<b>Learning Rate Schedule</b>	Cosine Annealing with a warm-up.
<b>Gradient Clipping</b>	Max norm of 1.0.
<b>Regularization</b>	Geodesic regularization with coefficient $\lambda_{\text{reg}} = 0.001$ .

Data augmentation techniques such as random cropping, horizontal flipping, and color jitter are applied during training. All images are resized to  $224 \times 224$  pixels.

## 4.2 RESULTS AND DISCUSSION

**Model Architecture Comparison** We compare both Base, Large, and Huge architectures to assess the scalability and architectural distinctions between the standard Vision Transformer (ViT) models and our proposed Hyperbolic Vision Transformer (HVT) variants. Table 3 summarizes each model’s vital architectural parameters and the total number of parameters.

As depicted in Table 3, the HVT variants mirror the architectural configurations of their corresponding ViT counterparts in terms of the number of layers, attention heads, hidden dimensions, and MLP dimensions. We notice that the hyperbolic version of has the same number of parameters as its respective ViT variant.

Table 3: Architectural Comparison of ViT and HVT Model Variants.

Model	Version	Layers	Attention Heads	Hidden Dim ( $d$ )	MLP Dim	Parameters (M)
ViT Dosovitskiy et al. (2021a)	Base	12	12	768	3072	86
	Large	24	16	1024	4096	307
	Huge	32	16	1280	5120	632
HVT (Ours)	Base	12	12	768	3072	86
	Large	24	16	1024	4096	307
	Huge	32	16	1280	5120	632

This architectural alignment allows for a direct comparison of performance metrics across models of similar scales, highlighting the efficacy of integrating hyperbolic geometry into vision transformers without substantial increases in model complexity.

**Performance Comparison** To evaluate the performance impact of incorporating hyperbolic geometry, we compare the Top-1 and Top-5 accuracies of both ViT and HVT variants on the ImageNet validation set. Table 4 presents these results.

Table 4: Performance Comparison of ViT and HVT Model Variants on ImageNet.

Model	Version	Top-1 Acc (%)	Top-5 Acc (%)
ViT Dosovitskiy et al. (2021a)	Base	77.9	93.8
	Large	82.5	95.8
	Huge	84.2	96.7
HVT (Ours)	Base	80.1	95.1
	Large	85.0	96.8
	Huge	<b>87.4</b>	<b>97.6</b>

From Table 4, it is evident that each HVT variant consistently outperforms its ViT counterpart across all scales. These performance gains demonstrate the effectiveness of integrating hyperbolic geometry into the Vision Transformer architecture. This enables better capture of hierarchical relationships and more nuanced representations without significantly increasing model complexity.

**Conclusion on Architectural Comparison** The architectural comparisons and performance evaluations indicate that HVT variants leverage hyperbolic geometry to significantly improve standard ViT models across all scales. This enhancement is achieved with the same parameter size, affirming the scalability and efficiency of the HVT architecture for image classification tasks.

**Ablation Study** To analyze the contributions of different components of HVT, we conduct an ablation study as shown in Table 5. We evaluate the impact of hyperbolic positional encoding on the model’s performance.

Table 5: Ablation Study on the Impact of Hyperbolic Components in ViT-Base Model Variants.

Model Variant	Hyper Embeddings	Hyper Attention	Hyper Linear Layers	Top-1 Acc (%)	Top-5 Acc (%)
Baseline ViT-B (Euclidean)	–	–	–	77.9	93.8
+ Hyper Embeddings	✓	–	–	78.4	94.1
+ Hyper Attention	✓	✓	–	79.3	94.7
+ Hyper Linear Layers	✓	✓	✓	<b>80.1</b>	<b>95.1</b>
+ Remove Hyper Embeddings	–	✓	✓	78.6	94.2
+ Remove Hyper Attention	✓	–	✓	79.0	94.5
+ Remove Hyper Linear Layers	✓	✓	–	79.5	94.8

Note: ✓ indicates the presence of the component.

**Conclusion** Our experiments demonstrate that integrating hyperbolic geometry into transformer architectures leads to significant performance improvements on the ImageNet dataset. The HVT

486 model benefits from the ability to model hierarchical structures and capture complex relationships in  
487 image data more effectively than Euclidean-based models.  
488

## 489 5 CONCLUSION AND FUTURE WORK 490

491 In this work, we introduced the Hyperbolic Vision Transformer (HVT), a novel architecture that  
492 integrates hyperbolic geometry into the Vision Transformer (ViT) framework. By leveraging the  
493 properties of hyperbolic space, HVT effectively models complex, hierarchical relationships within  
494 visual data, which is especially beneficial for large-scale image classification tasks like ImageNet that  
495 exhibit such structures.  
496

497 Our extensive experiments on the ImageNet dataset demonstrate that incorporating hyperbolic compo-  
498 nents into the ViT framework can significantly improve performance. HVT consistently outperforms  
499 standard Vision Transformers and state-of-the-art convolutional neural networks, showcasing the  
500 strength of hyperbolic geometry in enhancing deep learning models for vision tasks.

501 This work serves as a foundational step in exploring hyperbolic image classification mechanisms.  
502 The success of our initial experiments suggests that hyperbolic geometry holds great promise for  
503 advancing vision architectures, opening the door to new research opportunities and innovations in  
504 this field.

### 505 5.1 KEY CONTRIBUTIONS 506

507 The key contributions of this paper are as follows:  
508

- 509 • **Hyperbolic Transformer Components:** We extended the Vision Transformer framework  
510 by introducing hyperbolic analogs of critical components, such as positional embeddings,  
511 attention mechanisms, and feed-forward layers.
- 512 • **Learnable Curvature in Positional Embeddings:** Our model employs learnable curvature  
513 in hyperbolic positional embeddings, allowing it to adapt to different levels of complexity in  
514 the data.
- 515 • **Geodesic Regularization and Stability:** We introduced geodesic regularization to better  
516 separate class embeddings in hyperbolic space and employed layer scaling and gradient  
517 clipping to ensure stability during training in hyperbolic space.
- 518 • **Empirical Performance:** We showed that the Hyperbolic Vision Transformer achieves supe-  
519 rior performance on ImageNet, demonstrating the effectiveness of hyperbolic representations  
520 for image classification.  
521

### 522 5.2 FUTURE WORK 523

524 While the Hyperbolic Vision Transformer offers significant improvements in image classification,  
525 several promising research directions remain:

- 526 • **Hybrid Architectures:** Exploring hybrid architectures that combine Euclidean and hyper-  
527 bolic spaces, selectively applying hyperbolic operations where they offer the most benefit  
528 while maintaining standard operations elsewhere.
- 529 • **Optimizing Hyperbolic Training:** Further refining training strategies in hyperbolic space,  
530 such as advanced optimization techniques or dynamic curvature adaptation, to enhance  
531 performance.
- 532 • **Hyperbolic Large Language Models:** The potential of a hyperbolic large language model  
533 is untapped and has potential for increased performance.
- 534 • **Medical Imaging:** Due to the inherently hierarchical structure of medical imaging data, we  
535 foresee improved performance using our proposed method on such data.  
536

537 In conclusion, our work shows that hyperbolic geometry provides a powerful new perspective for  
538 modeling visual data, improving transformer architectures for image classification. The strong results  
539 on ImageNet highlight the potential of this approach, and we look forward to future developments  
and applications of hyperbolic representations in machine learning.

## REFERENCES

- 540  
541  
542 Gregor Bachmann, Gary Bécigneul, and Octavian-Eugen Ganea. Constant curvature graph convolutional networks, 2020. URL <https://arxiv.org/abs/1911.05076>.
- 544 Gary Bécigneul and Octavian-Eugen Ganea. Riemannian adaptive optimization methods. In *International Conference on Learning Representations (ICLR)*, 2019.
- 546  
547 Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives, 2014. URL <https://arxiv.org/abs/1206.5538>.
- 549  
550 Irving Biederman. Recognition-by-components: A theory of human image understanding. *Psychological review*, 94(2):115, 1987.
- 551  
552 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.
- 554  
555 Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers, 2021. URL <https://arxiv.org/abs/2104.14294>.
- 558  
559 Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.
- 561  
562 Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021a.
- 564  
565 Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021b.
- 566  
567 Alaaeldin El-Nouby, Natalia Neverova, Ivan Laptev, and Hervé Jégou. Training vision transformers for image retrieval, 2021. URL <https://arxiv.org/abs/2102.05644>.
- 569  
570 Aleksandr Ermolov, Leyla Mirvakhabova, Valentin Khrulkov, Nicu Sebe, and Ivan Oseledets. Hyperbolic vision transformers: Combining improvements in metric learning, 2022. URL <https://arxiv.org/abs/2203.10833>.
- 572  
573 Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks, 2018. URL <https://arxiv.org/abs/1805.09112>.
- 575  
576 Caglar Gulcehre, Misha Denil, Mateusz Malinowski, Ali Razavi, Razvan Pascanu, Karl Moritz Hermann, Peter Battaglia, Victor Bapst, David Raposo, Adam Santoro, and Nando de Freitas. Hyperbolic attention networks, 2018. URL <https://arxiv.org/abs/1805.09786>.
- 578  
579 Valentin Khrulkov, Leyla Mirvakhabova, Evgeniya Ustinova, Ivan Oseledets, and Victor Lempitsky. Hyperbolic image embeddings, 2020. URL <https://arxiv.org/abs/1904.02239>.
- 581  
582 Max Kochurov, Rasul Karimov, and Serge Kozlukov. Geoopt: Riemannian optimization in pytorch, 2020.
- 583  
584 Matthias Leimeister and Benjamin J. Wilson. Skip-gram word embeddings in hyperbolic space, 2019. URL <https://arxiv.org/abs/1809.01498>.
- 586  
587 Shaoteng Liu, Jingjing Chen, Liangming Pan, Chong-Wah Ngo, Tat-Seng Chua, and Yu-Gang Jiang. Hyperbolic visual embedding learning for zero-shot recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9270–9278, 2020. doi: 10.1109/CVPR42600.2020.00929.
- 588  
589  
590  
591  
592  
593 Maximilian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations, 2017. URL <https://arxiv.org/abs/1705.08039>.

594 Maximilian Riesenhuber and Tomaso Poggio. Hierarchical models of object recognition in cortex.  
595 *Nature neuroscience*, 2(11):1019–1025, 1999.  
596

597 Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz  
598 Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing*  
599 *Systems*, 30, 2017.

600 Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus. Regularization of neural  
601 networks using dropconnect. In Sanjoy Dasgupta and David McAllester (eds.), *Proceedings of*  
602 *the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine*  
603 *Learning Research*, pp. 1058–1066, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL  
604 <https://proceedings.mlr.press/v28/wan13.html>.

605 Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. Hypformer:  
606 Exploring efficient hyperbolic transformer fully in hyperbolic space, 2024. URL [https://](https://arxiv.org/abs/2407.01290)  
607 [arxiv.org/abs/2407.01290](https://arxiv.org/abs/2407.01290).  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647