INTRINSIC LORENTZ NEURAL NETWORK

Anonymous authors
Paper under double-blind review

ABSTRACT

Real-world data frequently exhibit latent hierarchical structures, which can be naturally represented by hyperbolic geometry. Although recent hyperbolic neural networks have demonstrated promising results, many existing architectures remain partially intrinsic, mixing Euclidean operations with hyperbolic ones or relying on extrinsic parameterizations. To address it, we propose the *Intrinsic Lorentz Neural Network* (ILNN), a fully intrinsic hyperbolic architecture that conducts all computations within the Lorentz model. At its core, the network introduces a novel *point-to-hyperplane* fully connected layer (FC), replacing traditional Euclidean affine logits with closed-form hyperbolic distances from features to learned Lorentz hyperplanes, thereby ensuring that the resulting geometric decision functions respect the inherent curvature. Around this fundamental layer, we design intrinsic modules: GyroLBN, a Lorentz batch normalization that couples gyro-centering with gyro-scaling, consistently outperforming both LBN and GyroBN while reducing training time. We additionally proposed a gyro-additive bias for the FC output, a Lorentz patch-concatenation operator that aligns the expected log-radius across feature blocks via a digamma-based scale, and a Lorentz dropout layer. Extensive experiments conducted on CIFAR-10/100 and two genomic benchmarks (TEB and GUE) illustrate that ILNN achieves state-of-the-art performance and computational cost among hyperbolic models and consistently surpasses strong Euclidean baselines.

1 INTRODUCTION

Hierarchical structures exist across a wide range of machine learning applications, including computer vision (Khrulkov et al., 2020; Ghadimi Atigh et al., 2022; Bdeir et al., 2024; Pal et al., 2025), natural language processing (Ganea et al., 2018; Tifrea et al., 2019; Yang et al., 2024a), knowledge-graph reasoning (Nickel & Kiela, 2017; Balazevic et al., 2019; Welz et al., 2025), graph learning (Chami et al., 2019; Yang et al., 2022; Li et al., 2024), and genomics tasks (Khan et al., 2025; Zhou & Sharpee, 2021; Yang et al., 2025). While Euclidean neural networks often incur high distortion or require excessive dimensionality when embedding hierarchical or scale-free structures (Nickel & Kiela, 2018), hyperbolic geometry, characterized by constant negative curvature, offers exponential representational capacity and enables more compact embeddings (Ganea et al., 2018; Shimizu et al., 2020). Consequently, *hyperbolic neural networks* (HNNs) have demonstrated notable success in applications with hierarchical structure (Ganea et al., 2018; Chami et al., 2019; Shimizu et al., 2020; Gulcehre et al., 2019; Li et al., 2024; Fan et al., 2024; Leng et al., 2024; Bdeir et al., 2024; Pal et al., 2025; Chen et al., 2025a; Nguyen et al., 2025; He et al., 2025).

Early hyperbolic neural networks predominantly operated in the Poincaré model because its gyrovector formalism makes many neural primitives straightforward to define (Ganea et al., 2018; Shimizu et al., 2020). However, the unit-ball constraint and boundary saturation render the Poincaré ball more susceptible to numerical instabilities than the Lorentz model, motivating a recent shift toward Lorentz neural networks (LNNs) that exhibit improved optimization stability (Bdeir et al., 2024). Despite this trend leading to the emergence of excellent work (He et al., 2024; Fan et al., 2024; Liang et al., 2024), recent advanced LNNs (Bdeir et al., 2024; Khan et al., 2025) architectures remain only partially intrinsic: they mix Euclidean affine transformations with manifold operations or rely on extrinsic parameterizations that compromise geometric consistency. For example, the Lorentz fully connected layer (LFC) (Bdeir et al., 2024) that applies Euclidean mappings to Lorentz vector, then extract the space part from mapping output and accordingly calculate the time part to form the

output Lorentz vector. Normalization further illustrates this tension. Lorentz batch normalization (LBN) (Bdeir et al., 2024) recenter features but either ignore gyro-variance, while GyroBN (Chen et al., 2025a) offers gyrogroup-based control yet can still be computationally heavy due to depending on Fréchet-type statistics. These compromises limit the representational power and efficiency of Lorentz neural networks.

An effective way to mitigate partially intrinsic designs is the *point-to-hyperplane* formulation, which has been verified on hyperbolic Poincaré (Shimizu et al., 2020) and matrix manifolds (Nguyen et al., 2025; 2024). Inspired by this, we introduce ILNN, a fully intrinsic hyperbolic network in which every operation, parameter, and update is defined inside the Lorentz model. At its core is a *Point-to-hyperplane Lorentz Fully Connected* (PLFC) layer that replaces Euclidean affine transformation with closed-form Lorentzian distances to learned hyperplanes, yielding curvature-aware, margin-interpretable decision functions. Surrounding PLFC, we develop intrinsic components: *GyroLBN*, a batch normalization that couples gyro-centering with variance-controlled gyro-scaling and outperforms LBN and GyroBN while reducing wall-clock time. Furthermore, we introduced a log-radius algin Lorentz patch-concatenation to build the stable CNN module, a gyro-additive bias, and a Lorentz dropout layer to further improve its performance. By designing intrinsic geometric operations, ILNN maintains mathematical consistency and geometric interpretability throughout the network.

The main contributions of our work can be summarized as follows:

1. We propose an *intrinsic hyperbolic neural network* that eliminates reliance on extrinsic Euclidean operations, thus fully harnessing hyperbolic geometry.
2. We introduce a novel *point-to-hyperplane Lorentz fully connected layer*(PLFC), which replaces traditional affine transformations with intrinsic hyperbolic distances, significantly enhancing representational fidelity.
3. We introduce a GyroLBN, a Lorentz batch normalization that combines gyro-centering with gyro-scaling, consistently outperforming both LBN and GyroBN while reducing training time.
4. Extensive experiments demonstrate the effectiveness of ILNN, achieving state-of-the-art performance on CIFAR-10/100 datasets and genomic benchmarks (TEB and GUE), consistently outperforming Euclidean and existing hyperbolic counterparts.

2 RELATED WORK

Hyperbolic embeddings. A large body of work demonstrates that negatively curved representations can efficiently encode hierarchical and scale-free structure across modalities. Foundational results on hyperbolic embeddings (eg, Poincaré embeddings) show strong benefits for symbolic data with latent trees (Nickel & Kiela, 2017), and early hyperbolic neural architectures extend core layers and classifiers to the Poincaré model for language tasks (Ganea et al., 2018). On graphs, hyperbolic GNNs capture hierarchical neighborhoods and outperform Euclidean counterparts on link prediction and node classification (Chami et al., 2019); knowledge graphs likewise benefit from multi-relational hyperbolic embeddings (Balazevic et al., 2019). In vision, hyperbolic image embeddings improve retrieval and classification under class hierarchies (Khrulkov et al., 2020), and fully hyperbolic CNNs (HCNN) generalize convolution, normalization, and MLR directly in the Lorentz model, yielding strong encoder-side gains (Bdeir et al., 2024). Hyperbolic modeling has also been adapted to genomics, where hyperbolic genome embeddings leverage evolutionary signal to surpass Euclidean baselines across diverse benchmarks (Khan et al., 2025). In this paper, we target these two application fronts, *vision* and *genomics*.

Hyperbolic neural networks. Designing *fully hyperbolic* neural networks means replacing every Euclidean building block with operations that are intrinsic to a negatively curved manifold, without shuttling features back and forth between Euclidean and hyperbolic spaces. Early work on the Poincaré ball showed how to lift common layers and primitives (linear/FC maps, activations, concatenation, MLR) using gyrovector-space calculus and Möbius operations (Ganea et al., 2018; Shimizu et al., 2020), while concurrent efforts formulated end-to-end hyperbolic models beyond simple projection heads (Chen et al., 2022). Due to the Poincaré ball's greater susceptibility to numerical instability, subsequent work tried to generalize network components to the Lorentz model (Gulcehre et al., 2019; Chami et al., 2019; Fan et al., 2023). Beyond classification heads, attention mecha-

nisms (Gulcehre et al., 2019), graph convolutional layers (Chami et al., 2019), and fully hyperbolic generative models (Qu & Zou, 2022) were also developed in this setting. A major advance came with HCNN, which introduced Lorentz-native formulations of convolution, batch normalization, and multinomial logistic regression, bringing vision encoders fully into the hyperbolic domain (Chen et al., 2022; Bdeir et al., 2024; Yang et al., 2024b). But it still remains partially intrinsic, mixing Euclidean operations with hyperbolic ones. Our work follows this *intrinsic-first* principle and departs in key aspects: a point-to-hyperplane fully connected layer in the Lorentz model with closed-form h-distances for logits and GyroLBN, a Lorentz batch normalization that couples gyro-centering with gyro-scaling.

Normalization in HNNs. Normalization layers are central to stable and efficient training, yet extending them to curved spaces remains challenging. A general Riemannian batch normalization (RBN) based on the Fréchet mean was first made practical by differentiable solvers, enabling manifold-aware centering and variance control; however, its iterative nature can be slow in practice (Lou et al., 2020). Recently, Bdeir et al. (2024) introduced *Lorentz Batch Normalization (LBN)*, which uses the closed-form Lorentzian centroid (Law et al., 2019) for re-centering and a principled tangent-space rescaling. Concurrently, *Gyrogroup Batch Normalization (GyroBN)* generalizes RBN to manifolds with gyro-structure, performing gyro-centering and variance-controlled gyro-scaling that better preserve distributional shape under gyro-operations (Chen et al., 2025a). While LBN provides efficient Lorentzian re-centering and global rescaling, it does not explicitly align batch statistics under gyro-operations; GyroBN performs such alignment but is computationally heavier because it relies on Fréchet means. Motivated by these trade-offs, we propose *GyroLBN*, which aligns batch statistics under gyro-operations within the Lorentz model while retaining the speed and practicality of closed-form re-centering.

3 Background

There are five isometric hyperbolic models that one can work with (Cannon et al., 1997). We adopt the Lorentz (hyperboloid) model due to numerical stability (Mishne et al., 2023).

Lorentz model. The Lorentz model embeds hyperbolic space as the upper sheet of a two-sheeted hyperboloid in $(n+1)$-dimensional Minkowski space. It is defined as

$$\mathbb{L}_K^n = \left\{ x \in \mathbb{R}^{n+1} \;\middle|\; \langle x, x \rangle_L = \tfrac{1}{K},\ x_0 > 0 \right\}.$$

where $K < 0$ denotes the constant sectional curvature, and $\langle x, y \rangle_L := -x_0 y_0 + \sum_{i=1}^n x_i y_i$ is the Lorentz inner product. Following Ratcliffe (2006), we write $x = (x_0, x_s)$ with *time* coordinate x_0 and *space* coordinates $x_s \in \mathbb{R}^n$. The origin is $o = (K^{-1/2}, 0, \ldots, 0)$. The closed-form squared distance is given by $d_{\mathcal{L}}^2(\boldsymbol{x}, \boldsymbol{y}) = ||\boldsymbol{x} - \boldsymbol{y}||_{\mathcal{L}}^2 = \frac{2}{K} - 2\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathcal{L}}$. Other Riemannian operators are presented in Appendix B, such as exponential and logarithmic maps, and parallel transport.

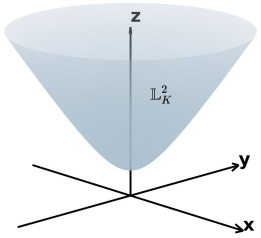

Figure 1: 2-dimensional Lorentz model in the 3-dimensional Minkowski space.

Gyrovector space. It forms the algebraic foundation for the hyperbolic space, as the vector space for the Euclidean space (Ungar, 2022). A gyrovector space has gyroaddition and scalar gyromultiplication, corresponding to the vector addition and scalar multiplication in Euclidean space. Recently, Chen et al. (2025b) proposed the gyroaddition and gyromultiplication over the Lorentz model:

$$x \oplus y = \operatorname{Exp}_x \left(\operatorname{PT}_{\overline{\mathbf{0}} \to x} \left(\operatorname{Log}_{\overline{\mathbf{0}}}(y) \right) \right), \quad \forall x, y \in \mathbb{L}_K^n, \tag{1}$$

$$t \odot x = \operatorname{Exp}_{\overline{\mathbf{0}}} \left(t \operatorname{Log}_{\overline{\mathbf{0}}}(x) \right), \quad \forall t \in \mathbb{R}, \forall x \in \mathbb{L}_K^n. \tag{2}$$

Particularly, the inverse is $\ominus x := (-1) \odot x = [x_t, -x_s]$, which satisfies $x \oplus (\ominus x) = (\ominus x) \oplus x = \overline{\mathbf{0}}$. Eqs. 1 and 2 admit closed-form expressions, which are more efficient.

4 INTRINSIC LORENTZ NEURAL NETWORK

We introduce the Intrinsic Lorentz Neural Network (ILNN), whose components are entirely defined by the Lorentz geometry. Specifically, we propose (i) a point-to-hyperplane fully connected (FC) layer, (ii) GyroLBN, a Lorentz batch normalization layer, and (iii) several other intrinsic modules, including log-radius concatenation, Lorentz activations, and Lorentz dropout.

4.1 POINT-TO-HYPERPLANE LORENTZ FULLY-CONNECTED LAYER

FC layers perform an affine transformation defined by $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}$, which can be element-wisely expressed as $y_k = \boldsymbol{a}_k\boldsymbol{x} - b_k$, where $\boldsymbol{x}, \boldsymbol{a}_k \in \mathbb{R}^n$ and $b_k \in \mathbb{R}$ (Shimizu et al., 2020). Geometrically, this transformation can be interpreted as mapping the input vector $\boldsymbol{x}$ to an output score y_k, representing either the coordinate value or the signed distance relative to a hyperplane passing through the origin and orthogonal to the k-th coordinate axis in the output space $\mathbb{R}^m$. Motivated by this geometric interpretation, in this section, we first derive the Lorentz multinomial logistic regression (Lorentz MLR) to obtain the signed distance, and subsequently present the formulation of the point-to-hyperplane Lorentz fully connected (PLFC) layer.

Lorentz MLR. In Euclidean space, the *multinomial logistic regression* (MLR) for class $c \in \{1, \dots, C\}$ can be formulated as the logits of the Euclidean MLR classifier using the distance from instances to hyperplanes describing the class region (Lebanon & Lafferty, 2004), which can be written as

$$p(y = c \mid \boldsymbol{x}) \propto \exp(v_c(\boldsymbol{x})), \quad v_c(\boldsymbol{x}) = \text{sign}(\langle \boldsymbol{a}_c, \boldsymbol{x} - \boldsymbol{p}_c \rangle) ||\boldsymbol{a}_c|| d(\boldsymbol{x}, H_{\boldsymbol{a}_c, \boldsymbol{p}_c}), \quad \boldsymbol{a}_c \in \mathbb{R}^n, \tag{3}$$

where $H_{\boldsymbol{a}_c, \boldsymbol{p}_c} = \{\boldsymbol{x} \in \mathbb{R} : \langle \boldsymbol{a}_c, \boldsymbol{x} - \boldsymbol{p}_c \}$ is the Euclidean hyperplane of class c and $d(\cdot, \cdot)$ denotes Euclidean distance. Once we have the Lorentz hyperplane and closed-form point-hyperplane distance, we can extend Euclidean MLR to the Lorentz model.

In Lorentz model $\mathbb{L}_K^n$, following Bdeir et al. (2024), for $\boldsymbol{p} \in \mathbb{L}_K^n$ and $\boldsymbol{w} \in \mathcal{T}_{\boldsymbol{p}}\mathbb{L}_K^n$, the hyperplane passing through $\boldsymbol{p}$ and perpendicular to $\boldsymbol{w}$ is given by $H_{\boldsymbol{w}, \boldsymbol{p}} = \{\boldsymbol{x} \in \mathbb{L}_K^n \mid \langle \boldsymbol{w}, \boldsymbol{x} \rangle_{\mathcal{L}} = 0\}$, where $\boldsymbol{w}$ should satisfy the condition $\langle \boldsymbol{w}, \boldsymbol{w} \rangle > 0$. To eliminate this condition, $\boldsymbol{w} \in \mathcal{T}_{\boldsymbol{p}}\mathbb{L}_K^n$ is parameterized by a vector $\overline{\boldsymbol{z}} \in \mathcal{T}_{\overline{\boldsymbol{0}}}\mathbb{L}_K^n = [0, a\boldsymbol{z}/||\boldsymbol{z}||]$, where $a \in \mathbb{R}$ and $\boldsymbol{z} \in \mathbb{R}^n$. As $\boldsymbol{w} \in \mathcal{T}_{\boldsymbol{p}}\mathbb{L}_K^n$, $\overline{\boldsymbol{z}}$ is parallel transport to $\boldsymbol{p}$, the Lorentz hyperplane defined by

$$\tilde{H}_{\boldsymbol{z}, a} = \{\boldsymbol{x} \in \mathbb{L}_K^n \mid \cosh\Big(\sqrt{-K}a\Big)\langle \boldsymbol{z}, \boldsymbol{x}_s \rangle - \sinh\Big(\sqrt{-K}a\Big) ||\boldsymbol{z}|| \, x_t = 0\}, \tag{4}$$

where a and z represent the distance and orientation to the origin, respectively. Due to the $\boldsymbol{z} \in \mathcal{T}_{\overline{\boldsymbol{0}}}\mathbb{L}_K^n$ and the sign-preserving property of parallel transported, this construction naturally satisfies $\langle \boldsymbol{w}, \boldsymbol{w} \rangle > 0$. Then for $x \in \mathbb{L}_K^n$, the distance to a Lorentz hyperplane is given by

$$d_{\mathcal{L}}(\boldsymbol{x}, \tilde{H}_{\boldsymbol{z}, a}) = \frac{1}{\sqrt{-K}} \left| \sinh^{-1}\Big(\sqrt{-K} \frac{\cosh\big(\sqrt{-K}a\big)\langle \boldsymbol{z}, \boldsymbol{x}_s \rangle - \sinh\big(\sqrt{-K}a\big)\|\boldsymbol{z}\|_2 \, x_t}{\sqrt{\|\cosh\big(\sqrt{-K}a\big)\boldsymbol{z}\|_2^2 - \big(\sinh\big(\sqrt{-K}a\big)\|\boldsymbol{z}\|_2\big)^2}}\Big) \right|, \tag{5}$$

Substituting Eq. 4 and Eq. 5 into Eq. 3, we obtain that the Lorentz MLR's output logit corresponding to class c is given by

$$v_{\boldsymbol{z}_c, a_c}(\boldsymbol{x}) = \frac{1}{\sqrt{-K}} \text{sign}(\alpha_c)\beta_c \left| \sinh^{-1}\left(\sqrt{-K}\frac{\alpha_c}{\beta_c}\right) \right|, \tag{6}$$

$$\alpha_c = \cosh\Big(\sqrt{-K}a\Big)\langle \boldsymbol{z}, \boldsymbol{x}_s \rangle - \sinh\Big(\sqrt{-K}a\Big),$$

$$\beta_c = \sqrt{||\cosh\Big(\sqrt{-K}a\Big)\boldsymbol{z}||^2 - (\sinh\Big(\sqrt{-K}a\Big)||\boldsymbol{z}||)^2}.$$

Lorentz fully-connected layer (PLFC). Shimizu et al. (2020) interpreted the Euclidean FC layer as an operation that transforms the input $\boldsymbol{x}$ via $v_k(\boldsymbol{x})$, treating the output $\boldsymbol{y_k}$ as the signed distance from the hyperplane $H_{\boldsymbol{e}^{(k)}, 0}$ passing through the origin and orthogonal to the k-th axis of the output space $\mathbb{R}^m$, which can be written as

$$d^{\pm}(\boldsymbol{y}_k, H_{\boldsymbol{e}^{(k)}, 0}) = v_k(\boldsymbol{x}), \qquad k = 1, \dots, m. \tag{7}$$

where $d^{\pm}(\cdot,\cdot)$ denote the signed distance in Euclidean space. We now collect the above ingredients into an *intrinsic* FC layer that maps an input $\boldsymbol{x}\in\mathbb{L}_K^n$ to an output $\boldsymbol{y}=(y_0,\boldsymbol{y}_s)\in\mathbb{L}_K^m$. Let $\{(\boldsymbol{z}_k,a_k)\}_{k=1}^m$ be learnable hyperplane parameters and define $v_k(\boldsymbol{x})=v_{\boldsymbol{z}_k,a_k}(\boldsymbol{x})$ as in Eq. 6. Matching each $v_k(\boldsymbol{x})$ to the signed h-distance from $\boldsymbol{y}$ to the k-th *coordinate hyperplane* $\bar{H}_{e^{(k)},0}=\{\boldsymbol{x}=(x_0,x_1,\ldots,x_m)^T\in\mathbb{L}_K^m \mid \langle \boldsymbol{e}^{(k)},x\rangle=\boldsymbol{x}_k=0, k=1,2\ldots,m\}$ fixes the spatial coordinates, while the time coordinate follows from the hyperboloid constraint.

Theorem 1 (PLFC layer). *Let $\boldsymbol{x}\in\mathbb{L}_K^n$, $Z=\{\boldsymbol{z}_k\}_{k=1}^m\subset\mathbb{R}^n$ and $a=\{a_k\}_{k=1}^m\subset\mathbb{R}$. The* point–to–hyperplane Lorentz fully connected layer $\mathrm{PLFC}_K:\mathbb{L}_K^n\to\mathbb{L}_K^m$ *is*

$$v_k(\boldsymbol{x}) = v_{\boldsymbol{z}_k,a_k}(\boldsymbol{x}), \quad k=1,\ldots,m, \tag{8a}$$

$$y_{s,k} = \frac{1}{\sqrt{-K}}\sinh\!\big(\sqrt{-K}\,v_k(\boldsymbol{x})\big), \tag{8b}$$

$$y = \left[\sqrt{(-K)^{-1}+\|\boldsymbol{y}_s\|_2^2},\boldsymbol{y}_s\right], \tag{8c}$$

where $\boldsymbol{y}_s=(y_{s,1},\ldots,y_{s,m})^\top$. In the flat-space limit $K\to 0$, equation 8 reduces to the Euclidean affine map $\boldsymbol{y}=Av+b$ with $A_k=\boldsymbol{z}_k^\top$ and bias $b=a$.

It can be shown that the signed distance from $\boldsymbol{y}$ to each Lorentz hyperplane passing through the origin and orthogonal to the k-th coordinate axis is given by $v_k(\boldsymbol{x})$, as detailed in the Appendix E.1, thereby fulfilling the properties described above.

Gyro-bias. A learnable offset $\boldsymbol{b}\in\mathbb{L}_K^m$ can be added intrinsically via the gyroaddition, $\boldsymbol{y}\leftarrow\boldsymbol{y}\oplus\boldsymbol{b}$, yielding the final PLFC output.

Discussion. The previous Lorentz fully connected layer (LFC) used in HCNN was formed by

$$\mathbf{y}=\left[\sqrt{|\phi(W\mathbf{x},\mathbf{v})|^2-1/K},\phi(W\mathbf{x},\mathbf{v})\right],$$

where $\mathbf{x}\in\mathcal{L}_K^n$ and $W\mathbf{x}$ is a standard matrix–vector product in $\mathbb{R}^n$. This $W\mathbf{x}$ is defined using the ambient linear structure of the Minkowski space, not any operation on the Lorentz manifold so itself only partially intrinsic. The PLFC depends *only* on Lorentzian operations, avoids tangent-space linearisation, and enjoys closed-form gradients, making it an efficient and curvature-consistent replacement for Euclidean FC layers in hyperbolic networks. More detailed discussions are in Appendix D and a theoretical advantage of PLFC over LFC is shown in Appendix E.2.

4.2 Gyrogroup Lorentz Batch Normalization (GyroLBN)

Batch Normalization (BN) facilitates training by normalizing batch statistics. Recently, Bdeir et al. (2024) proposed LBN, which uses a Lorentzian centroid to efficiently compute the batch mean. However, their approach fails to normalize sample statistics. Besides, Chen et al. (2025a;b) extended BN into manifolds based on gyrogroup structures, referred to GyroBN (Chen et al., 2025a). Although it can normalize sample mean/variance on Lorentz spaces, the involved Fréchet mean could be inefficient. Therefore, we propose *GyroLBN* to combine the gyrogroup normalization with Lorentzian centroid & radius statistics, retaining GyroBN's effectiveness while eliminating computationally expensive Fréchet mean.

Batch centering and dispersion. Generally, the Fréchet mean must be solved iteratively, which significantly delays the training speed. Therefore, following Law et al. (2019); Bdeir et al. (2024), with $\boldsymbol{x}_i\in\mathbb{L}_K^n$, $\nu_i\geq 0,\sum_{i=1}^m\nu_i>0$ and a batch of features $B=\{x_i\in\mathbb{L}_K^n\}_{i=1}^m$, is given by, we define the batch mean by Lorentzian centroid which can be calculated efficiently in closed form (Law et al., 2019):

$$\mu_B = \frac{\sum_{i=1}^m \nu_i\boldsymbol{x}_i}{\sqrt{-K}\left|\|\sum_{i=1}^m\nu_i\boldsymbol{x}_i\|_{\mathcal{L}}\right|}, \tag{9}$$

which solves $\min_{\mu_B\in\mathbb{L}_K^n}\sum_{i=1}^m\nu_i d_{\mathcal{L}}^2(\boldsymbol{x}_i,\mu_B)$. Moreover, the mean is not weighted, which gives $\nu_i=\frac{1}{m}$.

As for dispersion, we adopt the Fréchet variance $\sigma^2\in\mathbb{R}^+$, defined as the expected squared Lorentzian distance between a point $\boldsymbol{x}_i$ and the mean μ_B, and given by $\sigma_B^2=\frac{1}{m}\sum_{i=1}^m d_{\mathcal{L}}^2(\boldsymbol{x}_i,\mu)$ (Kobler et al., 2022).

Normalization map. Denote learned scale $\gamma\in\mathbb{R}_{>0}$ (per-channel) and learned bias $\beta\in L_K^d$; For each sample x, the GyroLBN output is

$$\forall i\leq N,\quad \tilde{x}_i\leftarrow \overbrace{\beta\oplus}^{\text{Biasing}}\left(\overbrace{\frac{\gamma}{\sqrt{\sigma_B+\epsilon}}}^{\text{Scaling}}\odot\left(\overbrace{\ominus\mu_B\oplus x_i}^{\text{Centering}}\right)\right),\qquad \varepsilon>0. \tag{10}$$

where $\oplus$ denotes left gyroaddition Eq. 1, $\ominus$ denotes gyro inverse, and $\odot$ denotes gyro scalar product Eq. 2. Unless otherwise stated, during inference, we maintain per-channel running statistics by updating the Lorentzian centroid (mean) and dispersion (variance) with momentum and, at test time, substitute these for batch stats.

Discussion. GyroLBN unifies the *gyrogroup* normalization paradigm of GyroBN with the *efficient* Lorentz statistics of LBN, differing from GyroBN only in the choice of statistics by replacing Fréchet-based batch estimates with closed-form Lorentzian centroid and variance while retaining the same gyrogroup centering and scaling scheme. Compared to Fréchet variance used in generic Riemannian BN (Lou et al., 2020; Chen et al., 2022), the mean-radius statistic avoids iterative solvers and proved numerically stable in our vision/genomics settings. It therefore (i) remains fully intrinsic, (ii) avoids iterative Fréchet solvers (important for large batches and 2D convolutions), and (iii) integrates naturally with gyro-additive residual/bias layers used in our encoder. Compared to LBN and Fréchet-based GyroBN (Lou et al., 2020), GyroLBN consistently reduced wall-clock time while improving accuracy in our settings, as shown in Table 4.

4.3 Other Lorentz modules

Log-radius concatenation. When concatenating N Lorentz patches, each with $(1+d)$ coordinates (one time and d spatial), naively stacking the Nd spatial components biases the resulting feature norm toward higher dimensions: the expected radius grows with Nd, skewing subsequent layers. We introduce a *log-radius*–preserving concatenation that makes the expected *log* spatial radius invariant to the number of concatenated blocks. Let $v \in \mathbb{R}^{n_i}$ denote the spatial part of a block and assume its radius factorizes as $\|v\| = \sigma\sqrt{T}$ with $T \sim \chi^2_{n_i}$. Then

$$\mathbb{E}[\log \|v\|] = \log \sigma + \tfrac{1}{2}\big(\psi\big(\tfrac{n_i}{2}\big) + \log 2\big)\,,$$

where ψ is the digamma function. To keep $\mathbb{E}[\log \|v\|]$ constant across dimensions, we scale each block's spatial part by

$$s(n, n_i) \;=\; \exp\left(\frac{1}{2}\Big[\psi\Big(\frac{n}{2}\Big) - \psi\Big(\frac{n_i}{2}\Big)\Big]\right),$$

with $n = Nd$ the total post-concat spatial dimension and $n_i = d$ the per-block spatial dimension. Concretely, given per-window tensors $(t_1, \ldots, t_N)$ (times) and $(u_1, \ldots, u_N)$ with $u_i \in \mathbb{R}^d$ (spaces), we form the scaled space $\tilde{u}_i = s\, u_i$ and recompute a single time coordinate that keeps the output on the hyperboloid:

$$t' \;=\; \sqrt{-\tfrac{1}{K} + s^2 \sum_{i=1}^{N} (t_i^2 + \tfrac{1}{K})}\,,$$

where $k > 0$ sets the origin time via $t_0 = \sqrt{k}$ (in practice, with $K = -1$, $k = 1$). The final concatenated vector is $[\, t', \tilde{u}_1, \ldots, \tilde{u}_N \,] \in \mathbb{R}^{1+Nd}$. This *log-radius* alignment (i) is parameter-free, (ii) is robust to heavy-tailed radii because it matches *geometric* means, (iii) preserves the Lorentz constraint by design, and (iv) avoids domination by any single wide block, improving stability as kernel size or channel count grows.

Lorentz convolutional layer. We use channel-last feature map representations throughout HCNNs, and add the Lorentz model's time component as an additional channel dimension, following Bdeir et al. (2024). A hyperbolic feature map can be define as an ordered set of n-dimensional hyperbolic vectors, where every spatial position contains a vector that can be combined with its neighbors

The convolutional layer can be formulated as a matrix multiplication between a linearized kernel and the concatenation of values within its receptive field (Shimizu et al., 2020). Then, we extend this definition by replacing the Euclidean FC and concatenation with our PLFC and *log-radius*–preserving concatenation.

Let $\mathbf{x} = \{\boldsymbol{x}_{h,w} \in \mathbb{L}^n_K\}_{h,w=1}^{H,W}$ be an input hyperbolic feature map and let $\tilde{H} \times \tilde{W}$ denotes the kernel size with stride δ. For each spatial location (h, w) we gather the patch $\{\boldsymbol{x}_{h+\delta\tilde{h},\, w+\delta\tilde{w}}\}_{\tilde{h},\tilde{w}=1}^{\tilde{H},\tilde{W}} \subset \mathbb{L}^n_K$, pad with the origin $(\sqrt{1/(-K)}, 0, \ldots, 0)$ and concatenate these $\tilde{H}\tilde{W}$ vectors by the *log-radius* scheme introduced above. We defined the Lorentz convolutional layer as

$$\boldsymbol{y}_{h,w} = \mathrm{PLFC}(\mathrm{LogCat}(\{\boldsymbol{x}_{h'+\delta\tilde{h},w'+\delta\tilde{w}} \in \mathbb{L}^n_K\}_{\tilde{h},\tilde{w}=1}^{\tilde{H},\tilde{W}})), \tag{11}$$

where h' and w' denote the starting position, and LogCat denotes our *log-radius*–preserving concatenation.

Lorentz dropout. To regularize features without leaving the manifold, we adopt a dropout operator that acts on Lorentz coordinates and then *reprojects* to the hyperboloid. Concretely, during training, we apply an elementwise Bernoulli mask to the current representation $x \in \mathbb{L}^n_K$ (probability p of zeroing each entry), yielding $\tilde{x}$ in ambient $\mathbb{R}^{n+1}$. Because naive masking can violate the hyperboloid constraint and the positivity of the time

component, we immediately map back via a projection $\mathrm{proj}_{\mathcal{L}}(\tilde{x})$ that restores $\langle x, x\rangle_{\mathcal{L}} = -1/K$ and $x_0 > 0$. At evaluation, the operator is the identity. In practice, we observe that the "mask & project" scheme outperforms the variant "log–exp" scheme that applies a log map to $T_0\mathbb{L}_K^n$, performs Euclidean dropout, and then returns via the exponential map, because the nonlinearity of $\exp_0$ couples all coordinates so that masking any tangent component perturbs the entire point after mapping back to the manifold. It is parameter-free, numerically stable, and compatible with subsequent intrinsic layers (e.g., GyroLBN and PLFC), since its output again lies on $\mathbb{L}_K^n$.

Lorentz activation. Following Bdeir et al. (2024), we define activations *directly* in the Lorentz model by acting on the spatial coordinates only and then recomputing the time coordinate from the hyperboloid constraint. For example, given $x = (x_0, x_s) \in L_K^n$ and an elementwise nonlinearity activation function ReLU, *Lorentz ReLU* can be defined as

$$\mathrm{L{-}ReLU}(x) \;=\; \Big[\sqrt{\tfrac{1}{-K} + \|\mathrm{ReLU}(x_s)\|_2^2}\;,\; \mathrm{ReLU}(x_s)\Big].$$

5 EXPERIMENT

We evaluate ILNN on image classification dataset CIFAR-10/100 (Krizhevsky et al., 2009) and genomics (TEB (Khan et al., 2025), GUE (Zhou et al., 2023)) and compare against Euclidean and multiple hyperbolic baselines under matched training recipes. For fairness, each Euclidean backbone is translated to the hyperbolic setting via *one-to-one* module replacement, keeping depth/width, parameter count, and schedule as close as possible. All models are implemented in PyTorch (Paszke et al., 2019) with 32-bit precision. We fix curvature to $K = -1$ and train with Riemannian optimizers from Geoopt (Kochurov et al., 2020): RiemannianSGD for CIFAR and RiemannianAdam for genomics. Unless otherwise stated, results are averaged over five random seeds and reported as mean ± std; classification uses accuracy on CIFAR-10/100, and MCC on genomics. All experiments are conducted on NVIDIA A100 80GB GPUs. Additional configuration details (batch size, learning rate) appear in Appendix F.

5.1 IMAGE CLASSIFICATION

Experimental setup. Following the evaluation setup of HCNN (Bdeir et al., 2024) and symmetric space (Nguyen et al., 2025), we benchmark ILNN on CIFAR-10 and CIFAR-100. For each method, we instantiate a ResNet-18 backbone (He et al., 2015) in the corresponding geometry: The original ResNet-18 with BN and linear classifier (Euclidean); Euclidean encoder + hyperbolic head (Poincaré or Lorentz); all layers fully to the target manifold (HCNN (Bdeir et al., 2024) and our ILNN). The relative distortion from input manifold to classifier (Bdeir et al., 2024) is comparable across runs (δ_{rel}=0.26 for CIFAR-10; 0.23 for CIFAR-100).

Main results. Table 1 summarizes test accuracy as mean±sd over five runs. ILNN attains the highest average accuracy on both datasets, 95.36% on CIFAR-10 and 78.41% on CIFAR-100, exceeding the Euclidean ResNet-18 baseline by +0.22 and +0.69 percentage points (pp), respectively. Relative to the strongest prior hyperbolic competitor (HCNN-Lorentz), ILNN also improves by +0.22 pp on CIFAR-10 and +0.34 pp on CIFAR-100. Although ILNN exhibits a slightly larger standard deviation than HCNN-Lorentz, on CIFAR-10, even the conservative lower bound of ILNN (95.23%) exceeds the upper bound of HCNN (95.22%); on CIFAR-100, the lower bound of ILNN is specifically 78.24%, equal to the upper bound of HCNN. These gains demonstrate that our intrinsic approach preserves the geometry of the Lorentz model without distortion: by combining point-to-hyperplane FC and GyroBN within fully negative-curvature space, ILNN leverages the native manifold structure more effectively than others.

Table 1: Classification accuracy (%) of ResNet-18 models. We estimate the mean and standard deviation from five runs. The best performance is highlighted in bold (higher is better).

	CIFAR-10 ($\delta_{\text{rel}} = 0.26$)	CIFAR-100 ($\delta_{\text{rel}} = 0.23$)
Euclidean (He et al., 2015)	95.14 ± 0.12	77.72 ± 0.15
Hybrid Poincaré (Guo et al., 2022)	95.04 ± 0.13	77.19 ± 0.50
Poincaré ResNet (Van Spengler et al., 2023)	94.51 ± 0.15	76.60 ± 0.32
Euclidean-Poincaré-H (Fan et al., 2023)	81.72 ± 7.84	44.35 ± 2.93
Euclidean-Poincaré-G (Ganea et al., 2018)	95.14 ± 0.11	77.78 ± 0.09
Euclidean-Poincaré-B (Nguyen et al., 2025)	95.23 ± 0.08	77.78 ± 0.15
Hybrid Lorentz (Bdeir et al., 2024)	94.98 ± 0.12	78.03 ± 0.21
HCNN Lorentz (Bdeir et al., 2024)	95.14 ± 0.08	78.07 ± 0.17
ILNN (ours)	$\mathbf{95.36 \pm 0.13}$	$\mathbf{78.41 \pm 0.23}$

Visualization. We visualize embeddings by mapping network outputs from the Lorentz model to the Poincaré ball and, in parallel, applying the logarithmic map at the origin to view them in the tangent plane; colors indicate output label prediction. In Figure 2 (CIFAR-10), the ILNN embeddings form ten compact clusters, whose areas

are visibly smaller than the corresponding clusters produced by HCNN. Only marginal colour bleeding occurs at the boundaries, indicating that the decision margins learned by the point-to-hyperplane FC induce margins aligned with the data geometry. The effect becomes even more pronounced on the harder CIFAR-100 task, as shown in Figure 3. Despite packing 100 classes into the same 2D manifold, ILNN still yields dense colour "islands" with clear gaps in between, whereas HCNN exhibits overlapping clouds and several mixed-colour zones. These visual trends are consistent with the quantitative improvements in Table 1.

5.2 Genomic Classification

Experimental setup. We evaluate on the most challenging subsets of the **TEB** and **GUE** genomics benchmarks, **specifically those on which the published HCNN (Khan et al., 2025) does not surpass a Euclidean convolutional baseline.** For every dataset, we instantiate four models that share the same convolutional stem and classifier width: In our comparisons, the Euclidean CNN uses standard convolutions, Euclidean BN, and a linear classifier; HCNN-S employs Lorentz modules with a single global curvature K across the network; HCNN-M uses Lorentz modules with layer-wise curvatures K; and ILNN is fully intrinsic, featuring GyroBN and a point-to-hyperplane FC with gyro-bias, while fixing a global curvature $K = -1$. All hyperbolic models are optimized with RiemannianAdam (lr 10^{-3}); the Euclidean baseline uses Adam with an identical schedule. We measure performance by the Matthews correlation coefficient (MCC).

Main results. Table 2 shows that ILNN achieves the best score on every task. On the two TEB pseudogene sets, it improves over the Euclidean baseline by +9.6 and +13.0 pp, and still exceeds the stronger HCNN-S by +2.0 and 8.8 pp, respectively, demonstrating clear gains where previous hyperbolic models were already competitive. The advantage is even more striking on GUE. For Covid-variant classification, HCNN collapses (MCC 36.7/14.8) much lower than the Euclidean model, whereas ILNN matches and slightly surpasses the Euclidean score (64.8 vs. 63.6). On promoter-related tasks, ILNN consistently raises the bar: Tata core-promoter detection jumps from 79.9 (best prior) to 83.9, and the most difficult "all" split goes from 67.2 to 70.7. Across the board, ILNN tightens the worst-case gap between hyperbolic and Euclidean models and converts several cases of HCNN under-performance into clear wins. These results confirm that fully intrinsic design choices, point-to-hyperplane FC, GyroLBN, and other proposed components, translate into tangible accuracy gains on real-world genomic data.

Table 2: Model performance (MCC) on all real-world genomics datasets, averaged over five random seeds (mean ± standard deviation). The two highest-scoring models are in bold. * denotes that the result was reproduced under the same setting.

Benchmark	Task	Dataset	Model			
			Euclidean CNN	Hyperbolic HCNN-S	Hyperbolic HCNN-M	Hyperbolic ILNN
TEB	Pseudogenes	processed	60.66±0.82	68.30±0.93	65.41±5.54	**70.26±0.32**
		unprocessed	51.94±2.69	56.13±0.56	58.36±1.80	**64.90±0.74**
GUE	Covid Variant Classification	Covid	63.62±1.34*	36.71±9.69	14.81±0.46	**64.76±0.54**
	Core Promoter Detection	tata	78.26±2.85	79.54±1.61	79.87±2.50	**83.90±0.53**
		notata	66.60±1.07	66.52±0.28	65.95±0.51	**72.59**±0.69
		all	66.47±0.74	65.26±1.11	67.16±0.55	**70.89±0.43**
	Promoter Detection	tata	78.58±3.39	79.74±2.66	78.77±0.78	**83.26±1.90**
		notata	90.81±0.51	89.86±0.76	90.28±0.37	**92.48±0.35**
		all	88.00±0.39	87.60±0.51	87.93±0.76	**91.34±0.38**

5.3 Graphs

Experimental setup. To further evaluate the effectiveness of our method, we extend it to three widely-used graph datasets (Airport, Cora and PubMed), which exhibit intricate topological and hierarchical relationships, making them an ideal testbed for evaluating the effectiveness of hyperbolic networks (e.g., HGNN, HGCN, HGAT, HAN, HNN++ and Hypformer). We choose the Hypformer as baseline and replace the Linear layer in Hypformer.

Main results. The quantitative results are summarized in Table 3. Overall, hyperbolic models consistently outperform their Euclidean counterparts on all three benchmarks, confirming that negatively curved representations are well-suited for graphs with rich hierarchical structure. Among existing methods, HAN, HNN++, and SGFormer constitute the strongest baselines, while Hypformer further improves their performance. Building on

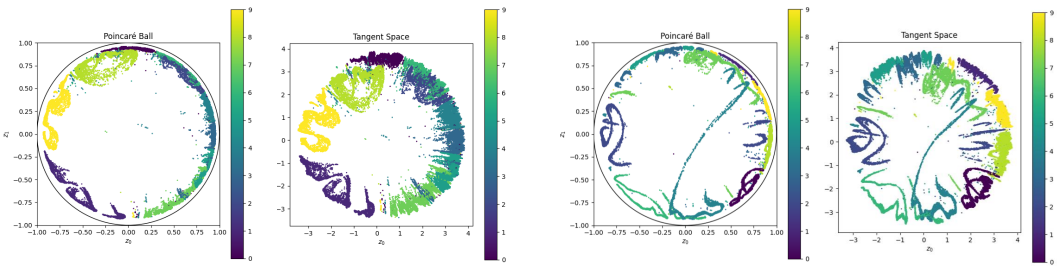

Figure 2: Embedding visualization of CIFAR-10 dataset in Poincaré and Tangent Space. Colors represent labels. HCNN (94.98, left) and ILNN (95.48, right).

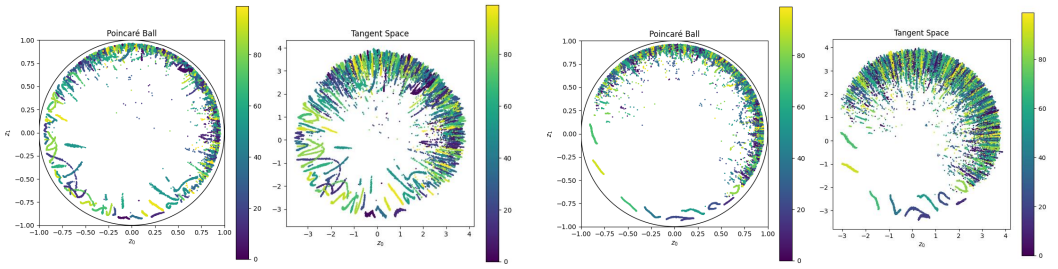

Figure 3: Embedding visualization of CIFAR-100 dataset in Poincaré and Tangent Space. Colors represent labels. HCNN (77.67, left) and ILNN (78.64, right).

this backbone, Hypformer+PLFC achieves the best results on all datasets, reaching $96.03\%\pm0.34$ on AIRPORT, $85.68\%\pm0.19$ on CORA, and $82.52\%\pm0.33$ on PUBMED. Compared with the original Hypformer, this corresponds to absolute gains of $+1.03$, $+0.68$, and $+1.22$ percentage points, respectively. The standard deviations remain small, and the confidence intervals of Hypformer+PLFC are largely separated from those of competing methods on AIRPORT and PUBMED, indicating that the observed gains are statistically stable. These results demonstrate that replacing the Euclidean linear classifier in Hypformer with our hyperbolic point-to-hyperplane PLFC layer provides a simple yet effective way to enhance graph representation learning on graph benchmarks.

5.4 ABLATION STUDY

To better understand the contributions of each architectural component, we conduct an ablation study on our two main innovations: the point-to-hyperplane fully connected head (PLFC) and the Gyrogroup Lorentz Batch Normalization (GyroLBN). We compare them against their Lorentz counterparts (LFC, LBN) as well as GyroBN, measuring both classification accuracy across CIFAR-10 and genomics tasks, and training efficiency to disentangle the effects of the classifier design and the normalization strategy, and to assess whether their combination leads to complementary improvements.

Table 3: Testing results (Accuracy) on AIRPORT, CORA and PUBMED. The best results are in bold, respectively.

Models	AIRPORT	CORA	PUBMED
GCN (Kipf & Welling, 2017)	81.4 ± 0.6	81.3 ± 0.3	78.1 ± 0.2
GAT (Veličković et al., 2017)	81.5 ± 0.3	83.0 ± 0.7	79.0 ± 0.3
SGC (Wu et al., 2019)	82.1 ± 0.5	80.1 ± 0.2	78.7 ± 0.1
HGNN (Liu et al., 2019)	84.7 ± 1.0	77.1 ± 0.8	78.3 ± 1.2
HGCN (Chami et al., 2019)	89.3 ± 1.2	76.5 ± 0.6	78.0 ± 1.0
HGAT (Chami et al., 2019)	89.6 ± 1.0	77.4 ± 0.7	78.3 ± 1.4
GraphFormer (Ying et al., 2021)	88.1 ± 1.2	60.0 ± 0.5	73.3 ± 0.7
GraphTrans (Wu et al., 2021)	94.3 ± 0.6	77.6 ± 0.8	77.5 ± 0.7
GraphGPS (Rampášek et al., 2022)	94.5 ± 0.9	73.0 ± 1.4	72.8 ± 1.4
FPS-T (Cho et al., 2023)	96.0 ± 0.6	82.3 ± 0.7	78.5 ± 0.6
HAN (Gulcehre et al., 2019)	92.9 ± 0.6	83.1 ± 0.5	79.0 ± 0.6
HNN++ (Shimizu et al., 2020)	92.3 ± 0.3	82.8 ± 0.6	79.9 ± 0.4
F-HNN (Chen et al., 2022)	93.0 ± 0.7	81.0 ± 0.7	77.5 ± 0.8
NodeFormer (Wu et al., 2022)	80.2 ± 0.6	82.2 ± 0.9	79.9 ± 1.0
SGFormer (Wu et al., 2023)	92.9 ± 0.5	83.2 ± 0.9	80.0 ± 0.8
Hypformer (Yang et al., 2024b)	95.0 ± 0.5	85.0 ± 0.3	81.3 ± 0.3
Hypformer+PLFC	$\mathbf{96.0 \pm 0.3}$	$\mathbf{85.7 \pm 0.2}$	$\mathbf{82.5 \pm 0.3}$

PLFC vs. LFC. Holding the normalizer fixed, replacing the Lorentz fully connected head with the point-to-hyperplane fully connected head consistently improves accuracy. With GyroLBN, PLFC outperforms LFC on CIFAR-10 (95.36 vs. 95.19,

Table 4: Ablation on PLFC and GyroLBN. Fit time denotes the training time of the CIFAR-10 dataset per epoch.

Benchmark	Task	Dataset	Model			
			LFC LBN	LFC GyroLBN	PLFC GyroBN	PLFC GyroLBN
CIFAR-10			95.14±0.08	95.19±0.15	95.28±0.17	**95.36**±0.13
GUE	Core Promoter Detection	tata	78.26±2.85	81.33±3.19	80.89±3.11	**83.90±0.53**
		notata	66.60±1.07	71.92±0.52	72.22±0.82	**72.59**±0.69
		all	66.47±0.74	69.74±1.3	70.14±0.45	**70.89±0.43**
	Promoter Detection	tata	78.58±3.39	80.46±0.99	81.16±1.99	**83.26±1.90**
		notata	90.81±0.51	91.88±1.01	91.67±0.49	**92.48±0.35**
		all	88.00±0.39	90.28±1.04	91.02±0.56	**91.34±0.38**
Fit Time(s)			142	125	314	169

+0.17) and on all six genomics subsets, with gains of +2.57 on *tata* core-promoter detection (83.90 vs. 81.33), +0.67 on *notata* (72.59 vs. 71.92), and +0.97 on the *all* split (70.89 vs. 69.74); for promoter detection, the improvements are +2.80 (*tata*: 83.26 vs. 80.46), +0.60 (*notata*: 92.48 vs. 91.88), and +1.06 (*all*: 91.34 vs. 90.28). Although the PLFC variant with GyroBN also performs strongly, exceeding the LFC baseline by more than +3 on all genomics subsets and matching or surpassing it on CIFAR-10 (95.28 vs. 95.14). These results indicate that decision functions based on point-to-hyperplane distance provide a more effective inductive bias than affine logits in both vision and genomics.

GyroLBN vs. LBN, GyroBN. Under the same FC, GyroLBN improves over LBN across all tasks. With the LFC head, CIFAR-10 increases from 95.14 to 95.19; on genomics, the gains range from +1.07 to +5.32 (e.g., core-promoter *notata*: 71.92 vs. 66.60). With the PLFC head, GyroLBN is superior to GyroBN on every dataset, including *notata* in core-promoter detection, where it reaches 72.59 versus 72.22. The improvements span +0.32 to +3.01 on genomics and +0.08 on CIFAR-10. Training time measurements show that GyroLBN attains these gains with favorable efficiency, running faster than PLFC with GyroBN (169 s vs. 314 s) and even faster than LBN for the comparison of LFC between GyroLBN and LBN (125 s vs. 142 s). Overall, GyroLBN offers a better accuracy and efficiency trade-off than both LBN and GyroBN in our settings.

Furthermore, to ensure a fair comparison with GyroBN, we further vary the number of Fréchet mean iterations used by the normalizers (1, 2, 5, 10, and a fixed-point solve denoted by ∞; see Appendix G). Across all iteration cases, *PLFC+GyroLBN* remains the best-performing configuration on CIFAR-10 and on every genomics split, exhibiting the effectiveness of our GyroLBN again.

6 Conclusion

This study presented the Intrinsic Lorentz Neural Network, an architecture whose computations remain entirely within the Lorentz model of hyperbolic space. We introduced a point-to-hyperplane fully connected layer that converts signed hyperbolic distances into logits, together with GyroLBN and log-radius concatenation for numerically stable normalization and feature aggregation. Integrated into a coherent network, these components yield superior performance on CIFAR-10, CIFAR-100, and two challenging genomics benchmarks, surpassing Euclidean, hybrid, and prior hyperbolic baselines while preserving competitive training cost. The results underscore the value of keeping every layer intrinsic to the manifold and provide practical building blocks for future work on representation learning in negatively curved geometries.

Reproducibility Statement

All theoretical results are established under explicit conditions. Experimental details are given in Appendix F. The code will be released upon acceptance.

Ethics Statement

This work only uses publicly available datasets and does not involve human subjects or sensitive information. We identify no specific ethical concerns.

REFERENCES

Ivana Balazevic, Carl Allen, and Timothy Hospedales. Multi-relational poincaré graph embeddings. *Advances in neural information processing systems*, 32, 2019.

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

James W. Cannon, William J. Floyd, Richard Kenyon, and Walter R. Parry. Hyperbolic geometry. In Silvio Levy (ed.), *Flavors of Geometry*, volume 31 of *MSRI Publications*. Mathematical Sciences Research Institute, 1997.

Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. Hyperbolic graph convolutional neural networks. *Advances in neural information processing systems*, 32, 2019.

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 (Volume 1: Long Papers)*, pp. 5672–5686, 2022.

Ziheng Chen, Yue Song, Xiao-Jun Wu, and Nicu Sebe. Gyrogroup batch normalization. In *International Conference on Learning Representations (ICLR)*, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/fb19caf91f4966e426026685028cb73a-Paper-Conference.pdf.

Ziheng Chen, Xiao-Jun Wu, and Nicu Sebe. Riemannian batch normalization: A gyro approach. *arXiv preprint arXiv:2509.07115*, 2025b.

Sungjun Cho, Seunghyuk Cho, Sungwoo Park, Hankook Lee, Honglak Lee, and Moontae Lee. Curve your attention: Mixed-curvature transformers for graph representation learning. *arXiv preprint arXiv:2309.04082*, 2023.

Xiran Fan, Chun-Hao Yang, and Baba C. Vemuri. Horospherical decision boundaries for large margin classification in hyperbolic space. *arXiv preprint arXiv:2302.06807*, 2023.

Xiran Fan, Minghua Xu, Huiyuan Chen, Yuzhong Chen, Mahashweta Das, and Hao Yang. Enhancing hyperbolic knowledge graph embeddings via lorentz transformations. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 4575–4589, 2024. URL https://aclanthology.org/2024.findings-acl.272.pdf.

Octavian Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. *Advances in neural information processing systems*, 31, 2018.

Mina Ghadimi Atigh, Julian Schoep, Erman Acar, Nanne van Noord, and Pascal Mettes. Hyperbolic image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4429–4439, 2022.

Caglar Gulcehre, Misha Denil, Mateusz Malinowski, Ali Razavi, Razvan Pascanu, Karl Moritz Hermann, Peter Battaglia, Victor Bapst, David Raposo, Adam Santoro, et al. Hyperbolic attention networks. In *ICLR*, 2019.

Y. Guo, X. Wang, Y. Chen, and S. X. Yu. Clipped hyperbolic classifiers are super-hyperbolic classifiers. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–10, Los Alamitos, CA, USA, jun 2022. IEEE Computer Society. doi: 10.1109/CVPR52688.2022.00010. URL https://doi.ieeecomputersociety.org/10.1109/CVPR52688.2022.00010.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385.

Neil He, Menglin Yang, and Rex Ying. Lorentzian residual neural networks. *arXiv preprint arXiv:2412.14695*, 2024. URL https://arxiv.org/abs/2412.14695.

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. *arXiv preprint arXiv:2505.24722*, 2025.

Raiyan R Khan, Philippe Chlenski, and Itsik Pe'er. Hyperbolic genome embeddings. *arXiv preprint arXiv:2507.21648*, 2025.

Andrei Khrulkov, Leyla Mirvakhabova, Evgeniya Ustinova, Ivan Oseledets, and Victor Lempitsky. Hyperbolic image embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6418–6428, 2020.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

Reinmar J. Kobler, Jun-ichiro Hirayama, and Motoaki Kawanabe. Controlling the fréchet variance improves batch normalization on the symmetric positive definite manifold. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3863–3867, 2022. doi: 10.1109/ICASSP43922.2022.9746629.

Max Kochurov, Rasul Karimov, and Serge Kozlukov. Geoopt: Riemannian optimization in pytorch. *arXiv preprint arXiv:2005.02819*, 2020. URL https://arxiv.org/abs/2005.02819.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Marc T. Law, Renjie Liao, Jake Snell, and Richard S. Zemel. Lorentzian distance learning for hyperbolic representations. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3672–3681. PMLR, 2019. URL https://proceedings.mlr.press/v97/law19a.html.

Guy Lebanon and John Lafferty. Hyperplane margin classifiers on the multinomial manifold. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 66, 2004.

Zhiying Leng, Tolga Birdal, Xiaohui Liang, and Federico Tombari. Hypersdfusion: Bridging hierarchical structures in language and geometry for enhanced 3d text2shape generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/papers/Leng_HyperSDFusion_Bridging_Hierarchical_Structures_in_Language_and_Geometry_for_Enhanced_CVPR_2024_paper.pdf.

Yancong Li, Xiaoming Zhang, Ying Cui, and Shuai Ma. Hyperbolic graph neural network for temporal knowledge graph completion. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 8474–8486, Torino, Italia, 2024. ELRA and ICCL. URL https://aclanthology.org/2024.lrec-main.743/.

Qiuyu Liang, Weihua Wang, Feilong Bao, and Guanglai Gao. Fully hyperbolic rotation for knowledge graph embedding. *arXiv preprint arXiv:2411.03622*, 2024. URL https://arxiv.org/abs/2411.03622.

Qi Liu, Maximilian Nickel, and Douwe Kiela. Hyperbolic graph neural networks. In *NeurIPS*, pp. 8230–8241, 2019.

Aaron Lou, Nina Miolane, Tien-Minh Le, Xavier Pennec, Susan Holmes, and Quoc Gu. Differentiating through the fréchet mean. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pp. 6393–6403, 2020. URL https://proceedings.mlr.press/v119/lou20a.html.

Gal Mishne, Zhengchao Wan, Yusu Wang, and Sheng Yang. The numerical stability of hyperbolic representation learning. In *International Conference on Machine Learning*, pp. 24925–24949. PMLR, 2023.

Xuan Son Nguyen and Shuo Yang. Building neural networks on matrix manifolds: A gyrovector space approach. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*. PMLR, 2023.

Xuan Son Nguyen, Boris Muzellec, and Agnieszka Słowik. A gyrovector space approach for symmetric positive semi-definite matrices. In *Proceedings of the 17th European Conference on Computer Vision (ECCV)*. Springer, 2022.

Xuan Son Nguyen, Shuo Yang, and Aymeric Histace. Matrix manifold neural networks++. *arXiv preprint arXiv:2405.19206*, 2024.

Xuan Son Nguyen, Shuo Yang, and Aymeric Histace. Neural networks on symmetric spaces of noncompact type. In *The Thirteenth International Conference on Learning Representations*, 2025.

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*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3776–3785. PMLR, 2018.

Avik Pal, Max van Spengler, Guido Maria D'Amely di Melendugno, Alessandro Flaborea, Fabio Galasso, and Pascal Mettes. Compositional entailment learning for hyperbolic vision-language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html.

Eric Qu and Dongmian Zou. Lorentzian fully hyperbolic generative adversarial network. *arXiv preprint arXiv:2201.12825*, 2022.

Ladislav Rampášek, Michael Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. In *NeurIPS*, volume 35, pp. 14501–14515, 2022.

John G. Ratcliffe. *Foundations of Hyperbolic Manifolds*, volume 149 of *Graduate Texts in Mathematics*. Springer, New York, 2 edition, 2006.

Ryohei Shimizu, Yusuke Mukuta, and Tatsuya Harada. Hyperbolic neural networks++. *arXiv preprint arXiv:2006.08210*, 2020.

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

Alexandru Tifrea, Gary Bécigneul, and Octavian-Eugen Ganea. Poincaré glove: Hyperbolic word embeddings. *arXiv preprint arXiv:1810.06546*, 2019. URL https://arxiv.org/abs/1810.06546.

Abraham A. Ungar. *Analytic Hyperbolic Geometry and Albert Einstein's Special Theory of Relativity*. World Scientific, 2008.

Abraham Albert Ungar. *Analytic Hyperbolic Geometry in N Dimensions: An Introduction*. CRC Press, Boca Raton, 2014.

Abraham Albert Ungar. *Analytic Hyperbolic Geometry and Albert Einstein's Special Theory of Relativity (Second Edition)*. World Scientific, 2022.

Max Van Spengler, Erwin Berkhout, and Pascal Mettes. Poincare resnet. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5419–5428, 2023.

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

Simon Welz, Lucie Flek, and Akbar Karimi. Multi-hop reasoning for question answering with hyperbolic representations. *arXiv preprint arXiv:2507.03612*, 2025.

Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *ICML*, pp. 6861–6871. PMLR, 2019.

Qitian Wu, Wentao Zhao, Zenan Li, David P Wipf, and Junchi Yan. Nodeformer: A scalable graph structure learning transformer for node classification. In *NeurIPS*, volume 35, pp. 27387–27401, 2022.

Qitian Wu, Wentao Zhao, Chenxiao Yang, Hengrui Zhang, Fan Nie, Haitian Jiang, Yatao Bian, and Junchi Yan. Simplifying and empowering transformers for large-graph representations. In *NeurIPS*, volume 36, 2023.

Zhanghao Wu, Paras Jain, Matthew Wright, Azalia Mirhoseini, Joseph E Gonzalez, and Ion Stoica. Representing long-range context for graph neural networks with global attention. In *NeurIPS*, volume 34, pp. 13266–13279, 2021.

Jiaqi Yang, Wenting Chen, Xiaohan Xing, Sean He, Xiaoling Luo, Xinheng Lyu, Linlin Shen, and Guoping Qiu. Hysurvpred: Multimodal hyperbolic embedding with angle-aware hierarchical contrastive learning and uncertainty constraints for survival prediction. *arXiv preprint arXiv:2503.13862*, 2025.

Menglin Yang, Min Zhou, Zhihao Li, Jiahong Liu, Lujia Pan, Hui Xiong, and Irwin King. Hyperbolic graph neural networks: A review of methods and applications. *arXiv preprint arXiv:2202.13852*, 2022.

Menglin Yang, Aosong Feng, Bo Xiong, Jihong Liu, Irwin King, and Rex Ying. Hyperbolic fine-tuning for large language models. *arXiv preprint arXiv:2410.04010*, 2024a.

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, 2024b.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? In *NeurIPS*, 2021.

Yuansheng Zhou and Tatyana O. Sharpee. Hyperbolic geometry of gene expression. *iScience*, 24(3):102225, 2021. doi: 10.1016/j.isci.2021.102225.

Zhihan Zhou, Yanrong Ji, Weijian Li, Pratik Dutta, Ramana Davuluri, and Han Liu. Dnabert-2: Efficient foundation model and benchmark for multi-species genome. *arXiv preprint arXiv:2306.15006*, 2023.

APPENDIX

A USE OF LARGE LANGUAGE MODELS

We use large language models to aid or polish writing.

B OPERATIONS IN THE LORENTZ MODEL

Setup and notation Fix a negative curvature $K < 0$. Let $\langle \cdot, \cdot \rangle_{\mathcal{L}}$ denote the Minkowski bilinear form with signature $(-, +, \ldots, +)$ on $\mathbb{R}^{n+1}$, and write $\| \cdot \|_{\mathcal{L}} = \sqrt{\langle \cdot, \cdot \rangle_{\mathcal{L}}}$ for the induced (Riemannian) norm on tangent vectors. The n-dimensional hyperbolic space in the Lorentz (hyperboloid) model is

$$\mathbb{L}_K^n := \Big\{ \mathbf{x} \in \mathbb{R}^{n+1} \ : \ \langle \mathbf{x}, \mathbf{x} \rangle_{\mathcal{L}} = \tfrac{1}{K}, \ \ x_0 > 0 \Big\},$$

and we use $\overline{\mathbf{0}} := \big(\frac{1}{\sqrt{-K}}, \mathbf{0}\big)$ as the pole ("origin"). When no confusion can arise, $\| \cdot \|$ denotes the Euclidean norm in a tangent space.

Distance For $\mathbf{x}, \mathbf{y} \in \mathbb{L}_K^n$, the geodesic distance inherited from Minkowski space is

$$D_{\mathcal{L}}(\mathbf{x}, \mathbf{y}) = \frac{1}{\sqrt{-K}} \ \cosh^{-1}\big(K \, \langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}}\big).$$

A useful identity for computations is the "Lorentzian chord" expression (squared distance)

$$d_{\mathcal{L}}^2(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_{\mathcal{L}}^2 = \frac{2}{K} - 2 \, \langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}}.$$

Specializing to the pole $\overline{\mathbf{0}}$,

$$D_{\mathcal{L}}(\mathbf{x}, \overline{\mathbf{0}}) = \big\| \log_{\overline{\mathbf{0}}}^K(\mathbf{x}) \big\|, \qquad d_{\mathcal{L}}^2(\mathbf{x}, \overline{\mathbf{0}}) = \frac{2}{K} - 2 \, \langle \mathbf{x}, \overline{\mathbf{0}} \rangle_{\mathcal{L}} = \frac{2}{K} + \frac{2 \, x_0}{\sqrt{-K}},$$

and, equivalently,

$$D_{\mathcal{L}}(\mathbf{x}, \overline{\mathbf{0}}) = \frac{1}{\sqrt{-K}} \ \cosh^{-1}\big(\sqrt{-K} \, x_0\big).$$

Tangent space The tangent space at $\mathbf{x} \in \mathbb{L}_K^n$ is the Minkowski-orthogonal complement of $\mathbf{x}$,

$$\mathcal{T}_{\mathbf{x}} \mathbb{L}_K^n = \big\{ \mathbf{v} \in \mathbb{R}^{n+1} \ : \ \langle \mathbf{v}, \mathbf{x} \rangle_{\mathcal{L}} = 0 \big\}.$$

Restricted to $\mathcal{T}_{\mathbf{x}} \mathbb{L}_K^n$, the metric is positive definite and coincides with the Riemannian metric of $\mathbb{L}_K^n$.

Exponential and logarithmic maps For $\mathbf{z} \in \mathcal{T}_{\mathbf{x}} \mathbb{L}_K^n$,

$$\exp_{\mathbf{x}}^K(\mathbf{z}) = \cosh(\alpha) \, \mathbf{x} \ + \ \sinh(\alpha) \, \frac{\mathbf{z}}{\alpha}, \qquad \alpha = \sqrt{-K} \, \|\mathbf{z}\|_{\mathcal{L}}.$$

The inverse map $\log_{\mathbf{x}}^K : \mathbb{L}_K^n \to \mathcal{T}_{\mathbf{x}} \mathbb{L}_K^n$ sends $\mathbf{y} \in \mathbb{L}_K^n$ to

$$\log_{\mathbf{x}}^K(\mathbf{y}) = \frac{\cosh^{-1}(\beta)}{\sqrt{\beta^2 - 1}} \ \big(\mathbf{y} - \beta \, \mathbf{x}\big), \qquad \beta = K \, \langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}}.$$

At the pole $\overline{\mathbf{0}}$ these simplify to

$$\exp_{\overline{\mathbf{0}}}^K(\mathbf{z}) = \frac{1}{\sqrt{-K}} \left(\cosh\big(\sqrt{-K} \, \|\mathbf{z}\|\big) \, , \ \sinh\big(\sqrt{-K} \, \|\mathbf{z}\|\big) \ \frac{\mathbf{z}}{\|\mathbf{z}\|} \right),$$

$$\log_{\overline{\mathbf{0}}}^K(\mathbf{y}) = \begin{cases} \mathbf{0}, & \mathbf{y} = \overline{\mathbf{0}}, \\ \dfrac{\cosh^{-1}(\beta)}{\sqrt{\beta^2 - 1}} \big(\mathbf{y} - \beta \, \overline{\mathbf{0}}\big), & \text{otherwise}, \end{cases} \qquad \beta := K \, \langle \overline{\mathbf{0}}, \mathbf{y} \rangle_{\mathcal{L}}.$$

with the convention $\mathbf{z}/\|\mathbf{z}\| = \mathbf{0}$ when $\mathbf{z} = \mathbf{0}$.

Parallel transport Transporting $\mathbf{v} \in \mathcal{T}_{\mathbf{x}} \mathbb{L}_K^n$ along the geodesic from $\mathbf{x}$ to $\mathbf{y}$ yields

$$\mathrm{PT}_{\mathbf{x} \to \mathbf{y}}^K(\mathbf{v}) = \mathbf{v} - \frac{\langle \log_{\mathbf{x}}^K(\mathbf{y}), \, \mathbf{v} \rangle_{\mathcal{L}}}{d_{\mathcal{L}}(\mathbf{x}, \mathbf{y})} \Big(\log_{\mathbf{x}}^K(\mathbf{y}) + \log_{\mathbf{y}}^K(\mathbf{x}) \Big) = \mathbf{v} + \frac{\langle \mathbf{y}, \mathbf{v} \rangle_{\mathcal{L}}}{\frac{1}{-K} - \langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}}} \, (\mathbf{x} + \mathbf{y}).$$

Lorentzian centroid and average pooling Given points $\mathbf{x}_1, \dots, \mathbf{x}_m \in \mathbb{L}_K^n$ with nonnegative weights ν_i (not all zero), the weighted Fréchet mean with respect to the squared Lorentzian distance, $\min_{\boldsymbol{\mu} \in \mathbb{L}_K^n} \sum_{i=1}^m \nu_i \, d_{\mathcal{L}}^2(\mathbf{x}_i, \boldsymbol{\mu})$, is obtained in closed form by

$$\boldsymbol{\mu} = \frac{\sum_{i=1}^m \nu_i \, \mathbf{x}_i}{\sqrt{-K} \left| \| \sum_{i=1}^m \nu_i \, \mathbf{x}_i \|_{\mathcal{L}} \right|}.$$

In neural architectures, an "average pooling" over a hyperbolic receptive field can be implemented by taking this Lorentzian centroid of the features in the field.

Lorentz transformations A matrix $\mathbf{A} \in \mathbb{R}^{(n+1)\times(n+1)}$ is a Lorentz transformation if it preserves the Minkowski product: $\langle \mathbf{Ax}, \mathbf{Ay} \rangle_{\mathcal{L}} = \langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}}$ for all $\mathbf{x}, \mathbf{y}$. Such matrices form the Lorentz group $\mathbf{O}(1, n)$ (equivalently, $\mathbf{A}^\top \eta \, \mathbf{A} = \eta$ for the Minkowski metric η). Restricting to transformations that map the upper sheet to itself yields the time-orientation–preserving subgroup

$$\mathbf{O}^+(1, n) = \{ \mathbf{A} \in \mathbf{O}(1, n) \ : \ (\mathbf{Ax})_0 > 0 \text{ for all } \mathbf{x} \in \mathbb{L}_K^n \},$$

which acts by isometries on $\mathbb{L}_K^n$.

Every $\mathbf{A} \in \mathbf{O}^+(1, n)$ admits a polar decomposition into a Lorentz rotation and a Lorentz boost, $\mathbf{A} = \mathbf{RB}$. The rotation fixes the time axis and rotates spatial coordinates:

$$\mathbf{R} = \begin{bmatrix} 1 & \mathbf{0}^\top \\ \mathbf{0} & \tilde{\mathbf{R}} \end{bmatrix}, \qquad \tilde{\mathbf{R}} \in \mathbf{SO}(n).$$

A boost with velocity vector $\mathbf{v} \in \mathbb{R}^n$ ($\|\mathbf{v}\| < 1$) has the block form

$$\mathbf{B} = \begin{bmatrix} \gamma & -\gamma \, \mathbf{v}^\top \\ -\gamma \, \mathbf{v} & \mathbf{I}_n + \dfrac{\gamma^2}{1+\gamma} \, \mathbf{v}\mathbf{v}^\top \end{bmatrix}, \qquad \gamma = \frac{1}{\sqrt{1 - \|\mathbf{v}\|^2}}.$$

(Equivalently, the spatial block can be written $\mathbf{I}_n + \frac{\gamma - 1}{\|\mathbf{v}\|^2} \mathbf{v}\mathbf{v}^\top$ when $\mathbf{v} \neq \mathbf{0}$.)

C Gyrovector structures on the Lorentz model

C.1 Gyrogroups

We recall the algebraic notion of a gyrogroup, which extends the concept of a group to settings where associativity is relaxed and corrected by gyrations (Ungar, 2008; 2014).

Definition 1 (Gyrogroup). *Let G be a nonempty set endowed with a binary operation $\oplus \colon G \times G \to G$. The pair $(G, \oplus)$ is a* gyrogroup *if, for all $a, b, c \in G$, the following axioms hold:*

(G1) There exists an element $e \in G$ such that $e \oplus a = a$ (left identity).
(G2) For each $a \in G$ there exists $\ominus a \in G$ such that $\ominus a \oplus a = e$ (left inverse).
(G3) There exists a map $\mathrm{gyr}[a, b] \colon G \to G$ (the gyration generated by a, b) such that

$$a \oplus (b \oplus c) = (a \oplus b) \oplus \mathrm{gyr}[a, b](c) \quad \textit{(left gyroassociative law).}$$

(G4) $\mathrm{gyr}[a, b] = \mathrm{gyr}[a \oplus b, b]$ (left reduction law).

If in addition

$$a \oplus b = \mathrm{gyr}[a, b](b \oplus a), \qquad \forall \, a, b \in G,$$

then $(G, \oplus)$ is called gyrocommutative.

Gyrogroups generalize groups: when all gyrations are the identity map, $(G, \oplus)$ reduces to a usual group. The lack of strict associativity is compensated by the gyration operators, which encode curvature induced nonlinearity.

C.2 From gyrogroups to gyrovector spaces

To model both addition and scalar multiplication in curved geometries, gyrogroups can be enriched to gyrovector spaces (Ungar, 2008; Nguyen et al., 2022).

Definition 2 (Gyrovector space). *Let $(G, \oplus)$ be a gyrocommutative gyrogroup and let $\odot \colon \mathbb{R} \times G \to G$ be a map called scalar multiplication. The triple $(G, \oplus, \odot)$ is a* gyrovector space *if, for all $a, b, c \in G$ and $s, t \in \mathbb{R}$,*

(V1) $1 \odot a = a$, $0 \odot a = e$, $t \odot e = e$, and $(-1) \odot a = \ominus a$.

(V2) $(s+t)\odot a = s\odot a \oplus t\odot a$.
(V3) $(st)\odot a = s\odot(t\odot a)$.
(V4) $\mathrm{gyr}[a,b](t\odot c) = t\odot \mathrm{gyr}[a,b](c)$.
(V5) $\mathrm{gyr}[s\odot a, t\odot a]$ *is the identity map on* G.

These axioms mirror the familiar properties of vector spaces, with gyrations accounting for the deviation from linearity. In particular, (V2) and (V3) play the role of distributivity and associativity for scalar multiplication, while (V4)–(V5) guarantee a consistent interaction between gyrations and scaling.

C.3 Gyrostructures induced by Riemannian geometry

The above algebraic objects can be constructed on a Riemannian manifold from the exponential and logarithmic maps at a distinguished origin. Following Nguyen et al. (2022); Nguyen & Yang (2023); Chen et al. (2025a), let $(\mathcal{M}, g)$ be a complete Riemannian manifold with identity element $E \in \mathcal{M}$. Denote by Exp_x and Log_x the Riemannian exponential and logarithmic maps at $x \in \mathcal{M}$, and by $\mathrm{PT}_{x\to y}$ the parallel transport from $T_x\mathcal{M}$ to $T_y\mathcal{M}$. For $P, Q, R \in \mathcal{M}$ and $t \in \mathbb{R}$, define

$$P \oplus Q = \mathrm{Exp}_P\big(\mathrm{PT}_{E\to P}(\mathrm{Log}_E\, Q)\big), \tag{12}$$

$$t \odot P = \mathrm{Exp}_E\big(t\ \mathrm{Log}_E\, P\big), \tag{13}$$

$$\ominus P = (-1)\odot P = \mathrm{Exp}_E\big(-\mathrm{Log}_E\, P\big), \tag{14}$$

$$\mathrm{gyr}[P,Q]R = (\ominus(P\oplus Q))\oplus\big(P\oplus(Q\oplus R)\big). \tag{15}$$

The induced gyro inner product, norm, and distance are

$$\langle P, Q\rangle_{\mathrm{gr}} = \big\langle \mathrm{Log}_E\, P, \mathrm{Log}_E\, Q\big\rangle_{T_E\mathcal{M}}, \tag{16}$$

$$\|P\|_{\mathrm{gr}} = \sqrt{\langle P, P\rangle_{\mathrm{gr}}}, \tag{17}$$

$$d_{\mathrm{gr}}(P, Q) = \big\|\ominus P \oplus Q\big\|_{\mathrm{gr}}. \tag{18}$$

Under mild regularity assumptions, $(\mathcal{M}, \oplus, \odot)$ forms a gyrovector space and the gyrodistance d_{gr} coincides with the geodesic distance on a wide class of manifolds, including constant curvature spaces (Nguyen et al., 2022; Chen et al., 2025a). In Euclidean space, these constructions reduce exactly to standard vector addition, scalar multiplication, and the Euclidean metric.

C.4 Constant curvature model and Möbius operations

For hyperbolic geometry it is convenient to introduce the constant curvature model

$$\mathcal{M}_K = \begin{cases} \mathbb{P}^n_K, & K < 0, \\ \mathbb{R}^n, & K = 0, \end{cases}$$

where $\mathbb{P}^n_K$ is the Poincaré ball of curvature $K < 0$ and radius $1/\sqrt{-K}$. On $\mathbb{P}^n_K$ the gyrostructures in Eqs. 12 and 13 admit closed form expressions known as Möbius addition and Möbius scalar multiplication (Ungar, 2008; Ganea et al., 2018; Skopek et al., 2020).

Let $x, y \in \mathbb{P}^n_K$ and set $c = -K > 0$. The Möbius addition is

$$x \oplus_K y = \frac{\big(1 + 2c\langle x, y\rangle + c\|y\|^2\big)x + \big(1 - c\|x\|^2\big)y}{1 + 2c\langle x, y\rangle + c^2\|x\|^2\|y\|^2}, \tag{19}$$

and the Möbius scalar multiplication is

$$t \otimes_K x = \begin{cases} \dfrac{1}{\sqrt{c}}\ \tanh\big(t\ \mathrm{artanh}(\sqrt{c}\,\|x\|)\big)\ \dfrac{x}{\|x\|}, & x \neq 0, \\ 0, & x = 0. \end{cases} \tag{20}$$

Equipped with $\oplus_K$ and $\otimes_K$, the ball $\mathbb{P}^n_K$ is a real gyrovector space whose gyrodistance coincides with the hyperbolic geodesic distance.

C.5 Induced gyrovector operations on the Lorentz model

The Lorentz model $\mathbb{L}^n_K$ is isometric to the Poincaré ball $\mathbb{P}^n_K$ via a standard mapping. Let $K < 0$ and denote $r = 1/\sqrt{-K}$. For $x = (x_0, x_s) \in \mathbb{L}^n_K$ with $x_0 > 0$ and $-x_0^2 + \|x_s\|^2 = 1/K$, define

$$\psi\colon \mathbb{L}^n_K \to \mathbb{P}^n_K, \qquad \psi(x) = \frac{x_s}{x_0 + r}, \tag{21}$$

$$\psi^{-1}\colon \mathbb{P}^n_K \to \mathbb{L}^n_K, \qquad \psi^{-1}(u) = \left(\frac{1 + c\|u\|^2}{1 - c\|u\|^2}\, r,\ \frac{2u}{1 - c\|u\|^2}\right), \tag{22}$$

where $c = -K > 0$. The map ψ is a Riemannian isometry between $(\mathbb{L}_K^n, g_K)$ and $(\mathbb{P}_K^n, g_K)$ (Ratcliffe, 2006; Skopek et al., 2020).

We transfer the gyrovector structure from the ball to the hyperboloid by conjugation with ψ. For $x, y \in \mathbb{L}_K^n$ and $t \in \mathbb{R}$, define

$$x \oplus_K^{\mathcal{L}} y = \psi^{-1}\big(\psi(x) \oplus_K \psi(y)\big), \tag{23}$$

$$t \otimes_K^{\mathcal{L}} x = \psi^{-1}\big(t \otimes_K \psi(x)\big). \tag{24}$$

Substituting the explicit expressions Eqs. 19 and 20 and the coordinate maps Eqs. 21 and 22 yields closed form formulas for $x \oplus_K^{\mathcal{L}} y$ and $t \otimes_K^{\mathcal{L}} x$ in Lorentz coordinates. By construction, $(\mathbb{L}_K^n, \oplus_K^{\mathcal{L}}, \otimes_K^{\mathcal{L}})$ is a gyrovector space that is isomorphic to $(\mathbb{P}_K^n, \oplus_K, \otimes_K)$. In particular, the gyrodistance induced by Eq. 23 agrees with the hyperbolic geodesic distance on $\mathbb{L}_K^n$, and the gyroaddition and gyro scalar multiplication act as hyperbolic analogues of Euclidean vector addition and Euclidean scaling.

D Discussion about Intrinsic

In our paper, "intrinsic Lorentz" refers to using only operations that are well-defined on the Lorentz model itself, rather than on its ambient Minkowski space. We define a layer ($F : \mathbb{L}_K^n \to \mathbb{L}_K^m$) intrinsic if

1. its input, output, and all intermediate states lie on some $\mathbb{L}_K^d$ (they always satisfy $\langle z, z\rangle_L = -1/K,\ z_0 > 0$).
2. it is expressed entirely in terms of the Lorentzian geometry: $\langle\cdot,\cdot\rangle_L$, the induced distance d_L, and operators derived from them (exp/log maps, parallel transport, gyroaddition/gyroscaling, Lorentzian centroids, etc.), without ever using arbitrary Euclidean linear maps on Lorentz vectors in the ambient space.

Under this definition, the previous Lorentz fully connected layer (LFC) used in HCNN is not intrinsic. Its update has the form

$$\mathbf{y} = \left[\sqrt{|\phi(W\mathbf{x}, \mathbf{v})|^2 - 1/K}, \phi(W\mathbf{x}, \mathbf{v})\right],$$

where $\mathbf{x} \in \mathcal{L}_K^n$, $W\mathbf{x}$ is a standard matrix–vector product in $\mathbb{R}^n$ and the operation

$$\phi(\mathbf{W}\boldsymbol{x}, \boldsymbol{v}) = \lambda\sigma(\boldsymbol{v}^T\boldsymbol{x} + \boldsymbol{b}')\frac{\mathbf{W}\psi(\boldsymbol{x}) + \boldsymbol{b}}{||\mathbf{W}\psi(\boldsymbol{x}) + \boldsymbol{b}||}. \tag{25}$$

This $W\mathbf{x}$ is defined using the ambient linear structure of the Minkowski space, not any operation on the Lorentz manifold itself. Because the core transformation is an ambient Minkowski multiplication rather than a Lorentzian/geodesic operation, this layer is only partially intrinsic.

By contrast, our PLFC is constructed entirely from Lorentz-geometry primitives that have closed-form definitions on $\mathcal{L}_K^n$. Each logit is the signed Lorentzian distance from the input point to a learned Lorentz hyperplane, and the output $\mathbf{y}$ is then recovered in closed form as the unique point on $\mathcal{L}_K^m$ whose signed distances to a set of coordinate hyperplanes equal these logits. All steps (hyperplane parameterization, point-to-hyperplane distance, reconstruction of $\mathbf{y}$) are expressed only via the Lorentzian inner product and distance; no ambient Euclidean affine map $W\mathbf{x} + b$ is ever applied. In this sense, PLFC is an intrinsic Lorentz FC layer.

E Proofs

E.1 Proof of Theorem 1

Proof. We work in the hyperboloid model $\mathbb{L}_K^m = \{\boldsymbol{x} = [x_t, \boldsymbol{x}_s] \in \mathbb{R}^{1+m} : \langle \boldsymbol{x}, \boldsymbol{x}\rangle_{\mathcal{L}} = 1/K,\ x_t > 0\}$ with $K < 0$ and Minkowski bilinear form $\langle [x_t, \boldsymbol{x}_s], [y_t, \boldsymbol{y}_s]\rangle_{\mathcal{L}} = -x_t y_t + \langle \boldsymbol{x}_s, \boldsymbol{y}_s\rangle$. Let $\overline{\mathbf{0}} = [(-K)^{-1/2}, \mathbf{0}]$ be the basepoint. Denote by $\mathbf{e}^{(k)} \in \mathbb{R}^m$ the k-th Euclidean basis vector in the spatial block and set $\boldsymbol{e}^{(k)} = [0, \mathbf{e}^{(k)}]$.

Lorentz coordinate hyperplanes. The k-th spatial axis is the geodesic through $\overline{\mathbf{0}}$ in the direction $\mathbf{e}^{(k)}$. The Lorentz *coordinate hyperplane* through $\overline{\mathbf{0}}$, orthogonal to this axis, is given by

Definition 3. *(Lorentz hyperplane containing $\overline{\mathbf{0}}$ and orthogonal to the k-th axis)*

$$\bar{H}^K_{\boldsymbol{e}^{(k)},0} = \big\{\boldsymbol{x} = [x_t, \boldsymbol{x}_s]^\top \in \mathbb{L}_K^m \ \big|\ \langle \boldsymbol{e}^{(k)}, \boldsymbol{x}\rangle_{\mathcal{L}} = \langle \mathbf{e}^{(k)}, \boldsymbol{x}_s\rangle = x_{s,k} = 0\big\}. \tag{26}$$

This is the special case of the Lorentz hyperplane $\tilde{H}_{\boldsymbol{z},a}$ in Eq. 4 with $a = 0$ and $\boldsymbol{z} = \mathbf{e}^{(k)}$ (hence $\|\boldsymbol{z}\|_2 = 1$), for which $\cosh\big(\sqrt{-K}a\big) = 1$ and $\sinh\big(\sqrt{-K}a\big) = 0$, giving $\langle \boldsymbol{z}, \boldsymbol{x}_s\rangle = x_{s,k} = 0$.

With Definition 3, the preparation for constructing $\boldsymbol{y}$ in Eq. 8 is complete.

Derivation of $\boldsymbol{y}$. Let $\boldsymbol{x} \in \mathbb{L}_K^n$ and $\boldsymbol{y} = [y_t, \boldsymbol{y}_s]^\top \in \mathbb{L}_K^m$ be the input and output of the PLFC layer, respectively. As in Eq. 8a, for $k = 1, \dots, m$ we define the scores $v_k(\boldsymbol{x}) = v_{\boldsymbol{z}_k, a_k}(\boldsymbol{x})$ via the Lorentz MLR logits in Eq. 6.

To endow $\boldsymbol{y}$ with the desired property—that the *signed* hyperbolic distance from $\boldsymbol{y}$ to the k-th coordinate hyperplane equals $v_k(\boldsymbol{x})$—we impose the simultaneous system

$$d_{\mathcal{L}}^{\pm}\big(\boldsymbol{y}, \bar{H}^K_{\boldsymbol{e}^{(k)},0}\big) = v_k(\boldsymbol{x}), \qquad k = 1, \dots, m. \tag{27}$$

Using Eqs. 4 and 5 with $a = 0$ and $\boldsymbol{z} = \mathbf{e}^{(k)}$ (so that the denominator equals 1), the unsigned point–to–hyperplane distance specializes to

$$d_{\mathcal{L}}\big(\boldsymbol{y}, \bar{H}^K_{\boldsymbol{e}^{(k)},0}\big) = \frac{1}{\sqrt{-K}} \left|\sinh^{-1}\big(\sqrt{-K}\, y_{s,k}\big)\right|.$$

Orienting $\bar{H}^K_{\boldsymbol{e}^{(k)},0}$ by the unit normal $\boldsymbol{e}^{(k)}$ and recalling the sign convention used in Eq. 6, the *signed* distance is therefore

$$d_{\mathcal{L}}^{\pm}\big(\boldsymbol{y}, \bar{H}^K_{\boldsymbol{e}^{(k)},0}\big) = \frac{1}{\sqrt{-K}} \sinh^{-1}\big(\sqrt{-K}\, y_{s,k}\big). \tag{28}$$

Substituting Eq. 28 into Eq. 27 yields, for each k,

$$\frac{1}{\sqrt{-K}} \sinh^{-1}\big(\sqrt{-K}\, y_{s,k}\big) = v_k(\boldsymbol{x}), \tag{29}$$

and hence

$$y_{s,k} = \frac{1}{\sqrt{-K}} \sinh\big(\sqrt{-K}\, v_k(\boldsymbol{x})\big), \qquad k = 1, \dots, m, \tag{30}$$

which is exactly Eq. 8b. Collecting these coordinates gives $\boldsymbol{y}_s = (y_{s,1}, \dots, y_{s,m})^\top$.

Finally, the time coordinate y_t is fixed by the hyperboloid constraint $\langle \boldsymbol{y}, \boldsymbol{y} \rangle_{\mathcal{L}} = 1/K$, i.e.,

$$-y_t^2 + \|\boldsymbol{y}_s\|_2^2 = \frac{1}{K} \quad \Longrightarrow \quad y_t = \sqrt{(-K)^{-1} + \|\boldsymbol{y}_s\|_2^2}, \tag{31}$$

with the positive root chosen to remain on the top sheet, which is Eq. 8c. Thus Eq. 8 follows.

Confirmation of the existence of $\boldsymbol{y}$. For any real scores $v_k(\boldsymbol{x})$, Eq. 30 yields real $y_{s,k}$ because $\sinh$ is entire. Since $-K > 0$, we have $(-K)^{-1} + \|\boldsymbol{y}_s\|_2^2 > 0$, so y_t in Eq. 31 is real and strictly positive. Therefore $\boldsymbol{y} = [y_t, \boldsymbol{y}_s] \in \mathbb{L}_K^m$ always exists and lies on the correct sheet. Moreover, Eq. 29 guarantees that the signed distance from $\boldsymbol{y}$ to each coordinate hyperplane $\bar{H}^K_{\boldsymbol{e}^{(k)},0}$ is exactly $v_k(\boldsymbol{x})$, as required.

Flat-space limit. As $K \to 0^-$, we have $\sinh\big(\sqrt{-K}\, v\big) = \sqrt{-K}\, v + O(K^{3/2})$, hence $y_{s,k} \to v_k(\boldsymbol{x})$ from Eq. 30. Using Eq. 6 and the expansion $\cosh\big(\sqrt{-K}a\big) = 1 + O(K)$, $\sinh\big(\sqrt{-K}a\big) = \sqrt{-K}\, a + O(K^{3/2})$, one obtains $v_k(\boldsymbol{x}) \to \langle \boldsymbol{z}_k, \boldsymbol{x}_s \rangle - a_k$, so the spatial part reduces to the Euclidean affine map with row vectors $A_k = \boldsymbol{z}_k^\top$ and bias $b_k = a_k$, as stated below Theorem 1. □

E.2 Proof of Theorem 2

Theorem 2 (Margin preservation and contraction of PLFC and LFC)**.** *Fix a curvature $K < 0$. For any input x, let the penultimate layer produce*

$$u(x) = (u_1(x), \dots, u_m(x)) \in \mathbb{R}^m,$$

and let c denote the true class. Define the pre-logit margin

$$\Delta(x) := u_c(x) - \max_{j \neq c} u_j(x).$$

Consider two Lorentz output-layer designs that use signed geodesic distances from the output point $y(x)$ to the coordinate hyperplanes as logits:

- PLFC head (intrinsic). *The spatial coordinates are*

$$y_{s,k}^{\mathrm{PLFC}}(x) = \frac{1}{\sqrt{-K}} \sinh\big(\sqrt{-K}\, u_k(x)\big),$$

and the signed Lorentzian distance to the k-th coordinate hyperplane is

$$d_k^{\mathrm{PLFC}}(x) = \frac{1}{\sqrt{-K}} \operatorname{asinh}\big(\sqrt{-K}\, y_{s,k}^{\mathrm{PLFC}}(x)\big).$$

- LFC head (extrinsic / linear). *The spatial coordinates are taken directly as*

$$y_{s,k}^{\mathrm{LFC}}(x) = u_k(x),$$

and the signed Lorentzian distance to the same hyperplane is

$$d_k^{\mathrm{LFC}}(x) = \frac{1}{\sqrt{-K}}\,\mathrm{asinh}\big(\sqrt{-K}\,u_k(x)\big).$$

Define the distance-based margins

$$\Delta^{\mathrm{PLFC}}(x) := d_c^{\mathrm{PLFC}}(x) - \max_{j\neq c} d_j^{\mathrm{PLFC}}(x), \qquad \Delta^{\mathrm{LFC}}(x) := d_c^{\mathrm{LFC}}(x) - \max_{j\neq c} d_j^{\mathrm{LFC}}(x).$$

Then, for every sample x:

1. ***(Margin preservation of PLFC)***

$$\Delta^{\mathrm{PLFC}}(x) = \Delta(x).$$

2. ***(Margin contraction of LFC)*** *The LFC head preserves the sign of the margin and contracts its magnitude:*

$$\mathrm{sign}\,\big(\Delta^{\mathrm{LFC}}(x)\big) = \mathrm{sign}\,\big(\Delta(x)\big), \qquad \big|\Delta^{\mathrm{LFC}}(x)\big| \le \big|\Delta(x)\big|,$$

with strict inequality $\big|\Delta^{\mathrm{LFC}}(x)\big| < \big|\Delta(x)\big|$ whenever $\Delta(x) \neq 0$.

Proof. Define

$$h(t) := \frac{1}{\sqrt{-K}}\,\mathrm{asinh}\big(\sqrt{-K}\,t\big), \qquad K < 0.$$

By the point–to–hyperplane distance formula in the Lorentz model, for any Lorentz point $y = (y_0, y_s)$ the signed distance to the k-th coordinate hyperplane is exactly $h(y_{s,k})$.

(1) PLFC preserves the margin. For the PLFC head we have

$$y_{s,k}^{\mathrm{PLFC}}(x) = \frac{1}{\sqrt{-K}}\,\sinh\big(\sqrt{-K}\,u_k(x)\big),$$

and thus

$$d_k^{\mathrm{PLFC}}(x) = h\big(y_{s,k}^{\mathrm{PLFC}}(x)\big) = \frac{1}{\sqrt{-K}}\,\mathrm{asinh}\Big(\sqrt{-K}\cdot\frac{1}{\sqrt{-K}}\,\sinh\big(\sqrt{-K}\,u_k(x)\big)\Big) = \frac{1}{\sqrt{-K}}\,\mathrm{asinh}\big(\,\sinh\Big(\sqrt{-K}\,u_k(x)\Big)\big).$$

Since $\sinh : \mathbb{R} \to \mathbb{R}$ is a bijection with inverse asinh, we have $\mathrm{asinh}(\sinh z) = z$ for all $z \in \mathbb{R}$, hence

$$d_k^{\mathrm{PLFC}}(x) = u_k(x).$$

Consequently,

$$\Delta^{\mathrm{PLFC}}(x) = d_c^{\mathrm{PLFC}}(x) - \max_{j\neq c} d_j^{\mathrm{PLFC}}(x) = u_c(x) - \max_{j\neq c} u_j(x) = \Delta(x),$$

which proves (1).

(2) LFC contracts the margin. For the LFC head we have $y_{s,k}^{\mathrm{LFC}}(x) = u_k(x)$, hence

$$d_k^{\mathrm{LFC}}(x) = h\big(u_k(x)\big).$$

We first record two basic properties of h. Differentiating,

$$h'(t) = \frac{\mathrm{d}}{\mathrm{d}t}\Big[\frac{1}{\sqrt{-K}}\,\mathrm{asinh}\big(\sqrt{-K}\,t\big)\Big] = \frac{1}{\sqrt{-K}}\cdot\frac{\sqrt{-K}}{\sqrt{1+(-K)t^2}} = \frac{1}{\sqrt{1+(-K)t^2}},$$

so $h'(t) > 0$ for all t and $h'(t) \le 1$ with $h'(t) = 1$ if and only if $t = 0$. Therefore, h is strictly increasing and 1-Lipschitz.

Let $j^\star$ be any index of a maximizer of the competing pre-logits:

$$j^\star \in \arg\max_{j\neq c} u_j(x).$$

Since h is strictly increasing, the same index maximizes the distance-based logits:

$$\max_{j\neq c} d_j^{\mathrm{LFC}}(x) = \max_{j\neq c} h(u_j(x)) = h\big(u_{j^\star}(x)\big).$$

Hence

$$\Delta(x) = u_c(x) - u_{j^\star}(x), \qquad \Delta^{\mathrm{LFC}}(x) = h\big(u_c(x)\big) - h\big(u_{j^\star}(x)\big).$$

If $\Delta(x) = 0$, then $u_c(x) = u_{j^\star}(x)$ and consequently $\Delta^{\mathrm{LFC}}(x) = 0$, so the conclusion holds trivially. Suppose now $\Delta(x) \neq 0$ and, without loss of generality, set

$$a := u_c(x), \qquad b := u_{j^\star}(x), \qquad a \neq b.$$

Assume $a > b$; the case $a < b$ is analogous by symmetry. Using the fundamental theorem of calculus,

$$h(a) - h(b) = \int_b^a h'(t)\,\mathrm{d}t.$$

Because $h'(t) > 0$ for all t, we have $h(a) - h(b) > 0$, so $\operatorname{sign}(h(a) - h(b)) = \operatorname{sign}(a - b)$. Moreover, since $h'(t) \leq 1$ everywhere and $h'(t) < 1$ for all $t \neq 0$, the integrand is strictly less than 1 on a subset of $[b, a]$ with positive measure whenever $a \neq b$. Thus

$$0 < h(a) - h(b) = \int_b^a h'(t)\,\mathrm{d}t < \int_b^a 1\,\mathrm{d}t = a - b.$$

Taking absolute values yields

$$\big|h(a) - h(b)\big| < |a - b|.$$

Substituting back $a = u_c(x)$ and $b = u_{j^\star}(x)$, we obtain

$$\operatorname{sign}\big(\Delta^{\mathrm{LFC}}(x)\big) = \operatorname{sign}\big(\Delta(x)\big), \qquad \big|\Delta^{\mathrm{LFC}}(x)\big| = \big|h(a) - h(b)\big| < |a - b| = \big|\Delta(x)\big|.$$

Combining with the $\Delta(x) = 0$ case, this gives

$$\big|\Delta^{\mathrm{LFC}}(x)\big| \leq \big|\Delta(x)\big|,$$

with strict inequality whenever $\Delta(x) \neq 0$, which proves (2). □

F Implementation Details

F.1 Datasets

All these datasets exhibit hierarchical class relations and high hyperbolicity (low δ_{rel}) as shown in Table 5, making the use of hyperbolic models well-motivated.

Image Datasets For image classification, we adopt the standard benchmarks CIFAR-10 and CIFAR-100(Krizhevsky et al., 2009). CIFAR-10 and CIFAR-100 each contain 60,000 32×32 color images drawn from 10 and 100 classes, respectively. Following the PyTorch setup and (Bdeir et al., 2024), we use 50,000 images for training and 10,000 for testing. CIFAR-10 and CIFAR-100 are standard proxies for visual object recognition whose categories naturally admit semantic hierarchies (e.g., animal → mammal → dog → specific breed). In CIFAR-100, this is made explicit by grouping the 100 fine-grained classes into 20 coarse superclasses in the original dataset design. This hierarchical structure has been extensively verified in prior hyperbolic vision work Bdeir et al. (2024); Nguyen et al. (2025) on CIFAR-10/100

Gene Datasets For genomic sequence classification, we evaluate on the Transposable Elements Benchmark (TEB) (Khan et al., 2025) and the Genome Understanding Evaluation (GUE) (Zhou et al., 2023) suite. TEB is a curated, multi-species collection of binary classification tasks spanning seven transposable-element families across retrotransposons, DNA transposons, and pseudogenes; we follow the authors' released preprocessing and data partitions. GUE aggregates 28 datasets covering seven biologically meaningful tasks—including transcription-factor binding, epigenetic mark prediction, promoter and splice-site detection—with sequences ranging roughly from 70 to 1000 base pairs originating from yeast, mouse, human, and viral genomes. Unless otherwise noted, we adopt the official train/validation/test splits and report the Matthews correlation coefficient (MCC) as our primary metric. The TEB and GUE suites are constructed directly from natural genomic sequences and inherit the biological hierarchies of their domains. TEB tasks span multiple transposable-element families across retrotransposons, DNA transposons, and pseudogenes, which themselves sit in a multi-level taxonomic hierarchy (orders → superfamilies → families). GUE aggregates datasets for transcription-factor binding, promoter and core-promoter detection, splice-site prediction, and COVID-variant classification across several species. These tasks are all manifestations of hierarchical regulatory structure (e.g., motifs → modules → promoters → gene expression).

Table 5: Hyperbolicity values of the datasets used in our experiments (δ_{rel}).

Benchmark	Task	Dataset	δ_{rel}
CIFAR	Image classification	CIFAR-10	0.26
		CIFAR-100	0.23
TEB	Pseudogenes	processed	0.19
	Pseudogenes	unprocessed	0.16
GUE	Covid variant classification	covid	0.42
	Core Promoter detection	all	0.23
		notata	0.21
		tata	0.14
	Promoter detection	all	0.26
		notata	0.26
		tata	0.14

F.2 Settings

Table 6 summarizes the hyperparameters used to train the model. We additionally note a dataset-specific choice regarding normalization statistics. On *CIFAR-10/100* we enable running statistics: we maintain per-channel running statistics by updating the Lorentzian centroid (mean) and dispersion (variance) with a momentum term and, at test time, substitute these running estimates for the batch statistics. In contrast, on the *TEB* and *GUE* genomic suites, we disable running statistics entirely, because enabling them consistently led to a collapse of MCC after a few dozen epochs. For these genomic datasets, we therefore compute statistics on-the-fly from each evaluation batch (i.e., no moving averages are used at test time). Compared with natural images, the genomic tasks exhibit stronger distributional non-stationarity (heterogeneous sequence lengths and tasks) and higher batch-to-batch variability. Under these conditions, momentum-based running estimates accumulate bias and lag behind the true data distribution; in a Lorentzian normalization layer, a biased centroid and underestimated dispersion can over- or under-normalize timelike features, shrinking margins and destabilizing optimization.

Table 6: Summary of hyperparameters used in different datasets.

Hyperparameter	CIFAR-10&100	TEB	GUE
Epochs	200	150	150
Batch size	128	256	256
Learning rate (LR)	1e-1	8e-4	9e-4
Drop LR epochs	60, 120, 160	100,130	100,130
Drop LR gamma	0.2	0.1	0.1
Weight decay	5e-4	6e-3	5e-3
Optimizer	(Riemannian)SGD	(Riemannian)Adam	(Riemannian)Adam
Floating point precision	32 bit	32 bit	32 bit
GPU type	RTX A100	RTX A100	RTX A100
Num. GPUs	1	1	1
Hyperbolic curvature K	-1	-1	-1
Dropout rate	0.05	0.05	0.05

G More Ablation Studies

Table 7 above expands the ablation to the number of Fréchet mean iterations used by each normalizer while keeping all other components fixed. Across all GUE datasets and both tasks, *PLFC+GyroLBN* achieves the best accuracy under every iteration budget, and the relative ordering of methods is unchanged as the budget increases. Moving from 1 to 2 and 5 iterations yields modest but consistent gains, whereas 10 steps and the fixed-point solution (cells with gray background, denoted by ∞) offer only marginal improvements at a higher computational cost. These trends indicate that the advantage of GyroLBN stems from the normalization rule itself rather than from merely computing a tighter Fréchet mean. In practice, allocating a small budget of two to

five iterations recovers nearly all of the attainable accuracy while preserving efficiency, making *PLFC+GyroLBN* the most effective and economical choice in our setting.

Table 7: Ablation study on the number of Fréchet mean iterations. The symbol ∞ indicates that iterations are performed until convergence, which is the setting used in our main ablation experiments.

Benchmark	Task	Dataset	Iteration	Model			
				LFC LBN	LFC GyroLBN	PLFC GyroBN	PLFC GyroLBN
GUE	Core Promoter Detection	tata	1	78.26±2.85	81.33±3.19	80.09±1.90	**83.90±0.53**
			2			81.38±2.74	
			5			79.47±2.21	
			10			81.27±1.75	
			∞			80.89±3.11	
		notata	1	66.60±1.07	71.92±0.52	71.99±0.68	**72.59**±0.69
			2			71.06±0.49	
			5			71.65±0.79	
			10			71.12±1.75	
			∞			72.22±1.44	
		all	1	66.47±0.74	69.74±1.3	70.47±0.85	**70.89±0.43**
			2			70.42±0.49	
			5			70.75±0.45	
			10			69.81±0.67	
			∞			70.14±0.45	
	Promoter Detection	tata	1	78.58±3.39	80.46±0.99	81.71±1.80	**83.26±1.90**
			2			80.19±1.90	
			5			81.69±3.90	
			10			79.79±1.91	
			∞			81.16±1.99	
		notata	1	90.81±0.51	91.88±1.01	92.15±0.76	**92.48±0.35**
			2			91.89±0.71	
			5			92.19±0.41	
			10			92.07±0.63	
			∞			91.67±0.56	
		all	1	88.00±0.39	90.28±1.04	90.45±0.72	**91.34±0.38**
			2			91.01±0.99	
			5			90.80±0.73	
			10			91.18±0.83	
			∞			91.02±0.56	