RETHINKING THE FULLY HYPERBOLIC VISION TRANSFORMER IN POLAR COORDINATES

Anonymous authors
Paper under double-blind review

ABSTRACT

Hyperbolic space can embed intrinsic hierarchies in data with low distortion due to the exponential growth of volume with distance from the origin. However, current Lorentz transformer blocks are formulated in ambient coordinates, where numerical errors increase at large radii due to instability in the Lorentzian inner product, leading many models to limit the radius to avoid this issue. This prevents us from utilizing the regions of hyperbolic space that motivate the geometry. To address this, we revisit the components of the transformer block in polar coordinates, where hyperbolic operations such as distance calculation and attention centroids can be computed without the numerical cancellation in their ambient-coordinate formulations. Specifically, we propose a polar fully connected layer that separately maps an embedding's direction and radius, allowing the radius of a feature to be learned rather than determined by the norm of a linear map. We additionally introduce horospherical shifts as relative positional encodings whose query-key distances grow logarithmically with the token gap. Finally, we reformulate the residual connection as a Lorentz boost, which is an isometry at any radius. Combining these components, we develop a fully hyperbolic transformer that substantially improves performance over Euclidean and hyperbolic baselines on standard vision tasks. We further evaluate our model on ImageNet and demonstrate its ability to generalize to other datasets using pre-trained weights, similar to Euclidean counterparts.

1 INTRODUCTION

The use of hyperbolic geometry has become more prevalent in both the deep learning architectures and the tasks they are applied to. Hyperbolic manifolds are able to embed hierarchical and relational structure with low distortion because the volume of the space grows exponentially with the distance from the origin. This makes it similar to the growth of trees with depth (Krioukov et al., 2010; Sarkar, 2011). However, the mechanism for volume growth also implies that the representational capacity of hyperbolic space greatly benefits from the ability to embed points at a large distance from the origin. But Mishne et al. (2023) show that the different models of hyperbolic space, mainly the Lorentz model and the Poincaré model, can only represent points accurately within certain radii that are fixed by machine precision. For the Lorentz model, this is approximately 19 in double precision and extending their argument to single precision gives us a radius of approximately 8. Beyond this, the Lorentz inner product, distance and weighted centroid operations lose their accuracy and are greatly affected by numerical instability (Mishne et al., 2023; Bdeir et al., 2025).

To avoid this, existing models confine their embeddings to a small radius through strict normalization in the fully connected layer (Chen et al., 2022), norm clipping after projection (Yang et al., 2024; Bdeir et al., 2024), and frequent rescaling (Bdeir et al., 2025). This mitigates the instability but also prevents us from utilizing more of the volume that motivates the geometry. Additionally, this failure mode is silent in the Lorentz model (Mishne et al., 2023) since the operations do not directly become undefined. Instead, they return incorrect points which lead to training degradation or divergence without any component being identifiable as the cause (Alyoussef et al., 2026). Alyoussef et al. (2026) identify this issue in the attention aggregation of hyperbolic transformers in particular and demonstrate the performance improvement of avoiding the operation entirely and replacing it with tangent-space aggregation (Alyoussef et al., 2026; Fein-Ashley et al., 2024).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Additionally, most of the hyperbolic Transformers rely on Euclidean components for positional encodings, residual connections, and layer normalization. These operations are typically performed on the space components of a hyperbolic embedding and then the time component is recomputed from the manifold constraint. Despite being effective, this approach introduces another source of numerical instability. The time component grows exponentially with the radius which makes it subject to cancellation errors when being recomputed from the space values.

Our work introduces Polar-ViT, a hyperbolic transformer that relies on polar form operations, representing each embedding by its geodesic radius and unit direction. When using polar coordinates, we show that the terms that cancel in ambient Lorentz coordinates are never formed, so the weighted centroid and the distance remain accurate at radii past the ambient representation limit. This lessens the need for normalization, clipping and rescaling components and allows us to use the space more effectively. Building on this representation, we additionally propose a polar fully connected layer that uses this representation to map direction and radius as separate channels. In terms of the remaining Euclidean components, we propose intrinsic hyperbolic residual and layer norm alternatives. We also present an argument for augmenting Euclidean positional encodings with a relative encoding derived from horospherical shifts, the parabolic subgroup of the Lorentz group. This approach shows increased model performance while encoding positions without recomputing any time coordinate. We evaluate Polar-ViT at Tiny, Small and Base scales and release ImageNet-1k (Deng et al., 2009) pre-trained weights to demonstrate their ability to transfer to downstream fine-tuning tasks similar to Euclidean Transformers (Dosovitskiy et al., 2021). The main contributions are then as follows:

- Precision analysis: We show that the ambient weighted centroid fails beyond a radius of 8.3 in single precision and derive a polar form that stays accurate up to a radius of about 17.3.
- Polar hyperbolic Transformer (Polar-ViT): A Transformer based on the polar form of the Lorentz model that supports embeddings at higher radii, along with a polar fully connected layer.
- Horospherical positional encodings: A relative positional encoding for hyperbolic attention built from horospherical shifts, which improves over Euclidean encodings alone.
- Intrinsic alternatives to residual connections and Layer Normalization.
- Pre-trained ImageNet-1k weights: Tiny and Small ViT checkpoints will be released, and we show that they transfer effectively to fine-tuning tasks, lowering the cost of adopting hyperbolic models.

2 Related Work

Hyperbolic Neural Networks Early work on hyperbolic embeddings and deep learning used the Poincaré ball model of hyperbolic space (Nickel & Kiela, 2017; Ganea et al., 2018; Shimizu et al., 2021). In parallel, the Lorentz model (Nickel & Kiela, 2018) followed a similar path with hybrid approaches until Chen et al. (2022) proposed their fully hyperbolic network that defined a Lorentz linear layer and other components needed for hyperbolic encoders. The Lorentz model was then used to develop hyperbolic convolutional blocks, residual connections and normalization layers and applied to vision tasks (Bdeir et al., 2024; He et al., 2025d; Shi et al., 2026; van Spengler et al., 2023; Mishra et al., 2026) and natural language tasks (Yang et al., 2025; He et al., 2025a). However, these models still rely on storing the embeddings in ambient coordinates, which limits the radius that keeps Lorentzian operations well-defined (Mishne et al., 2023).

Hyperbolic Transformers Hyperbolic attention networks (Gulcehre et al., 2019) were the first to calculate attention scores using distances on the hyperboloid. Chen et al. (2022) then kept the whole attention layer in the Lorentz model. They replaced the distance scores with squared distances and replaced the Einstein midpoint with the weighted Lorentzian centroid (Law et al., 2019). As for vision tasks, HVT (Fein-Ashley et al., 2024) replaced the attention and feed-forward layers with Möbius operations in the Poincaré ball and trained on ImageNet-1k. For the Lorentz model, LViT from HyperCore (He et al., 2025c) builds a fully hyperbolic Lorentz ViT using existing Lorentzian components. They are also the first to train a fully hyperbolic ViT on ImageNet-1k, but they use non-standard fine-tuning tests (using HypLoRA (Yang et al., 2025)) to transfer to CIFAR and Tiny-ImageNet. HexFormer (Alyoussef et al., 2026) argues for tangent space aggregation in the attention

layer due to the numerical precision issues with the Lorentzian centroid. In comparison, our method keeps the weighted centroid by formulating it in a more exact setting and introduces improved hyperbolic components to increase performance. Additionally, none of the previous ViTs release ImageNet-1k weights, and a recent survey (He et al., 2025b) lists pre-trained hyperbolic vision models as still missing and stresses their importance for easier adoption.

Numerical Stability of Hyperbolic Representations The number of bits needed to store a hyperbolic point grows with its radius (Sala et al., 2018). Prior fixes fall into three groups. The first group attempts to increase the precision through tiling the space (Yu & De Sa, 2019) or through multi-component floats (Yu & De Sa, 2021; Yu et al., 2022) which bound the error at any radius. The second group works on bounding the radius through clipping features before the exponential map (Guo et al., 2022), fixing a radius threshold (Skopek et al., 2020), normalizing inside the linear layer (Chen et al., 2022), or scaling the embeddings according to the largest representable distance for the current curvature value and precision (Bdeir et al., 2025). The third group changes the parametrization entirely. This is done through storing points in the tangent space at the origin (Mathieu et al., 2019). Closest to our parametrization, several of the graph embedding methods that Pérez Casulo et al. (2026) unify in one framework fit node positions directly in polar coordinates, so the polar form has to cover only distances. Polar-ViT extends the use of polar coordinates to the main operations of a transformer, including the linear layer, attention and more importantly, the centroid.

Positional Encodings Relative positional encodings make the attention score depend on the index gap (Shaw et al., 2018; Press et al., 2022). Rotary position embedding (Su et al., 2024) rotates each query and key by an angle that grows with its index, so the inner product of the pair depends only on their relative position. Two recent works replace the rotation of rotary encoding with a Lorentz transformation. HELM (He et al., 2025a) rotates the space coordinates and recomputes the time coordinate, and HoPE (Dai et al., 2025) applies a Lorentz boost to the Euclidean queries and keys of a standard language model. Our approach uses the third, horospherical shifts, which slide points along horospheres around a fixed ideal point. Horospheres have previously been used for prototypes (Ghadimi Atigh et al., 2021), decision boundaries (Fan et al., 2023), dimensionality reduction (Chami et al., 2021) and the linear layer (Chen et al., 2026), but not for positional encoding.

3 The Lorentz Model

Ambient coordinates. The n-dimensional Lorentz model with curvature $-1/k$ is the upper sheet of the hyperboloid defined as $\mathbb{L}_k^n = \{x \in \mathbb{R}^{n+1} : \langle x, x\rangle_{\mathcal{L}} = -k,\ x_0 > 0\}$. Here, the hyperbolic point is $x = (x_0, \mathbf{x})$ with time coordinate x_0 and space coordinates $\mathbf{x} \in \mathbb{R}^n$, and $\langle x, y\rangle_{\mathcal{L}} = -x_0 y_0 + \mathbf{x} \cdot \mathbf{y}$ is the Lorentzian inner product. Based on the definitions, the time coordinate can be computed from the space coordinates as $x_0 = \sqrt{k + \|\mathbf{x}\|^2}$. Applying this to a space vector of all 0s gives the origin $o = (\sqrt{k}, 0, \ldots, 0)$. Finally, the Lorentzian distance $d_{\mathcal{L}}(x, y)$ is the length of the geodesic connecting two points on the hyperboloid x and y,

$$d_{\mathcal{L}}(x, y) = \sqrt{k}\ \operatorname{arcosh}\big(-\langle x, y\rangle_{\mathcal{L}}/k\big). \tag{1}$$

Polar form. Instead of the standard time and space coordinate representation, we can determine a point by its polar form $(r, \mathbf{u})$, where $r = d_{\mathcal{L}}(o, x)$ is the distance from the origin and $\mathbf{u} = \mathbf{x}/\|\mathbf{x}\|$ is the unit direction of its space coordinates. Applying Eq. (1) to the origin gives $r = \sqrt{k}\operatorname{arcosh}(x_0/\sqrt{k})$, so $x_0 = \sqrt{k}\cosh(r/\sqrt{k})$, and the constraint $x_0^2 - \|\mathbf{x}\|^2 = k$ then gives $\|\mathbf{x}\| = \sqrt{k}\sinh(r/\sqrt{k})$. We can therefore convert back to ambient coordinates by

$$x = \Big(\sqrt{k}\cosh\tfrac{r}{\sqrt{k}},\ \sqrt{k}\sinh\tfrac{r}{\sqrt{k}}\,\mathbf{u}\Big), \tag{2}$$

and from ambient to polar by $r = \sqrt{k}\ \operatorname{arsinh}(\|\mathbf{x}\|/\sqrt{k})$ and $\mathbf{u} = \mathbf{x}/\|\mathbf{x}\|$. Throughout, we write $a = r/\sqrt{k}$ for the rescaled radius, so that $x = (\sqrt{k}\cosh a, \sqrt{k}\sinh a\,\mathbf{u})$. Both ambient coordinates grow as $e^a/2$, so their rounding error grows as εe^a for a unit roundoff ε. The radius and the direction, in contrast, are stored to relative precision ε at every radius. Polar-ViT stores every embedding in polar form $(r, \mathbf{u})$ and evaluates the sensitive operations in it.

3.1 Precision Loss in Ambient Coordinates

Most Lorentz operations rely on the Lorentzian inner product, either directly (distance, logarithmic map, parallel transport) or through a derived form (centroid). Mishne et al. (2023) bound the radius at which the coordinates of a single point can still be stored accurately. We extend this analysis to the distance and the centroid calculation, the two operations Lorentzian attention heavily relies on. In the following, we denote ε as the unit roundoff: 2^{-24} in single precision, 2^{-53} in double precision and 2^{-8} in bfloat16.

Cancellation in the inner product. Consider two points x and y on the hyperboloid with rescaled radii a and b, and whose directions form an angle θ. Then Eq. (2) gives

$$\frac{x_0 y_0}{k} = \cosh a \cosh b, \qquad \frac{\mathbf{x}\cdot\mathbf{y}}{k} = \sinh a \sinh b \cos\theta, \qquad -\frac{\langle x, y\rangle_{\mathcal{L}}}{k} = \cosh a \cosh b - \sinh a \sinh b \cos\theta. \tag{3}$$

The first two terms both grow as $e^{a+b}/4$, while their difference, $\cosh(d_{\mathcal{L}}(x,y)/\sqrt{k})$, stays close to one for nearby points. Each product is rounded to the relative precision ε, so the computed inner product carries an absolute error of around $\varepsilon e^{a+b}/4$. This error is independent of $d_{\mathcal{L}}(x,y)$ and depends mainly on the radii of the points. Mishne et al. (2023) give the simplest instance of this effect. For a point at radius ≈ 19.1 and $k=1$, $x_0 = 10^8$, $x_0^2 = 10^{16}$, and in double precision $x_0^2 - 1$ rounds to x_0^2, so $\langle x, x\rangle_{\mathcal{L}}$ returns 0 instead of -1.

Distance. By Eq. (1), we can find the normalized inner product with respect to the Lorentzian distance as $-\langle x, y\rangle_{\mathcal{L}}/k = \cosh(d_{\mathcal{L}}(x,y)/\sqrt{k})$, which is always at least one. We define the term $c = \cosh(d_{\mathcal{L}}(x,y)/\sqrt{k}) - 1$ as the difference from this minimum amount. Since $\cosh t - 1 \approx t^2/2$ for small t, nearby points have $c \approx d_{\mathcal{L}}^2(x,y)/2k$, so all information about the distance is based on this c. The absolute rounding error of the inner product is added directly to c. For two points at the same radius a this resolves to $\varepsilon e^{2a}/4$. As such we can see that once the error is larger than c itself, the computed distance is meaningless.

For example, take $k = 1$ and two points at radius $a = 10$ that are $d_{\mathcal{L}}(x,y) = 1$ apart. Their ambient coordinates are $x_0 = y_0 = \cosh 10$ and $\|\mathbf{x}\| = \|\mathbf{y}\| = \sinh 10$. The ambient operations evaluate $-\langle x, y\rangle_{\mathcal{L}} = x_0 y_0 - \mathbf{x}\cdot\mathbf{y}$, where $x_0 y_0 \approx 1.2129 \times 10^8$ and $\mathbf{x}\cdot\mathbf{y} \approx 1.2129 \times 10^8$. Here the true difference should be $\cosh 1 = 1.54$. In single precision each of these two products is rounded to about seven significant digits, so each carries an absolute error of order $2^{-24} \times 1.2 \times 10^8 \approx 7$. If we perform the subtraction in single precision we get around 24 and this is justified because the error is the same size as the error on the inputs. This value is several times larger than 1.54 and the distance between x and y becomes $\operatorname{arcosh} 24 = 3.87$ instead of 1. At $a = 12$ the two products are e^4 times larger and their errors reach about 400. The subtraction returns 0, and $\operatorname{arcosh} 0$ is NaN.

Proposition 1 (Ambient failure radius)**.** *In ambient coordinates with unit roundoff ε, the rounding error on the computed distance between two points at radius r exceeds the distance itself, for distances of order $\sqrt{k}$, once $r \geq r_\varepsilon = \frac{\sqrt{k}}{2}\ln(1/\varepsilon)$. In the case of $k = 1$, this is $r_\varepsilon = 8.3$ in single precision, 18.4 in double precision and 2.8 in bfloat16.*

The double-precision value agrees with the bound of Mishne et al. (2023), and the single-precision value agrees with the radius at which we measure a 100% error in the ambient distance (Appendix E.4). The logarithmic map, parallel transport and the tangent inner product of Riemannian optimizers use the same inner product and fail at the same radius.

Centroid. The weighted centroid (Law et al., 2019) rescales the weighted sum of its inputs onto the hyperboloid,

$$\mu(x, w) = \sqrt{k}\,\frac{\sum_i w_i x_i}{\left\|\sum_i w_i x_i\right\|_{\mathcal{L}}}, \qquad \|v\|_{\mathcal{L}} = \sqrt{|\langle v, v\rangle_{\mathcal{L}}|}. \tag{4}$$

This operation is important because it aggregates the values in Lorentz attention (Chen et al., 2022; Yang et al., 2024), and it is used to calculate the mean in Lorentz batch normalization (Bdeir et al., 2024), the Lorentzian residual (He et al., 2025d) and the manifold weight decay of Riemannian AdamW (Bdeir et al., 2025). However, the denominator is based on the ambient inner product and is then also subject to silent or absolute failure.

Let $T = \sum_i w_i \cosh a_i$ and $\mathbf{V} = \sum_i w_i \sinh a_i \, \mathbf{u}_i$ be the time and space parts of the weighted sum, divided by $\sqrt{k}$. The square of the denominator is then $T^2 - \|\mathbf{V}\|^2$. Ambient operation computes T^2 and $\|\mathbf{V}\|^2$ separately and since the size of each is about $e^{2a}/4$, their difference carries an error of about $\varepsilon e^{2a}/4$. The true difference is much smaller since it is actually of order e^a times a quantity that shrinks as the points move outward and as their directions align. Once the rounding error reaches the true value, the computed normalizer is noise.

For example, manifold weight decay pulls a parameter x towards the origin o by taking the centroid of $\{x, o\}$ with weights $(1 - w, w)$, where w is small (Bdeir et al., 2025). Take $k = 1$, $w = 10^{-5}$ and x at radius $a = 8$. The step should move x towards the origin to the radius 7.985. However, the two squared terms are $T^2 \approx \|\mathbf{V}\|^2 \approx 2.2 \times 10^6$ and single precision returns the difference as 0.75, which puts the output at radius 8.14. This means that the decay step pushed the parameter in the opposite direction. At $a = 9$ both squares round to an identical value, so the subtraction returns 0 and the division returns NaN.

3.2 Polar Operations and Their Precision

Every failure mode above shares the same pattern: two large ambient quantities are computed separately and then subtracted. In polar form, we expand each operation before evaluating it, so that the large parts cancel out, and the operation becomes a sum of non-negative terms.

Distance formulation. Subtracting one from Eq. (3) and using $1 - \cos\theta = \frac{1}{2}\|\mathbf{u} - \mathbf{v}\|^2$ gives the hyperbolic law of haversines,

$$c = \cosh\big(d_{\mathcal{L}}(x,y)/\sqrt{k}\big) - 1 = 2\sinh^2\big(\tfrac{a-b}{2}\big) + \sinh a \sinh b \, \tfrac{1}{2}\|\mathbf{u}-\mathbf{v}\|^2, \qquad d_{\mathcal{L}}(x,y) = \sqrt{k}\,\operatorname{arsinh}\sqrt{c\,(c+2)}. \tag{5}$$

The two terms depend only on the radial and the angular gap and do not involve subtraction. This identity is classical, and we use it in place of the ambient inner product. Appendix D shows that a closed-form gradient ensuring the term is not evaluated as $1 - \mathbf{u} \cdot \mathbf{v}$ does not incur any extra computational cost or memory footprint over ambient.

Centroid formulation. The centroid has no such classical form, so we derive one. We factorize the squared denominator as $T^2 - \|\mathbf{V}\|^2 = (T - \|\mathbf{V}\|)(T + \|\mathbf{V}\|)$. The second factor is a sum of non-negative terms which is not problematic. We decompose the first factor using $\hat{\mathbf{u}} = \mathbf{V}/\|\mathbf{V}\|$ as:

$$T - \|\mathbf{V}\| = \underbrace{\textstyle\sum_i w_i e^{-a_i}}_{E} + \underbrace{\textstyle\sum_i w_i \sinh a_i \, \frac{1}{2}\|\mathbf{u}_i - \hat{\mathbf{u}}\|^2}_{S}. \tag{6}$$

Here, E is a radial term that shrinks as the points move outward, and S is an angular term that measures the spread of the directions of the inputs and vanishes when they coincide. The true squared denominator is then $\approx (E + S)\, e^a$, and comparing it with the ambient error of $\varepsilon e^{2a}/4$ shows that the ambient centroid is only correct given $E + S \gtrsim \varepsilon\, e^a$. Both factors, $T + \|\mathbf{V}\|$ and $E + S$, are sums of non-negative terms which are now more stable, and the polar centroid computes them directly from the radii and directions. Its output has a direction of $\hat{\mathbf{u}}$ and a rescaled radius of

$$\sinh a_\mu = \frac{\|\mathbf{V}\|}{\sqrt{(E+S)(T+\|\mathbf{V}\|)}}. \tag{7}$$

Tangent vectors and precision bound. The tangent-space inner product used by Riemannian optimizers can also be reduced to a plain sum in the same way, so the optimizer also works in polar form. We show in Proposition 2 in Appendix B that every polar operation is accurate to the limit of the accuracy of the stored points. This gives a radius of around $r \approx 17.3$ in single precision (against 8.3 for ambient) and 6.2 in bfloat16 (against 2.8).

4 Methodology

Polar-ViT is a hyperbolic Vision Transformer that stores embeddings in polar form and evaluates the sensitive operations with the more stable formulations proposed in Section 3.2. Image patches are first mapped onto the manifold and embedded by a polar fully connected layer, and a learned

absolute position table is added to the space coordinates (Alyoussef et al., 2026). Each of the L Transformer blocks applies gyro layer normalization, a multi-head attention whose queries and keys carry the horospherical positional encoding, and a two-layer MLP. Boost residuals are used to maintain the signal. Finally, the output tokens are pooled by their polar centroid, as in LViT (He et al., 2025c), and classified by a hyperbolic multinomial logistic regression (Bdeir et al., 2024).

4.1 Polar Fully Connected Layer

The Lorentz fully connected layer of Chen et al. (2022) uses standard matrix multiplication on the input with a learned weight. In its general form, it then recomputes the time coordinate from the manifold constraint, so a single map outputs both the direction and the radius. In the form they use in practice, a separate sigmoid of a linear function of the input sets the time coordinate, and the space coordinates are rescaled to match, so the radius is bounded by a fixed constant. In a layer expressivity test, van der Klis et al. (2026) show that the necessary number of optimization steps the general form needs to reach a target radius grows exponentially with the distance from the origin. This is due to the logarithmic scaling of the output hyperbolic norms with respect to the number of gradient descent steps.

Our Polar-FC avoids both the slow growth and the fixed bound by giving the radius its own unbounded parametrization. As in the Lorentz layer, we multiply the input with a learned map $\mathbf{W}$ and apply the optional activation ϕ to produce $\mathbf{h}$. The output direction is then the direction of $\mathbf{h}$. We then parametrize the output radius with a scalar ℓ by taking it to be the logarithm of $\sinh a'$. This gives us:

$$\mathbf{h} = \phi\big(\mathbf{W}x/\sqrt{k}\big), \qquad \mathbf{u}' = \mathbf{h}/\|\mathbf{h}\|, \qquad a' = \operatorname{arsinh}\big(e^{\ell}\big), \quad \ell = \beta \ln\big(\|\mathbf{h}\|/\sqrt{k}\big) + \lambda a + \boldsymbol{\varphi}^{\top}\boldsymbol{\chi} + \varphi_0. \tag{8}$$

Here $\boldsymbol{\chi} = \tanh(a)\,\mathbf{u}$ is the Klein vector of the input, which equals the space coordinates divided by the time coordinate. We use this term because we want to include the input direction $\mathbf{u}$ in the radius computation but $\mathbf{u}$ has a gradient of $1/\|\mathbf{x}\|$ at the origin while $\boldsymbol{\chi}$ remains smooth there. The log radius ℓ then has three weighted contributions from the output norm controlled by β, the input radius controlled by λ, and the input direction controlled by $\boldsymbol{\varphi}$, plus a bias φ_0. This last term is a single linear unit that lets the layer push points inward or outward depending on where they point, which the direction map $\mathbf{W}$ cannot do since it only sets $\mathbf{u}'$. At initialization, we set $\beta = 1$, $\lambda = \varphi_0 = 0$, and $\boldsymbol{\varphi} = 0$, which reduces Eq. (8) to the general form of the Lorentz fully connected layer. For an input on $\mathbb{L}_k^n$ and an output on $\mathbb{L}_k^{n'}$, the number of parameters becomes $n' \times (n+1) + n + 3$ instead of $n' \times (n+1)$, which is a negligible overhead.

We repeat the expressivity test of van der Klis et al. (2026) in Appendix E.3 and show that the Polar-FC reaches the target radius in both Float64, the original testing precision, and Float32, which is the precision we use in our experiments. Additionally, we are able to achieve this without the need for clipping which increases the convergence speed dramatically. It should be noted that we had to use the polar distance formulation for the loss in both scenarios since the ambient distance would either fail or report false convergence. We note that FGG-LNN (van der Klis et al., 2026) was also able to reach the target radius with its hyperplane-based formulation, but it was less stable without gradient clipping. The instability showed only at radii past 13 so we still included it in the ablation study in Appendix E.1.

4.2 Attention with Polar Centroid Aggregation

We calculate the queries, keys and values using the proposed Polar-FC layers. For multi-head attention, we follow the Lorentz Transformer of Chen et al. (2022). In their work, the space coordinates are split into H slices, each with its own time coordinate, so that every head works on $\mathbb{L}_k^{n/H}$. The head outputs are later merged using Lorentz direct concatenation. We follow this process with its polar coordinate equivalent operations. The score s_{ij} between query q_i and key κ_j is calculated as the negative squared distance, computed with Eq. (5), and the values are aggregated using the polar centroid:

$$s_{ij} = -\frac{\gamma}{\tau}\, d^2_{\mathcal{L}}(q_i, \kappa_j), \qquad \mathbf{p}_i = \operatorname{softmax}_j(s_{ij}), \qquad z_i = \mu_p\big(v, \mathbf{p}_i\big), \tag{9}$$

where $\gamma = (n/H)^{-1/2}$ is a scale parameter and τ is the learned temperature, and μ_p is the polar centroid function. We fuse the score, the softmax and the centroid into a single kernel that, like FlashAttention (Dao et al., 2022), never stores the $N \times N$ score matrix for N tokens.

4.3 Horospherical Positional Encoding

Rotary position embedding (Su et al., 2024) multiplies the query and key at position m by a rotation $\Phi(m) = \exp(mA)$. Rotations preserve the inner product, so the score between positions m and m' depends only on the offset $m' - m$. The same argument holds on the hyperboloid for any one-parameter group of Lorentz isometries $\Phi(m) = \exp(mA)$, with A in the Lorentz Lie algebra: the Lorentz inner product is preserved, $\langle \Phi(m)\, q, \Phi(m')\, \kappa \rangle_{\mathcal{L}} = \langle q, \Phi(m' - m)\, \kappa \rangle_{\mathcal{L}}$, and the encoded points stay on the manifold. Prior work has used two of the three basic kinds of one-parameter subgroups: rotations of Lorentz embeddings (He et al., 2025a) and boosts of Euclidean queries and keys (Dai et al., 2025). We use the third, horospherical shifts, which fix an ideal point and slide each horosphere centered at it along itself. On the time axis and two reserved space axes e_1, e_2, the shift by t is

$$\nu = x_0 - x_1, \qquad x_2 \leftarrow x_2 + t\,\nu, \qquad x_0 \leftarrow x_0 + t\,x_2 + \tfrac{1}{2}t^2\nu, \qquad x_1 \leftarrow x_1 + t\,x_2 + \tfrac{1}{2}t^2\nu, \tag{10}$$

which leaves the Busemann coordinate ν towards e_1 unchanged. In polar form we evaluate $\nu = \sqrt{k}\,\big(e^{-a} + \sinh a\,(1 - u_1)\big)$, a sum of non-negative terms, so the shift never subtracts two large coordinates. The query-key distance is bounded in t under a rotation and linear under a boost. Under a horospherical shift it is

$$d_{\mathcal{L}}\big(x, \Phi(t)\, x\big) = 2\sqrt{k}\ \operatorname{arsinh}\Big(\frac{t\,\nu}{2\sqrt{k}}\Big), \tag{11}$$

which grows logarithmically, so attention separates nearby positions sharply and decays slowly at long range. We set $t = m\,\rho_h/M$ for the token at position m, with M the number of positions along the axis and ρ_h a learned rate per head, so that each head learns how local its attention is. The encoding adds $2H$ parameters per layer; Appendix C derives Eq. (11) and extends the encoding to patch grids.

4.4 Normalization and Residuals

Gyro layer normalization. Let $x_{b,i} \in \mathbb{L}^n_k$ be the i-th of the P tokens of sample b. For each sample we take the Lorentz centroid (Law et al., 2019) and dispersion of its own tokens,

$$\bar{x}_b = \mu_p\big(x_b, \tfrac{1}{P}\mathbf{1}\big), \qquad \sigma_b^2 = \frac{1}{P}\sum_{i=1}^{P} d^2_{\mathcal{L}}(x_{b,i}, \bar{x}_b),$$

and normalize every token of that sample with them. Following the gyro batch normalization of Chen et al. (2025), as adopted by Shi et al. (2026), each token is centered by gyroaddition, rescaled by the dispersion and a learned gain $\gamma_{\mathrm{LN}} = e^{\lambda_{\mathrm{LN}}}$, and translated by a learned bias point $\beta_{\mathrm{LN}} \in \mathbb{L}^n_k$:

$$\hat{x}_{b,i} \;=\; \beta_{\mathrm{LN}} \oplus \Big(\tfrac{\gamma_{\mathrm{LN}}}{\sqrt{\sigma_b^2 + \epsilon}} \otimes \big((\ominus \bar{x}_b) \oplus x_{b,i}\big)\Big).$$

Gyro batch normalization computes $\bar{x}$ and σ^2 over the batch axis b and keeps running estimates of them; here they are computed over the token axis i of each sample, so the layer carries no running state and no coupling between samples. In polar form the scalar multiplication $\otimes$ acts on the radius alone, $r \mapsto \gamma_{\mathrm{LN}}\, r/\sqrt{\sigma_b^2 + \epsilon}$, and leaves the direction unchanged. The gain is initialized so that the output radius matches that of a Euclidean layer norm on the space coordinates, $\gamma_{\mathrm{LN},0} = \sqrt{k}\ \operatorname{arsinh}\sqrt{n/k}$; the bias is initialized at the origin.

Boost residual. Lorentz Transformers form residuals by adding space coordinates and recomputing the time coordinate (Chen et al., 2022), or through Euclidean residual connections in the tangent space (Alyoussef et al., 2026). Both of these approaches require a radius cap to prevent the embeddings from drifting too far from the origin. Intrinsic alternatives already exist. Gyroaddition residuals $x \oplus f(x) = \exp_x\big(\mathrm{PT}_{o\to x}(\log_o f(x))\big)$ (van Spengler et al., 2023; Shi et al., 2026) parallel transport the block output from the origin to x and apply the exponential map there, and Riemannian

residual networks move x along the exponential map of a tangent vector at x (Katsman et al., 2023). We instead apply the residual update to x as boosts along its own direction and perpendicular to it, and center the radial step over the tokens of each sample, so that the residual cannot move the tokens outward on average and needs no radius cap.

Consider a residual connection $x \mapsto x + f(x)$, where x is the token embedding entering the block and $f(x)$ is the block output. Let the polar coordinates of the embedding with rescaled radius be $x = (a, \mathbf{u})$ and similarly $f(x) = (b, \mathbf{v})$. In terms of the residual connection, we can say that the block output proposes moving x a step with signed distance b in the direction $\mathbf{v}$. We split this step relative to the current direction $\mathbf{u}$ where the part along $\mathbf{u}$ has a signed length of $b_\parallel = b\,(\mathbf{v} \cdot \mathbf{u})$, and the remainder $b\,\mathbf{v} - b_\parallel \mathbf{u}$ has a length of $b_\perp$ and a unit direction $\hat{\mathbf{v}}_\perp$. The residual update can then be written as

$$a' = a + \alpha\,(b_\parallel - \bar{b}_\parallel), \qquad \mathbf{u}' = \frac{\tanh(a)\,\mathbf{u} + \sinh(\alpha b_\perp)\,\hat{\mathbf{v}}_\perp}{\left\| \tanh(a)\,\mathbf{u} + \sinh(\alpha b_\perp)\,\hat{\mathbf{v}}_\perp \right\|}, \tag{12}$$

with a learned step size α initialized at 0.1, and $\bar{b}_\parallel$ the mean of $b_\parallel$ over the tokens of the sample. The first equation is a boost of rapidity $\alpha(b_\parallel - \bar{b}_\parallel)$ along $\mathbf{u}$, which moves the input embedding along its own geodesic. Without the centering, the radial step grows with the radius of the block output, and the radius grows geometrically with depth when the block output and the stream point the same way. With it, the radial steps of a sample sum to zero, so the residual moves radius between tokens but keeps their mean radius unchanged. The second equation is the direction that x takes after a boost of rapidity $\alpha b_\perp$ is applied along $\hat{\mathbf{v}}_\perp$. This tilts $\mathbf{u}$ towards $\hat{\mathbf{v}}_\perp$, but the tilt is smaller for larger a since $\tanh(a) \to 1$. We take the radius from the first boost and the direction from the second, so the update as a whole is not a single boost, and the perpendicular step does not change the radius.

5 Experiments

We evaluate Polar-ViT on image classification from scratch, on ImageNet-1k and on transfer to fine-grained classification. While this work focuses on vision transformers, the Polar-FC and the polar form of the embeddings are not limited to transformers so we test a polar U-Net based on the setup of Mishra et al. (2026). It matches the Euclidean U-Net and beats hyperbolic baselines in mean Dice on almost all of the five medical segmentation datasets (Appendix F).

5.1 Image Classification from Scratch

We train Polar-ViT on CIFAR-100 (Krizhevsky, 2009) and Tiny-ImageNet (Le & Yang, 2015) at the two model sizes of HexFormer (Alyoussef et al., 2026), ViT-Tiny and ViT-Base, for 100 epochs with the HexFormer training recipe, and report the test accuracy at the last epoch. The results of the Euclidean ViT, HexFormer and HexFormer-Hybrid are taken from Alyoussef et al. (2026). We run our model in their codebase to match the recipe, and we take HVT (Fein-Ashley et al., 2024) and LViT (He et al., 2025c) from He et al. (2025c). Appendix D gives the architectures and the training protocol.

Table 1 shows the results. On CIFAR-100 with ViT-Tiny, Polar-ViT is 4.3% above the strongest baseline, HexFormer-Hybrid, and 5.5% above the Euclidean ViT. On Tiny-ImageNet, Polar-ViT also outperforms HexFormer-Hybrid by 4.3%. At the Base size, where the Euclidean ViT overfits under this recipe and falls below ViT-Tiny (Alyoussef et al., 2026), it is 4.8% above HexFormer-Hybrid on CIFAR-100 and it reaches 68.19 on Tiny-ImageNet, compared to 53.01 for LViT. Appendix E isolates the contribution of each component, compares rules for the output radius of the polar layer and reports the numerical precision and the computational cost.

5.2 ImageNet-1k

We train a Euclidean ViT and Polar-ViT on ImageNet-1k (Deng et al., 2009) at the ViT-Tiny (2.9M parameters) and ViT-Small (21.7M parameters) sizes. All four models are trained for 90 epochs with the recipe of Beyer et al. (2022), and at each size the two models have the same depth, width, number of heads and patch size (Appendix D.4). We report the mean and standard deviation of the top-1 validation accuracy over three seeds. Table 2 shows that Polar-ViT is 3.5% above the Euclidean ViT at the Tiny size and 2.2% above it at the Small size.

Table 1: Top-1 test accuracy on CIFAR-100 and Tiny-ImageNet, results averaged over 10 seeds. Baseline rows are taken from Alyoussef et al. (2026) (Euclidean ViT, HexFormer) and He et al. (2025c) (HVT, LViT), which reports single values.

Model	Size	CIFAR-100	Tiny-ImageNet
Euclidean ViT	Tiny	74.64 ± 0.41	60.74 ± 0.36
HexFormer	Tiny	75.65 ± 0.43	60.87 ± 0.64
HexFormer-Hybrid	Tiny	75.85 ± 0.34	62.93 ± 0.45
Polar-ViT w/o horoshift	Tiny	$\underline{78.04 \pm 0.53}$	$\underline{65.12 \pm 0.47}$
Polar-ViT	Tiny	$\mathbf{80.15 \pm 0.26}$	$\mathbf{67.23 \pm 0.32}$
HVT (Fein-Ashley et al., 2024)	Base	42.77	40.12
LViT (He et al., 2025c)	Base	69.11	53.01
Euclidean ViT	Base	70.54 ± 0.38	—
HexFormer	Base	76.49 ± 1.21	—
HexFormer-Hybrid	Base	76.81 ± 0.51	—
Polar-ViT w/o horoshift	Base	$\underline{80.33 \pm 0.57}$	$\underline{67.06 \pm 0.31}$
Polar-ViT	Base	$\mathbf{81.63 \pm 0.61}$	$\mathbf{68.19 \pm 0.28}$

Table 2: Top-1 accuracy on ImageNet-1k. Results are averaged over three seeds.

Model	Size	Top-1
Euclidean ViT	Tiny	68.19 ± 0.21
Polar-ViT	Tiny	$\mathbf{71.68 \pm 0.11}$
Euclidean ViT	Small	75.61 ± 0.13
Polar-ViT	Small	$\mathbf{77.79 \pm 0.10}$

Table 3: Top-1 accuracy after fine-tuning IN-1K models.

Model	Size	CUB	Aircraft
Euclidean ViT	Tiny	75.65 ± 1.01	65.17 ± 0.71
Polar-ViT	Tiny	$\mathbf{79.53 \pm 0.82}$	$\mathbf{70.48 \pm 0.50}$
Flat-ViT	Small	$\mathbf{84.69 \pm 0.31}$	$\mathbf{80.58 \pm 0.44}$
Euclidean ViT	Small	79.77 ± 0.64	76.47 ± 0.73
Polar-ViT	Small	$\underline{81.61 \pm 0.22}$	$\underline{80.03 \pm 0.49}$

5.3 Transfer to Fine-Grained Classification

We test whether the ImageNet-1k checkpoints of Section 5.2 transfer to fine-grained recognition on CUB-200-2011 (Wah et al., 2011) (CUB, 200 species) and FGVC-Aircraft (Maji et al., 2013) (Aircraft, 100 variants). These are the two benchmarks used by H-CAST (Park et al., 2025) and fine-tuned with their recipe (Park et al., 2025). we report the mean and standard deviation of the top-1 test accuracy at the last epoch over three seeds. For reference, we also fine-tune the DeiT-S checkpoint of the Flat-ViT baseline of Park et al. (2025) with the same recipe. Appendix D.4 gives the datasets and the fine-tuning recipe.

Table 3 shows that Polar-ViT outperforms the Euclidean ViT at both sizes. At the Tiny size, it is 3.9% above the Euclidean ViT on CUB and 5.3% above it on Aircraft. Flat-ViT however starts from a DeiT-S checkpoint that is trained for 300 epochs and reaches 79.9%. It is 3.1% above Polar-ViT on CUB and 0.6% above it on Aircraft. This shows that the stronger DeiT backbone is still more important than the geometry of the model.

6 Conclusion

Overall, we show that the weighted centroid and the distance of the Lorentz model fail in single precision and propose more stable polar forms. We use this to build Polar-ViT, with a fully connected layer that maps direction and radius separately, a relative positional encoding from horospherical shifts, and a residual connection that is an isometry at any radius. The model outperforms Euclidean and hyperbolic baselines on standard vision benchmarks and ImageNet-1k, and its ImageNet-1k checkpoints transfer to fine-grained classification. More work is needed for the finetuning process however and studying the sensitivity of the model to the pretraining recipe and hyperparameters.

7 STATEMENTS

AI USE STATEMENT

In this work, we used generative AI tools for two tasks with required disclosure. First, we used to assist in the writing of proofs. Specifically we derived the arguments for Propositions 1 and 2 and AI tools were used to translate the derivations in LaTeX and to suggest wording for the proof text. Every step of the LaTeX version was checked against the derivation. We also asked AI tools to comment on the experimental plan, such as which baselines and ablations would be useful to include. We factored these in but ultimately the final experimental design was decided by us. Additionally, we used generative AI tools for tasks with recommended disclosure: we used it to edit the text for readability and grammar, to search for and summarize related work when checking the novelty of our contributions, and to help locate and fix bugs in our written code. We have reviewed all AI-assisted work, and we read the references suggested by the AI tool and verified them against the cited papers. This is the same for debugging code produced. We take responsibility for the final content produced with the aid of generative AI.

ETHICS STATEMENT

We do not expect any ethical concerns specific to this work and we declare no conflicts of interest.

REPRODUCIBILITY STATEMENT

We believe all required information for the reproduction of the mathematical claims is available in the paper and the appendix. Additionally, all baseline numbers are taken from the cited papers under the same recipes, as stated in Section 5. The source code and the ImageNet-1k pre-trained Tiny and Small checkpoints will be released publicly upon publication.

REFERENCES

Walid Al-Dhabyani, Mohammed Gomaa, Hussien Khaled, and Aly Fahmy. Dataset of breast ultrasound images. *Data in Brief*, 28:104863, 2020.

Haya Alyoussef, Ahmad Bdeir, Diego Coello de Portugal Mecke, Tom Hanika, Niels Landwehr, and Lars Schmidt-Thieme. HexFormer: Hyperbolic vision transformer with exponential map aggregation. *arXiv preprint arXiv:2601.19849*, 2026.

Ahmad Bdeir, Kristian Schwethelm, and Niels Landwehr. Fully hyperbolic convolutional neural networks for computer vision. In *International Conference on Learning Representations*, 2024.

Ahmad Bdeir, Johannes Burchert, Lars Schmidt-Thieme, and Niels Landwehr. Robust hyperbolic learning with curvature-aware optimization. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Lucas Beyer, Xiaohua Zhai, and Alexander Kolesnikov. Better plain ViT baselines for ImageNet-1k. *arXiv preprint arXiv:2205.01580*, 2022.

Ines Chami, Albert Gu, Dat Nguyen, and Christopher Ré. HoroPCA: Hyperbolic dimensionality reduction via horospherical projections. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 1419–1429, 2021.

Weize Chen, Xu Han, Yankai Lin, Hexu Zhao, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. Fully hyperbolic neural networks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pp. 5672–5686, 2022.

Ziheng Chen, Yue Song, Xiao-Jun Wu, and Nicu Sebe. Gyrogroup batch normalization. In *International Conference on Learning Representations*, 2025.

Ziheng Chen, Bernhard Schölkopf, and Nicu Sebe. Hyperbolic Busemann neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Noel Codella, Veronica Rotemberg, Philipp Tschandl, M. Emre Celebi, Stephen Dusza, David Gutman, Brian Helba, Aadi Kalloo, Konstantinos Liopyris, Michael Marchetti, Harald Kittler, and Allan Halpern. Skin lesion analysis toward melanoma detection 2018: A challenge hosted by the international skin imaging collaboration (ISIC). *arXiv preprint arXiv:1902.03368*, 2019.

Ekin D. Cubuk, Barret Zoph, Dandelion Mané, Vijay Vasudevan, and Quoc V. Le. AutoAugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 113–123, 2019.

Chang Dai, Hongyu Shan, Mingyang Song, and Di Liang. HoPE: Hyperbolic rotary positional encoding for stable long-range dependency modeling in large language models. *arXiv preprint arXiv:2509.05218*, 2025.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

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*, 2021.

Deng-Ping Fan, Ge-Peng Ji, Tao Zhou, Geng Chen, Huazhu Fu, Jianbing Shen, and Ling Shao. PraNet: Parallel reverse attention network for polyp segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pp. 263–273, 2020.

Xiran Fan, Chun-Hao Yang, and Baba C. Vemuri. Horospherical decision boundaries for large margin classification in hyperbolic space. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Jacob Fein-Ashley, Ethan Feng, and Minh Pham. HVT: A comprehensive vision framework for learning in non-Euclidean space. *arXiv preprint arXiv:2409.16897*, 2024.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Mina Ghadimi Atigh, Martin Keller-Ressel, and Pascal Mettes. Hyperbolic Busemann learning with ideal prototypes. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

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. In *International Conference on Learning Representations*, 2019.

Yunhui Guo, Xudong Wang, Yubei Chen, and Stella X. Yu. Clipped hyperbolic classifiers are super-hyperbolic classifiers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

David Gutman, Noel C. F. Codella, Emre Celebi, Brian Helba, Michael Marchetti, Nabin Mishra, and Allan Halpern. Skin lesion analysis toward melanoma detection: A challenge at the international symposium on biomedical imaging (ISBI) 2016, hosted by the international skin imaging collaboration (ISIC). *arXiv preprint arXiv:1605.01397*, 2016.

Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassiulas, Menglin Yang, and Rex Ying. HELM: Hyperbolic large language models via mixture-of-curvature experts. In *Advances in Neural Information Processing Systems*, volume 38, 2025a.

Neil He, Hiren Madhu, Ngoc Bui, Menglin Yang, and Rex Ying. Hyperbolic deep learning for foundation models: A survey. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2025b.

Neil He, Menglin Yang, and Rex Ying. HyperCore: The core framework for building hyperbolic foundation models with comprehensive modules. *arXiv preprint arXiv:2504.08912*, 2025c.

Neil He, Menglin Yang, and Rex Ying. Lorentzian residual neural networks. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2025d.

Debesh Jha, Pia H. Smedsrud, Michael A. Riegler, Pål Halvorsen, Thomas de Lange, Dag Johansen, and Håvard D. Johansen. Kvasir-SEG: A segmented polyp dataset. In *International Conference on Multimedia Modeling*, pp. 451–462, 2020.

Isay Katsman, Eric Ming Chen, Sidhanth Holalkere, Anna Asch, Aaron Lou, Ser-Nam Lim, and Christopher De Sa. Riemannian residual neural networks. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

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

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Marc Teva Law, Renjie Liao, Jake Snell, and Richard S. Zemel. Lorentzian distance learning for hyperbolic representations. In *International Conference on Machine Learning*, 2019. URL `https://api.semanticscholar.org/CorpusID:174799760`.

Ya Le and Xuan Yang. Tiny ImageNet visual recognition challenge, 2015. CS231N course project report, Stanford University.

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

Emile Mathieu, Charline Le Lan, Chris J. Maddison, Ryota Tomioka, and Yee Whye Teh. Continuous hierarchical representations with Poincaré variational auto-encoders. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Gal Mishne, Zhengchao Wan, Yusu Wang, and Sheng Yang. The numerical stability of hyperbolic representation learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 24925–24949, 2023.

Swasti S. Mishra, Max van Spengler, Erwin Berkhout, and Pascal Mettes. Hyperbolic U-Net for robust medical image segmentation. In *Proceedings of the 9th International Conference on Medical Imaging with Deep Learning*, volume 315 of *Proceedings of Machine Learning Research*, pp. 1217–1251, 2026.

Maximilian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Maximilian Nickel and Douwe Kiela. Learning continuous hierarchies in the Lorentz model of hyperbolic geometry. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

Seulki Park, Youren Zhang, Stella X. Yu, Sara Beery, and Jonathan Huang. Visually consistent hierarchical image classification. In *International Conference on Learning Representations*, 2025.

Sofía Pérez Casulo, Marcelo Fiori, Bernardo Marenco, and Federico Larroca. A unified framework of hyperbolic graph representation learning methods. *arXiv preprint arXiv:2604.28070*, 2026.

Ofir Press, Noah A. Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*, 2022.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pp. 234–241, 2015.

Frederic Sala, Chris De Sa, Albert Gu, and Christopher Ré. Representation tradeoffs for hyperbolic embeddings. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4460–4469, 2018.

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

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*, 2018.

Xianglong Shi, Ziheng Chen, Yunhan Jiang, and Nicu Sebe. Intrinsic Lorentz neural network. In *International Conference on Learning Representations*, 2026.

Ryohei Shimizu, Yusuke Mukuta, and Tatsuya Harada. Hyperbolic neural networks++. In *International Conference on Learning Representations*, 2021.

Ondrej Skopek, Octavian-Eugen Ganea, and Gary Bécigneul. Mixed-curvature variational autoencoders. In *International Conference on Learning Representations*, 2020.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pp. 10347–10357, 2021.

Robert van der Klis, Ricardo Chávez Torres, Max van Spengler, Yuhui Ding, Thomas Hofmann, and Pascal Mettes. Fast and geometrically grounded Lorentz neural networks. *arXiv preprint arXiv:2601.21529*, 2026.

Max van Spengler, Erwin Berkhout, and Pascal Mettes. Poincaré ResNet. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The Caltech-UCSD birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. Hypformer: Exploring efficient transformer fully in hyperbolic space. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 3770–3781, 2024. doi: 10.1145/3637528.3672039.

Menglin Yang, Ram Samarth B B, Aosong Feng, Bo Xiong, Jihong Liu, Irwin King, and Rex Ying. Hyperbolic fine-tuning for large language models. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Tao Yu and Christopher M. De Sa. Numerically accurate hyperbolic embeddings using tiling-based models. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Tao Yu and Christopher M. De Sa. Representing hyperbolic space accurately using multi-component floats. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Tao Yu, Wentao Guo, Jianan Canal Li, Tiancheng Yuan, and Christopher De Sa. MCTensor: A high-precision deep learning library with multi-component floating-point. In *Hardware Aware Efficient Training Workshop, ICML*, 2022.

A Ambient Failure Sites in a Lorentz Transformer

Table 4 lists the centroid computations in a Lorentz Transformer together with the condition under which ambient arithmetic computes each of them correctly. Each condition follows from the condition $E + S \gtrsim \varepsilon\, e^a$ in Section 3.2. Manifold weight decay is the most extreme case. It pulls a parameter towards the origin by taking the centroid of $\{x, o\}$ with weights $(1 - w, w)$, where $w \approx 10^{-5}$ and $S = 0$. Beyond $r \approx 9$, the step that should move the parameter inward instead moves it outward, which we also measure in Appendix E.4.

When the condition fails, the computed normalizer is rounding noise of size about $\sqrt{\varepsilon}\, e^a$. Dividing by this noise places the output at the rescaled radius $\operatorname{arsinh}(1/2\sqrt{\varepsilon}) \approx r_\varepsilon/\sqrt{k}$ for any input. If the difference instead rounds to a negative value and is clamped to a floor η, the output is placed at the rescaled radius $a + \frac{1}{2}\ln(1/\eta)$. The backward pass differentiates through the same normalizer and breaks down before the forward pass does. The ambient centroid gradient is 10^3 to 10^5 times larger than the true tangent gradient, and for a leaked weight of $\delta = 10^{-5}$ its relative error in single precision already reaches 0.6 at $r = 8$. Gyroaddition, which is used to center the tokens in gyro normalization, and the exponential map for inward steps contain the same kind of difference. We give their polar forms in Appendix B.

Table 4: Centroid computations in a Lorentz Transformer and the condition under which ambient arithmetic computes them correctly. Here, $a = r/\sqrt{k}$ is the rescaled operating radius, θ is the angular spread of the aggregated points and δ is the weight that leaks from the dominant point.

Operation	Weights	Condition for correctness
Attention aggregation	Softmax row with leak δ	$\delta\theta^2 \gtrsim 4\varepsilon$
Manifold weight decay, centroid residual	$(1 - w, w)$	$w \gtrsim \varepsilon e^a$
Batch-norm mean, token pooling	Uniform	$\theta \gtrsim 2\sqrt{\varepsilon}$ or $r \le r_\varepsilon$
Gyro-norm centering $x \oplus (-\bar{x})$	—	$r \le r_\varepsilon$

B Polar Operations

In this section, we write $x = \sqrt{k}(\cosh a,\ \sinh a\, \mathbf{u})$ with $a = r/\sqrt{k}$ and unit direction $\mathbf{u}$, and similarly $y = \sqrt{k}(\cosh b,\ \sinh b\, \mathbf{v})$. The angle θ between $\mathbf{u}$ and $\mathbf{v}$ only appears through the chord $1 - \cos\theta = \frac{1}{2}\|\mathbf{u} - \mathbf{v}\|^2$. We denote tangent vectors by ξ and ζ.

B.1 Tangent Vectors in the Polar Frame

Riemannian optimizers evaluate the Lorentzian inner product on tangent vectors, where the same cancellation occurs. If we write the tangent vectors in terms of their radial and perpendicular components $(\xi_r, \boldsymbol{\xi}_\perp)$, the inner product becomes a plain sum, $\langle \xi, \zeta \rangle_{\mathcal{L}} = \xi_r \zeta_r + \boldsymbol{\xi}_\perp \cdot \boldsymbol{\zeta}_\perp$. The gradient projection, the exponential and logarithmic maps and the parallel transport can all be written in these components, as we show below, so the optimizer never has to leave the polar form.

A tangent vector $\xi = (\xi_0, \boldsymbol{\xi}_s)$ at x satisfies $\langle x, \xi \rangle_{\mathcal{L}} = 0$, which gives $\xi_0 = \tanh(a)\,(\mathbf{u} \cdot \boldsymbol{\xi}_s)$. The tangent vector is therefore determined by its space part. Let $e_r = (\sinh a,\ \cosh a\, \mathbf{u})$ be the outward unit radial vector at x. We decompose ξ as

$$\xi = \xi_r\, e_r + (0, \boldsymbol{\xi}_\perp), \qquad \xi_r = \frac{\mathbf{u} \cdot \boldsymbol{\xi}_s}{\cosh a}, \qquad \boldsymbol{\xi}_\perp = \boldsymbol{\xi}_s - (\mathbf{u} \cdot \boldsymbol{\xi}_s)\, \mathbf{u}. \tag{13}$$

Substituting ξ_0 and ζ_0 into the Lorentzian inner product gives

$$\langle \xi, \zeta \rangle_{\mathcal{L}} = -\tanh^2(a)\,(\mathbf{u} \cdot \boldsymbol{\xi}_s)(\mathbf{u} \cdot \boldsymbol{\zeta}_s) + (\mathbf{u} \cdot \boldsymbol{\xi}_s)(\mathbf{u} \cdot \boldsymbol{\zeta}_s) + \boldsymbol{\xi}_\perp \cdot \boldsymbol{\zeta}_\perp = \xi_r \zeta_r + \boldsymbol{\xi}_\perp \cdot \boldsymbol{\zeta}_\perp, \tag{14}$$

which is a sum of squares when $\xi = \zeta$. For the radial components, the ambient expression $-\xi_0\zeta_0 + \boldsymbol{\xi}_s \cdot \boldsymbol{\zeta}_s$ computes the same quantity as the difference $\cosh^2 a - \sinh^2 a$. In single precision, the computed norm of a radial unit vector is 0.87 at $a = 8$ and 0 at $a = 12$. At the origin, e_r is not defined and every tangent vector is treated as perpendicular.

B.2 Gradient Projection

The Riemannian gradient of a loss with Euclidean gradient $g = (g_0, \mathbf{g}_s)$ is the projection of the Minkowski-raised vector $(-g_0, \mathbf{g}_s)$ onto the tangent space. Since $\xi_r = \langle (-g_0, \mathbf{g}_s), e_r \rangle_{\mathcal{L}}$, its components in the frame of Eq. (13) are

$$\xi_r = g_0 \sinh a + (\mathbf{g}_s \cdot \mathbf{u}) \cosh a, \qquad \boldsymbol{\xi}_\perp = \mathbf{g}_s - (\mathbf{g}_s \cdot \mathbf{u})\, \mathbf{u}. \tag{15}$$

Neither component involves a division or a difference of large terms. Every polar operation only reads the space parts of points and tangent vectors, so a model built from these operations has $g_0 = 0$, and Eq. (15) is exact.

B.3 Exponential Map

Let ξ have the components $(\xi_r, \boldsymbol{\xi}_\perp)$ at x. We write $\omega_r = \xi_r/\sqrt{k}$ for the radial step length and $\omega^2 = \omega_r^2 + \|\boldsymbol{\xi}_\perp\|^2/k$ for the total step length, both in curvature units, and $\chi = \tanh a$ for the Klein radius of x. We insert $x = \sqrt{k}(\cosh a, \sinh a\, \mathbf{u})$ and $\xi = \xi_r e_r + (0, \boldsymbol{\xi}_\perp)$ into $\exp_x(\xi) = \cosh(\omega)\, x + \mathrm{sinhc}(\omega)\, \xi$ and divide the time and space coordinates of the result by $\sqrt{k}\cosh a$, which gives

$$\tilde{x}_0 = \cosh\omega + \mathrm{sinhc}(\omega)\, \omega_r\, \chi, \qquad \tilde{\mathbf{x}} = \big(\cosh\omega\, \chi + \mathrm{sinhc}(\omega)\, \omega_r\big)\, \mathbf{u} + \mathrm{sinhc}(\omega)\, \frac{\boldsymbol{\xi}_\perp}{\sqrt{k}\cosh a}, \tag{16}$$

where $\mathrm{sinhc}(\omega) = \sinh(\omega)/\omega$. The output direction is $\tilde{\mathbf{x}}/\|\tilde{\mathbf{x}}\|$. For the output radius, we use $e^{a'} = (x_0' + \|\mathbf{x}'\|)/\sqrt{k} = \cosh a\, (\tilde{x}_0 + \|\tilde{\mathbf{x}}\|)$ and $\cosh a = e^a/(1+\chi)$, which gives

$$a' = a + \ln\big(\tilde{x}_0 + \|\tilde{\mathbf{x}}\|\big) - \ln(1+\chi). \tag{17}$$

The radius is therefore updated by an increment in log space, and e^a is never formed. For an inward step with $\omega_r < 0$, the terms $\cosh\omega - \sinh\omega\, \chi$ in $\tilde{x}_0$ and $\cosh\omega\, \chi - \sinh\omega$ in the radial coefficient cancel. We define $\psi = 1 - |\omega_r|/\omega = \|\boldsymbol{\xi}_\perp\|^2/\big(k\omega(\omega + |\omega_r|)\big)$ and use $1 - \chi = 2/(e^{2a}+1)$ to rewrite both terms as

$$\tilde{x}_0 = e^{-\omega} + \sinh\omega\, \big[(1-\chi) + \chi\psi\big], \qquad \cosh\omega\, \chi + \mathrm{sinhc}(\omega)\, \omega_r = \chi\, e^{-\omega} + \sinh\omega\, \big[\psi - (1-\chi)\big]. \tag{18}$$

This can be verified by substituting $\omega_r = -\omega(1-\psi)$. In this form, all large terms have the same sign. Since the map is even in ω, its gradient through ω is zero at $\xi = 0$.

B.4 Logarithmic Map

We write $D = d_{\mathcal{L}}(x,y)/\sqrt{k}$ and $c = \cosh(D) - 1$, computed with Eq. (5). The logarithmic map $\xi = \log_x(y) = -\frac{1}{2}\nabla_x\, d^2_{\mathcal{L}}(x,y)$ then has the frame components

$$\xi_r = -\sqrt{k}\, \frac{D}{\sinh D}\, \Big[\sinh(a-b) + \cosh a \sinh b\, (1-\cos\theta)\Big], \qquad \boldsymbol{\xi}_\perp = \sqrt{k}\, \frac{D}{\sinh D}\, \sinh b\, \big(\mathbf{v} - (\mathbf{v}\cdot\mathbf{u})\, \mathbf{u}\big). \tag{19}$$

We obtain the radial component from $\partial c/\partial a = \sinh(a-b) + \cosh a \sinh b(1-\cos\theta)$, and the perpendicular component from the derivative of c with respect to $\mathbf{u}$ on the sphere of radius $\sqrt{k}\sinh a$. For two points with the same direction, this gives $\xi_r = \sqrt{k}\,(b-a)$ and $\boldsymbol{\xi}_\perp = 0$. For two points at equal radius with a small angle between them, $\|\boldsymbol{\xi}_\perp\| \to \sqrt{k}\sinh a\, \theta$.

B.5 Parallel Transport

The parallel transport of $\xi \in T_x$ to T_y along the geodesic is $P_{x\to y}(\xi) = \xi + \langle y, \xi\rangle_{\mathcal{L}}\, (x+y)/(k - \langle x, y\rangle_{\mathcal{L}})$. We evaluate $-\langle x, y\rangle_{\mathcal{L}}$ with Eq. (5), and $\langle y, \xi\rangle_{\mathcal{L}}$ with Eq. (14) after expressing ξ in the frame at y. The retraction used by the Riemannian optimizers is the exponential map above followed by this transport.

B.6 GYROADDITION

The gyroaddition $x \oplus y$ applies to y the Lorentz boost L_x that carries the origin to x. Its space part is

$$\mathbf{z} = \mathbf{y} + \Big[\frac{\mathbf{x}\cdot\mathbf{y}}{\sqrt{k}\,(x_0+\sqrt{k})} + \frac{y_0}{\sqrt{k}}\Big]\,\mathbf{x}, \tag{20}$$

which is smooth at $x = o$ with Jacobian $y_0/\sqrt{k}$ and does not require the direction of x. When $\mathbf{x}\cdot\mathbf{y} < 0$ and $\|\mathbf{x}\| > \sqrt{k}$, the two terms of the coefficient cancel. This is the case for the centering step $x \oplus (-\bar{x})$ of gyro normalization, and at $a = 8$ the centered token has an error of 50% in single precision. In this case, we instead evaluate the component of $\mathbf{z}$ along $\mathbf{u}$ as

$$\sqrt{k}\,\Big[\sinh(a-b) + \cosh a \sinh b\,\tfrac{1}{2}\|\mathbf{u}+\mathbf{v}\|^2\Big], \tag{21}$$

where both terms have the same sign, and take the perpendicular part as the projection of $\mathbf{y}$. The geodesic error of the centered token then stays within a factor of two of the resolution limit, with $5.6\cdot10^{-6}$ at $r = 4$, $4.0\cdot10^{-4}$ at $r = 8$ and $2.0\cdot10^{-2}$ at $r = 12$, compared to $1.2\cdot10^{-4}$, 0.70 and NaN for the ambient form.

B.7 CENTROID

As in Section 3.2, let $T = \sum_i w_i \cosh a_i$, $\mathbf{V} = \sum_i w_i \sinh a_i\,\mathbf{u}_i$, $\hat{\mathbf{u}} = \mathbf{V}/\|\mathbf{V}\|$, $E = \sum_i w_i e^{-a_i}$ and $S = \sum_i w_i \sinh a_i\,\frac{1}{2}\|\mathbf{u}_i - \hat{\mathbf{u}}\|^2$. The centroid of Eq. (4) has direction $\hat{\mathbf{u}}$ and radius

$$\ln\sinh a_\mu = \ln\|\mathbf{V}\| - \tfrac{1}{2}\ln(E+S) - \tfrac{1}{2}\ln\big(T+\|\mathbf{V}\|\big). \tag{22}$$

The identity $T - \|\mathbf{V}\| = E + S$ follows from $\|\mathbf{V}\| = \hat{\mathbf{u}}\cdot\mathbf{V} = \sum_i w_i \sinh a_i\,(1 - \frac{1}{2}\|\mathbf{u}_i - \hat{\mathbf{u}}\|^2)$ and $\cosh a - \sinh a = e^{-a}$. If all weights in a row are zero, the centroid returns the origin with a finite gradient. The manifold weight decay of Riemannian AdamW is the special case of two points x and o with weights $(1-w, w)$, for which $T = (1-w)\cosh a + w$, $\mathbf{V} = (1-w)\sinh a\,\mathbf{u}$, $E = (1-w)e^{-a} + w$ and $S = 0$.

B.8 PRECISION BOUND

The stored inputs are already rounded, so no formula can be more accurate than the stored points allow. The following proposition makes this precise.

Proposition 2 (Polar precision bound)**.** *Let a point be stored in polar form with radius error Δr and direction error $\Delta\theta$. Then the stored point lies within distance $\Delta r + \sqrt{k}\sinh(a)\,\Delta\theta$ of the true point, and every polar operation returns a result within a constant multiple of this bound at every radius below the overflow of* cosh*, $a < 88.7$ in single precision and bfloat16.*

We refer to this bound as the resolution limit. It applies to any method that stores the direction as a floating-point unit vector. With $\Delta\theta \approx \varepsilon$, the limit reaches $\sqrt{k}$ at $r \approx \sqrt{k}\ln(2/\varepsilon) \approx 2r_\varepsilon$. For $k = 1$, the polar operations therefore resolve unit distances up to $r \approx 17.3$ in single precision, compared to 8.3 for the ambient operations, and up to 6.2 in bfloat16, compared to 2.8. Appendix D describes how our implementation reaches this bound and what it costs.

B.9 PROOF OF PROPOSITION 2

Consider a stored point with radius $r + \Delta r$ and a direction $\mathbf{u}'$ at an angle $\Delta\theta$ from $\mathbf{u}$. This point lies on the sphere of radius $r + \Delta r$ around the origin, whose intrinsic geometry is that of a Euclidean sphere of radius $\sqrt{k}\sinh\big((r+\Delta r)/\sqrt{k}\big)$. By the triangle inequality, its geodesic distance from the true point is therefore at most $\Delta r + \sqrt{k}\sinh(a)\,\Delta\theta + O(\Delta r\,\Delta\theta)$.

Each operation of Section 3.2 evaluates a sum of non-negative terms, where each term has a relative rounding error of at most a small multiple of ε. The sum is then passed through arsinh, $\ln$ or a normalization, all of which have a condition number of at most one in the relevant argument. The constant in the bound is the number of roundings in the longest chain of operations, which is at most 12 for the centroid. For the range statement, the largest quantity that is formed is $\cosh a$ or $\sinh a$ of

an input radius, which overflows single precision at $a = 88.7$. The log form of Eq. (22) never forms products of these quantities. Table 7 confirms that the measured error stays within a factor of two of the bound.

C Horospherical Shift

C.1 Generator and Closed Form

Let A be the generator of the parabolic one-parameter subgroup of $\mathrm{SO}(1, n)$ that acts on the coordinates (x_0, x_1, x_2) as $Ae_0 = e_2$, $Ae_1 = -e_2$ and $Ae_2 = e_0 + e_1$ in the Minkowski metric, and is zero on the remaining coordinates. Since $A^3 = 0$, the exponential series terminates after the quadratic term, $\Phi(t) = \exp(tA) = I + tA + \frac{1}{2}t^2A^2$, which gives

$$\Phi(t): \quad \nu = x_0 - x_1, \qquad x_2 \mapsto x_2 + t\nu, \qquad x_0 \mapsto x_0 + tx_2 + \tfrac{1}{2}t^2\nu, \qquad x_1 \mapsto x_1 + tx_2 + \tfrac{1}{2}t^2\nu. \tag{23}$$

This is Eq. (10), and the coordinate ν is invariant under $\Phi(t)$. To see that $\Phi(t)$ preserves $\langle\cdot,\cdot\rangle_{\mathcal{L}}$, let $\Delta = (tx_2 + \frac{1}{2}t^2\nu,\ tx_2 + \frac{1}{2}t^2\nu,\ t\nu)$ be the change in the (x_0, x_1, x_2) coordinates. Then $\langle\Phi(t)x, \Phi(t)x\rangle_{\mathcal{L}} - \langle x, x\rangle_{\mathcal{L}} = 2\langle x, \Delta\rangle_{\mathcal{L}} + \langle\Delta, \Delta\rangle_{\mathcal{L}} = 2\nu\,(-tx_2 - \frac{1}{2}t^2\nu + tx_2) + t^2\nu^2 = 0$. The level sets of ν are the horospheres centered at the ideal point $e_0 + e_1$, and $-\ln(\nu/\sqrt{k})$ is the Busemann function towards this point. The shift $\Phi(t)$ therefore slides each horosphere along itself.

C.2 Relative Property

Since Φ is a one-parameter group of isometries,

$$\langle\Phi(m)\,q, \Phi(m')\,\kappa\rangle_{\mathcal{L}} = \langle q, \Phi(-m)\Phi(m')\,\kappa\rangle_{\mathcal{L}} = \langle q, \Phi(m' - m)\,\kappa\rangle_{\mathcal{L}}.$$

The squared distance in Eq. (9) between a query at position m and a key at position m' therefore only depends on $m' - m$. The same argument holds for the rotation and boost subgroups. On a grid of patches, we apply one shift per grid axis, each on its own pair of reserved space axes. The two shifts do not commute, so the relative property holds exactly along each grid axis and holds to first order in the rates when both axes are combined.

C.3 Distance Law

From the computation above, we get $\langle x, \Phi(t)\,x\rangle_{\mathcal{L}} = \langle x, x\rangle_{\mathcal{L}} + \langle x, \Delta\rangle_{\mathcal{L}} = -k - \frac{1}{2}t^2\nu^2$, which gives

$$\cosh\!\big(d_{\mathcal{L}}(x, \Phi(t)x)/\sqrt{k}\big) = 1 + \frac{t^2\nu^2}{2k}, \qquad d_{\mathcal{L}}(x, \Phi(t)x) = 2\sqrt{k}\,\operatorname{arsinh}\!\Big(\frac{t\,\nu}{2\sqrt{k}}\Big). \tag{24}$$

This is Eq. (11). At the origin, $\nu = \sqrt{k}$. For comparison, a rotation by an angle t moves a point at radius r by $\sqrt{k}\,\operatorname{arcosh}(\cosh^2 a - \sinh^2 a \cos t)$, which is bounded by $2r$, and a boost with rapidity t moves it by at least $\sqrt{k}\,t$, with equality on the boost axis, which is linear in t. The horospherical distance grows as $2\sqrt{k}\ln(t\nu/\sqrt{k})$ for large t. As a result, the attention score of Eq. (9) decreases quadratically in $\ln t$ instead of in t, so nearby positions are separated sharply and the score decays slowly at long range. Points far out towards the ideal point have a small ν, so the same t moves them less and they are less sensitive to position.

D Implementation Details

D.1 Polar Arithmetic

Our implementation relies on two details to reach the resolution limit of Proposition 2. First, we always compute the angular term as the chord $\frac{1}{2}\|\mathbf{u} - \mathbf{v}\|^2$ and never as $1 - \mathbf{u}\cdot\mathbf{v}$. The latter has a relative error of about ε/θ^2 for small angles, which destroys S in the aligned regime where it is needed. For N query and N key directions in $\mathbb{R}^n$, the chord creates an (N, N, n) tensor that automatic differentiation would store for the backward pass. We instead use the closed-form gradient

$$\frac{\partial\mathcal{J}}{\partial\mathbf{u}_i} = (\mathbf{u}_i - \bar{\mathbf{v}})\sum_j g_{ij} - \sum_j g_{ij}\,(\mathbf{v}_j - \bar{\mathbf{v}}), \tag{25}$$

where $\mathcal{J}$ is the loss, g_{ij} is the upstream gradient of the chord between $\mathbf{u}_i$ and $\mathbf{v}_j$, and $\bar{\mathbf{v}}$ is the mean key direction. This gradient only requires one row sum and one matrix product. Centering on $\bar{\mathbf{v}}$ does not change the value, but it is needed for precision. Without it, the two sums are nearly equal when the directions are clustered, and for directions aligned to 10^{-6} the relative error is $2 \cdot 10^{-2}$ instead of 10^{-7}. Second, we compute the centroid radius in the log domain, $\ln \sinh a_\mu = \ln \|\mathbf{V}\| - \frac{1}{2}\ln(E+S) - \frac{1}{2}\ln(T+\|\mathbf{V}\|)$, so that no intermediate quantity overflows before $\cosh a$ itself does.

When compiled, the polar attention of a 9-block ViT takes the same time as the ambient form and uses the same amount of memory. We report the cost of the full model in Appendix E.

D.2 Initialization

We initialize the direction maps with the Xavier scheme in the MLP and the patch embedding, and with orthogonal matrices in the attention layers. The scale of every layer and the radius gain of each attention head are calibrated on the first training batch, so that all models compared in Section 5 start at the same radius. We apply the horospherical shift to the query and key slices of every head after the head split, and we exclude the rates ρ_h from weight decay.

D.3 Training Protocol

Following HexFormer (Alyoussef et al., 2026), ViT-Tiny has 9 layers with a width of 192, an MLP width of 384 and 12 heads, while ViT-Base has 12 layers with a width of 768, an MLP width of 3072 and 12 heads. Both use a patch size of 4 and a curvature of -1. Following Alyoussef et al. (2026), we report the mean and standard deviation over 10 seeds for ViT-Tiny and 5 seeds for ViT-Base.

We train all models with the HexFormer recipe (Alyoussef et al., 2026) for 100 epochs with a batch size of 128. The augmentations are random crops and flips, AutoAugment (Cubuk et al., 2019), random erasing, and CutMix or MixUp with probability 0.5. We use a label-smoothing cross-entropy loss and Riemannian AdamW (Bdeir et al., 2025) with a weight decay of $5 \cdot 10^{-2}$ and a cosine schedule with 10 warmup epochs. The learning rate is $3 \cdot 10^{-3}$ for the Tiny models. Scale parameters are never weight-decayed, and the radius parameters of the polar fully connected layer are decayed towards their initial values.

D.4 ImageNet-1k Training and Fine-Tuning

Datasets. CUB-200-2011 (Wah et al., 2011) has 5,994 training and 5,794 test images of 200 bird species. FGVC-Aircraft (Maji et al., 2013) has 6,667 training images from the official `trainval` split and 3,333 test images of 100 aircraft variants. Neither dataset has a validation split, so we report the test accuracy at the last epoch without checkpoint selection.

Models. At the ViT-Small size, Polar-ViT has 12 layers with a width of 384, an MLP width of 1536, 6 heads and a patch size of 16, which gives 21.7M parameters. It uses sine-cosine absolute positions together with the horospherical positional encoding, the centroid readout and the hyperbolic multinomial logistic regression head of Section 4. The Euclidean ViT-S has the same depth, width, number of heads and patch size. At the ViT-Tiny size, both models have 9 layers with a width of 192 and 2.9M parameters and use learned absolute positions. We train all four models on ImageNet-1k for 90 epochs with the recipe of Beyer et al. (2022), where Polar-ViT uses Riemannian AdamW (Bdeir et al., 2025) in place of AdamW. For the Flat-ViT row, we fine-tune the public DeiT-S (Touvron et al., 2021) checkpoint that Park et al. (2025) start from, which is trained for 300 epochs and reaches 79.9%.

Fine-tuning. We fine-tune every model on the finest labels with the H-CAST recipe as released in the code of Park et al. (2025). This uses 100 epochs with 5 warm-up epochs and a cosine schedule, a learning rate of $5 \cdot 10^{-4} \cdot B/512$ with an effective batch size of $B = 256$, a weight decay of 0.05, label smoothing of 0.1, Mixup with 0.8, CutMix with 1.0, RandAugment with $(9, 0.5)$, random erasing with 0.25, stochastic depth of 0.1, no gradient clipping and an input size of 224×224. The Euclidean models use AdamW and Polar-ViT uses Riemannian AdamW (Bdeir et al., 2025). We load every pretrained weight except the ImageNet classifier, which we replace with a newly

Table 5: Component ablation on CIFAR-100, ViT-Tiny, test accuracy over ten seeds. Each row adds one change to the row above.

Change	Component	Accuracy
Split-head Lorentz ViT, class-token readout	—	73.99 ± 1.03
+ polar change of current ops and gyrolayer norm	-	76.27 ± 0.81
+ boost residual	Section 4.4	77.06 ± 0.43
+ polar boost residual	Eq. (12)	77.20 ± 0.13
+ polar fully connected layer	Eq. (8)	78.04 ± 0.53
+ horospherical positional encoding (Polar-ViT)	Eq. (10)	$\mathbf{80.15 \pm 0.26}$
Polar-ViT with Lorentz boost encoding in place of horoshift	after Dai et al. (2025)	79.82 ± 0.13

initialized head. Polar-ViT at the ViT-Small size reaches the effective batch size as 32×8 with gradient accumulation, and the other models as 128×2. Accumulation gives the same gradient except for Mixup, which draws one coefficient per micro-batch.

E Additional Experiments

E.1 Component Ablation

Table 5 isolates the contribution of each component on CIFAR-100 with ViT-Tiny over ten seeds. Changing to the polar form of the model and using the scaled gyro layer norm gains 2.3 points and also reduces the spread across seeds. Replacing the space-coordinate residual with the boost residual adds a further 0.8 points. Computing the boost residual in polar form keeps the accuracy about the same, but it removes the drift of the embeddings away from the origin and reduces the spread to a third. The polar fully connected layer then gains 0.8 points over the Lorentz fully connected layer of Chen et al. (2022). The horospherical encoding gains another 2.1 points and reduces the standard deviation from 0.53 to 0.26. Finally, we check whether the choice of subgroup matters for the positional encoding. We replace the horospherical shift with a Lorentz boost on the same axes, which applies the boost of HoPE (Dai et al., 2025) to Lorentz embeddings. This reaches 79.82 ± 0.13, which is 0.3 points below the horospherical shift.

E.2 Radius Rule of the Polar Fully Connected Layer

The polar fully connected layer is anchored at initialization, meaning that Eq. (8) starts out as the Lorentz fully connected layer. Table 6 compares this anchored layer on one seed against alternative rules for the output radius that start from a random point instead. These are an affine rule in the input radius, a rule with a free power of the pre-activation norm, and the sigmoid-bounded rule of Chen et al. (2022). All of the rules trained from a random start are 3.2 to 4.4 points below the anchored layer, and a rule with a hard radius bound diverges. Moving the initial parameters away from the anchor by a random deviation reduces the accuracy by 0.4 points at $\sigma = 0.1$ and by 1.1 points at $\sigma = 0.2$.

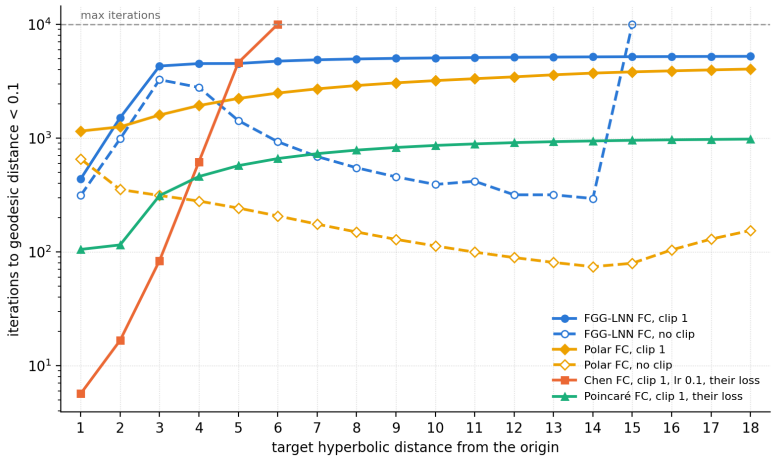
Table 6: Rules for the output radius of the polar fully connected layer on CIFAR-100 with ViT-Tiny, one seed. σ is the standard deviation of a random initial deviation from the anchor.

Radius rule	Accuracy
Anchored gain, Eq. (8)	$\mathbf{78.45}$
Anchored gain, $\sigma = 0.1$	78.02
Anchored gain, $\sigma = 0.2$	77.36
Free power of the pre-activation norm	75.22
Affine in the input radius, Klein input	75.06
Affine in the input radius, unit-direction input	74.42
Sigmoid-bounded radius (Chen et al., 2022)	74.04
Hard radius bound	diverged

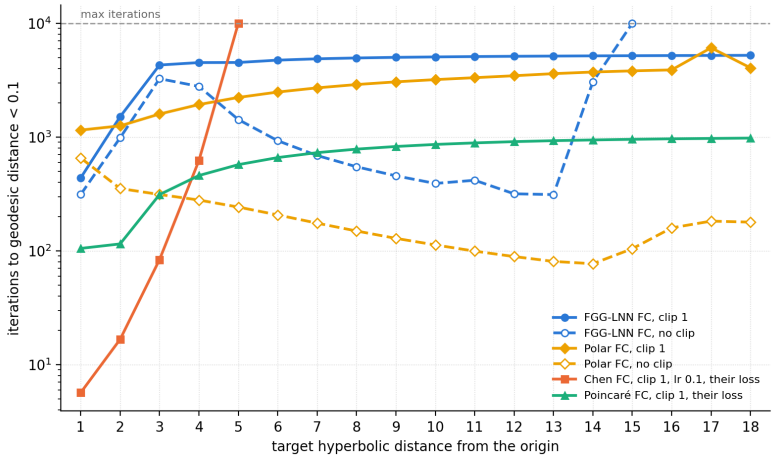
E.3 Expressivity Test

We follow the protocol of van der Klis et al. (2026, Fig. 2), in which a single fully connected layer maps a fixed input to a target point at a hyperbolic distance of 1 to 18 from the origin. The layer is trained with SGD at a learning rate of 10^{-3} for at most 10^4 iterations. We report the number of iterations until the geodesic distance to the target falls below 0.1, averaged over 3 seeds. For all Lorentz layers except that of Chen et al. (2022), the loss is the polar distance of Eq. (5). We run the layer of Chen et al. (2022) with its original loss and a learning rate of 0.1, and the Poincaré layer also uses its original loss.

Figure 1 shows the results. The layer of Chen et al. (2022) reaches the iteration limit at radius 6 in Float64 and at radius 5 in Float32. The Polar-FC reaches every target in both precisions. Without clipping, it needs 70 to 650 iterations, which is fewer than any other layer beyond radius 3. With a clipping threshold of 1, it needs 1,100 to 6,000 iterations. FGG-LNN converges at every radius with clipping, but without clipping it fails from radius 15 in Float64 and from radius 14 in Float32.



(a) Float64



(b) Float32

Figure 1: Iterations needed by a single fully connected layer to reach a target point, as a function of the distance of the target from the origin. We report the mean over 3 seeds. Runs that do not converge are shown at the limit of 10^4 iterations.

E.4 Numerical Precision

Table 7 reports the single-precision error of the ambient and polar operations against double precision, measured on clusters of 64 points with a geodesic spread of 1. As predicted by Proposition 1, the ambient squared distance and centroid deviate from the true values at $r = 8$, and beyond this radius the ambient centroid returns a point at radius 9 for every input. The polar operations stay within a factor of two of the resolution limit of Proposition 2 at every radius.

We also test the low-weight regime of Section 3.1. For one-hot attention rows at $r \geq 8$, the ambient centroid places the output at radius 9 or at radius $r + 9.2$ regardless of the values, and 88 to 100% of these rows are off by more than 0.5 in geodesic distance. The manifold weight decay step of Riemannian AdamW computed with the ambient centroid is off by 0.1 at $r = 8$ and increases the parameter radius from $r = 12$ onward. The polar step matches double precision at every radius.

Table 7: Single-precision error against double precision on clusters of 64 points with geodesic spread 1 at radius r ($k = 1$). We report the relative error for the squared distance and the geodesic error for the weighted centroid and gyro-norm centering. The last column is the bound of Proposition 2 for $\Delta\theta = 2^{-24}$.

	Squared distance		Centroid		Gyro-norm centering		
r	Ambient	Polar	Ambient	Polar	Ambient	Polar	Resolution limit
4	$3 \cdot 10^{-4}$	$4 \cdot 10^{-7}$	$2 \cdot 10^{-4}$	$9 \cdot 10^{-7}$	$1.2 \cdot 10^{-4}$	$5.6 \cdot 10^{-6}$	$5 \cdot 10^{-6}$
8	$7 \cdot 10^{-1}$	$1 \cdot 10^{-5}$	9	$5 \cdot 10^{-5}$	$7.0 \cdot 10^{-1}$	$4.0 \cdot 10^{-4}$	$2 \cdot 10^{-4}$
12	$2 \cdot 10^{3}$	$7 \cdot 10^{-4}$	9	$3 \cdot 10^{-3}$	NaN	$2.0 \cdot 10^{-2}$	$1 \cdot 10^{-2}$

E.5 Computational Cost

Table 8 reports the time of one forward and backward pass of ViT-Tiny at a batch size of 128 on an RTX A5000 under `torch.compile`. In single precision, the polar model takes 1.1 times as long as the ambient Lorentz model and 2.8 times as long as the Euclidean ViT. At this input size, the fused attention kernel reduces the time of the attention core from 4.5 to 3.0 ms. A polar model whose layers pass $(r, \mathbf{u})$ directly agrees with the ambient model to 10^{-14} in double precision.

Table 8: Forward and backward time per training step of ViT-Tiny on CIFAR-100 with batch size 128 on an RTX A5000, compiled. The times include only the model.

Model	fp32
Euclidean ViT	21 ms
Lorentz ViT, ambient arithmetic	52 ms
Polar-ViT, polar arithmetic	58 ms

F Polar Layers in a Convolutional Network

The Polar-FC of Section 4.1 does not depend on attention. A Lorentz convolution concatenates the points of each receptive field and applies a fully connected layer to the result (Bdeir et al., 2024), so replacing this layer, the batch normalization and the activation with their polar forms turns a Lorentz CNN into a polar one without changing its structure. We test this on the medical segmentation benchmark of the Hyperbolic U-Net (Mishra et al., 2026) and compare against the published Euclidean and Poincaré U-Nets and a Lorentz U-Net built from the ambient layers of Bdeir et al. (2024).

Datasets. We use the five datasets and splits of Mishra et al. (2026). These are skin lesions from ISIC 2016 (Gutman et al., 2016) (810 training, 90 validation and 379 test images) and ISIC 2018 (Codella et al., 2019) (2,335 / 259 / 1,000), polyps from Kvasir-SEG (Jha et al., 2020) (810 / 90 / 100) and from the PraNet collection (Fan et al., 2020) (SANET, 1,305 / 145 / 798, where the test set is the union of CVC-300, CVC-ClinicDB, CVC-ColonDB, ETIS-LaribPolypDB

Table 9: Medical image segmentation on the benchmark of Mishra et al. (2026). Test Dice (mean over foreground and background). The published rows are taken from Mishra et al. (2026), and all values are rounded to two decimals as there. The best value of each column is in bold, and ties are both in bold.

	Model	Params	ISIC 2016	ISIC 2018	Kvasir-SEG	SANET	BUSI
published	Euclidean U-Net	486.6 K	0.92	**0.89**	**0.86**	0.78	**0.82**
	Poincaré U-Net	$\approx$ 487 K	0.91	0.87	0.83	0.76	0.80
ours	Lorentz U-Net, ambient	216.0 K	0.92	0.81	0.79	0.73	0.71
	Lorentz U-Net, polar	215.5 K	**0.93**	**0.89**	**0.86**	**0.81**	0.78

and Kvasir), and breast ultrasound images from BUSI (Al-Dhabyani et al., 2020) (624 / 78 / 78). Images are zero-padded to a square, scaled to $[0, 1]$ and resized to 256×256, without augmentation. We reuse the reference data loaders, and our training, validation and test image lists are identical to theirs on all five datasets.

Models. All models share the U-Net topology of Mishra et al. (2026), which has four levels with 8 channels at the first level and twice as many at each following level, two 3×3 convolutions per level, max pooling between the encoder levels, and a 1×1 convolution as the output layer.

- *Euclidean U-Net* (Ronneberger et al., 2015): the network of the reference implementation, with transposed-convolution upsampling (486.6 K parameters).
- *Poincaré U-Net* (Mishra et al., 2026): Poincaré-ball convolutions with a trainable curvature initialized at -0.1.
- *Lorentz U-Net, ambient*: the convolution, batch normalization and ReLU of Bdeir et al. (2024) on points stored in ambient coordinates $(x_0, \mathbf{x}_s)$, with their soft radius cap on every convolution input and batch normalization output. Upsampling is bilinear, where each new point is the Lorentz centroid of its neighbors. Skip connections are joined by Lorentz direct concatenation as in Section 4.2, and the logits are the space part of the logarithmic map of the output layer at the origin (216.0 K parameters).
- *Lorentz U-Net, polar*: the same network with each layer replaced by its polar form. Each convolution concatenates its receptive field in polar form and applies the Polar-FC of Eq. (8). The batch normalization is that of Bdeir et al. (2024) computed in polar form, and the ReLU acts on the space part. Pooling, upsampling and concatenation are the same as in the ambient model, and the output layer is a polar 1×1 convolution. The polar model has no radius cap (215.5 K parameters).

For the Euclidean and Poincaré U-Nets, we take the published numbers of Mishra et al. (2026).

Training. Both Lorentz models use the same recipe. The loss is the sum of the Dice and cross-entropy losses on a two-class softmax, and we train for 50 epochs with a batch size of 8. We use AdamW, with Riemannian AdamW (Bdeir et al., 2025) for the manifold parameters, a learning rate of 10^{-3} with cosine decay to 10^{-6}, and a weight decay of $5 \cdot 10^{-2}$ that is not applied to scale parameters. On BUSI, which is the smallest dataset, we use a learning rate of $3 \cdot 10^{-4}$ with 5 warmup epochs for both models. Following the reference code, the validation Dice is computed five times per epoch, and the test score is that of the checkpoint with the best validation Dice.

Metric. We report the Dice score as the reference code computes it, which is the mean of the foreground and background Dice over the two softmax classes. The published numbers in Table 9 use this definition.

Results. Table 9 shows that the polar U-Net is above the ambient Lorentz U-Net on all five datasets by 0.01 to 0.08 Dice, with the same seeds and 0.5 K fewer parameters. It is above the published Poincaré U-Net on four of the five datasets and 0.02 below it on BUSI. Compared to the published Euclidean U-Net, it is higher on ISIC 2016 and SANET, equal on ISIC 2018 and Kvasir-SEG, and 0.04 lower on BUSI, while using 0.44 times its parameters.

The polar model is also the more stable of the two Lorentz models. In some cases, training of the ambient model had to be repeated with gradient clipping and a lower learning rate.