On the Universal Statistical Consistency of Expansive Hyperbolic Deep Convolutional Neural Networks

Anonymous authors
Paper under double-blind review

Abstract

The emergence of Deep Convolutional Neural Networks (DCNNs) has been a pervasive tool for accomplishing widespread applications in computer vision. Despite its potential capability to capture intricate patterns inside the data, the underlying embedding space remains Euclidean and primarily pursues contractive convolution. Several instances can serve as a precedent for the exacerbating performance of DCNNs. The recent advancement of neural networks in the hyperbolic spaces gained traction, incentivizing the development of convolutional deep neural networks in the hyperbolic space. In this work, we propose Hyperbolic DCNN based on the Poincaré Ball. The work predominantly revolves around analyzing the nature of expansive convolution in the context of the non-Euclidean domain. We further offer extensive theoretical insights about the universal consistency of the expansive convolution in the hyperbolic space. Several simulations were performed not only on the synthetic datasets but also on some real-world datasets. The experimental results reveal that the hyperbolic convolutional architecture outperforms the Euclidean ones by a commendable margin.

1 Introduction

The ubiquitous utility of Deep Convolutional Neural Networks (DCNNs) LeCun et al. (1998) dominated the arena of Computer Vision Yang & Li (2017); Fu et al. (2019); Sonata et al. (2021) over the past decade. This profound success can be attributed to the effectiveness of the CNNs in approximating the broader class of continuous functions Lin et al. (2022b). The prevalent convolutional neural architectures He et al. (2016); Simonyan (2014) predominantly operate in the Euclidean feature space. The choice of Euclidean space is mostly for implementable closed-form vector space and inner product structures, and their availability in tabular forms. We are focused on DCNN architectures which evolve around $1-$ dimensional convolution based on one input channel and ReLU (Rectified Linear Unit, $r(x) := \max(0, x)$ for $x \in \mathbb{R}$) activation function given to the computational units (Neurons). For two functions $f, g : \mathbb{R}^n \to \mathbb{R}$, we define their convolution as

$$f \otimes g(z) := \int_{\mathbb{R}^n} f(x)g(z-x)dx,$$

where $z \in \mathbb{R}^n$. In the Ballrete version, given a filter $\mathbf{w} := \{\mathbf{w_i}\}_{\mathbf{i}=-\infty}^{\infty}$, where only finitely many $w_j \neq 0$. We call $\mathbf{w}$ to be a filter of length s if $w_j \neq 0$ only for $0 \leq j \leq s$. For a one dimensional input vector $\mathbf{v} := \{\mathbf{v_1}, \mathbf{v_2}, ..., \mathbf{v_n}\} \in \mathbb{R}^{\mathbf{n}}$, we can define two types of convolution operations, namely *Expansive Convolution* $(\mathbf{w} * \mathbf{v})$ and *Contractive Convolution*$(\mathbf{w} \star \mathbf{v})$, given by the following forms of equations

$$(\mathbf{w} * \mathbf{v})_k := \sum_{i=1}^{n} w_{k-i} v_i, \;\; k = 1, 2, ..., n+s \tag{1}$$

and

$$(\mathbf{w} \star \mathbf{v})_k := \sum_{i=k-s}^{k} w_{k-i} v_i, \;\; k = s+1, s+2, ..., n \tag{2}$$

respectively. Now for a set of L filters $\{\mathbf{w_i}\}_{\mathbf{i=1}}^{\mathbf{L}}$, where L is the depth of our network with L many bias vectors $\{b_i\}_{i=1}^{L}$, we recursively define the output of an intermediate layer given in terms of the output of the previous layer as Zhou (2020)

$$h_i(x) = r\left(\mathbf{w}_i \circ h_{i-1}(x) + b_i\right), \text{ for } i = 1, 2, ..., L,$$

starting with the input as $h_0(x) = x$ and $\circ$ can be either $*$ or $\star$ as defined by equations 1 or 2 respectively. The final one-dimensional output of this network is defined as the scalar product between the output produced by the Lth layer with a trainable vector a_L of compatible length

$$h_o(x) := a_l \cdot h_L(x).$$

Although this form of Euclidean convolution has been proven to be enormously successful in several tasks in computer vision, several precedents Lin et al. (2022a); Djeddal et al. (2021); Long & van Noord (2023) can be put forth where Euclidean feature space seems unproductive, like datasets containing hierarchical structures. Learning embeddings of hierarchical data in Euclidean spaces often falls short in capturing meaningful structural information. To address this limitation, researchers have explored neural network architectures in non-Euclidean spaces, particularly hyperbolic geometry Ganea et al. (2018); Nickel & Kiela (2017b); Bdeir et al. (2023). Hyperbolic Neural Networks (HNNs) Ganea et al. (2018) have emerged as a promising framework, leveraging negatively curved spaces to better represent complex relationships and hierarchical structures. Building on this, Hyperbolic Deep Convolutional Neural Networks (HDCNNs) have shown success in various image-related tasks. However, despite these empirical advances, a rigorous theoretical understanding of hyperbolic convolution remains largely undeveloped.

Motivated by this gap, our work provides a comprehensive statistical analysis of hyperbolic convolution, focusing on the consistency of expansive convolutional operations in hyperbolic space—a topic that has received little attention compared to its Euclidean counterpart. Prior studies, such as Lin et al. (2022b), have established consistent results for Euclidean convolutional networks using bounds on packing numbers and error analysis. Yet, similar theoretical foundations for hyperbolic networks are lacking.

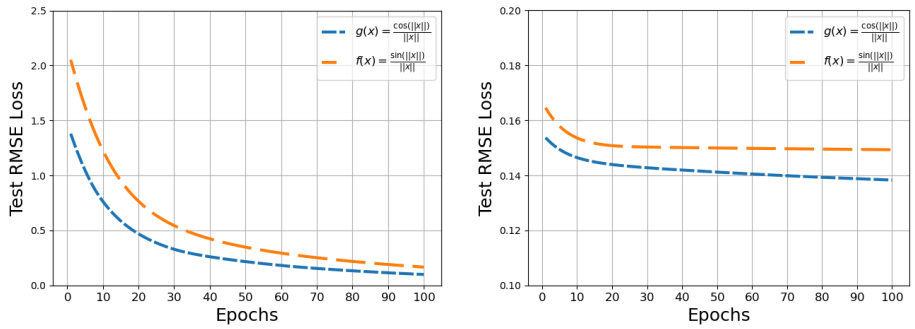
To this end, we introduce a theoretical framework for $1-$D expansive Hyperbolic Deep Convolutional Neural Networks (eHDCNNs), extending the conventional Euclidean DCNNs to the hyperbolic domain via the Poincaré Ball model. This foundation enables the formulation of key statistical properties and paves the way for theoretical consistency analysis. Empirical results on both synthetic and real-world datasets confirm the superiority of hyperbolic representations, with significantly faster convergence and lower error rates compared to Euclidean models, thus validating our theoretical contributions.

Contribution

Our main contributions could be summarized in the following way:

- We provide theoretical insights, including the consistency analysis of the expansive 1-D convolution in hyperbolic space. To the best of our knowledge, this is the first work to present a complete proof in the context of a fully hyperbolic set-up. In doing so, we have also introduced the concept of a fully hyperbolic convolution operation on the Poincaré Ball, which is the generalization of the conventional Euclidean convolution operation on hyperbolic spaces. Additionally, we have defined several statistical terminologies within the hyperbolic framework to derive universal consistency. All necessary proofs and derivations are provided in Section A.

- Our experimental simulations demonstrate that eHDCNN training converges more rapidly than the training of the Euclidean DCNN, which we have already established theoretically. The faster reduction of error rate reaffirms the requirements of the lower number of training iterations for hyperbolic convolutional networks compared to their conventional Euclidean counterparts, establishing the effectiveness of eHDCNN. Details of the experiments and simulations are provided in Section 7.



(a) c = 0 (b) c = 1

Figure 1: Test Root Mean Squared error for $f(x)$ and $g(x)$ plotted using (a) eDCNN architecture curvature 0 (i.e., Euclidean space) and (b) eHDCNN architecture with curvature 1.

2 A Motivating Example

We will demonstrate the efficacy of our proposed hyperbolic expansive convolution over conventional Euclidean expansive convolution through the following simulation.

Experimental Setup

Consider the following functions,

$$f(x) = \frac{\sin(\|x\|_2)}{\|x\|_2}, \qquad g(x) = \frac{\cos(\|x\|_2)}{\|x\|_2},$$

where $f(x)$ and $g(x)$ both are modeled as regression task like $y := h(x) + \epsilon$. Here, h can be replaced with either f or g. The training instances are generated by sampling $\epsilon \sim \mathcal{N}(0, 0.01)$ and $x \sim unif([-1, 1]^5)$. A total of 1000 instances will be generated for both cases where 800 and 200 samples will be respectively used for training and testing. Importantly, the test samples are considered without the Gaussian noise. The filter length is fixed at 8 with the number of layers is 4. Both models are trained for 100 epochs over the training set. The test Root Mean Squared Loss (RMSE) is recorded after the completion of training and presented in Figure 1. Experiments are conducted for two different curvatures $c = 0$ (Euclidean space) and $c = 1$. We considered unit radius Poincaré Ball as the hyperbolic space. Assuming the point set in Ballrete metric space, we employed Gromov Hyperbolicity (GH) Väisälä (2005) to measure the hyperbolicity (δ) of the corresponding data points. The metric offers hyperbolicity of f and g are respectively $\delta_f = 0.45$ and $\delta_g = 0.13$, indicating that g is more hyperbolic comparing to f.

3 Related Works

Hyperbolic Image Embedding and NLP Tasks

Developing a Hyperbolic Neural network for computer vision tasks has been mainly focused on combining Euclidean Encoders and Hyperbolic Embedding. These architectures were demonstrated to be effective in performing various vision tasks, for example, recognition Khrulkov et al. (2020),Guo et al. (2022a), generation Nagano et al. (2019), and image segmentation Atigh et al. (2022). While Hyperbolic Embedding has also been tremendously successful in performing various tasks related to Natural Language Processing Nickel & Kiela (2017a),Nickel & Kiela (2018). These ideas were mainly motivated by the expressive power of the hyperbolic spaces to represent graph or tree-like hierarchies in shallow dimensions with very low distortions. However, deploying Riemannian Optimization algorithms to train this architecture is difficult due to the

inability to extend them for visual data since NLP tasks lack the availability of Ballrete data Sala et al. (2018),Sarkar (2011).

Fully Connected Hyperbolic Neural Network

In 2018 Ganea et al. (2018), and in 2020 Shimizu et al. (2020) independently developed the structure of Hyperbolic Neural Networks on Poincaré Ball by utilizing the gyrovector space structure. They defined the generalized notions of different layers like fully connected, convolutional, or attention layers. Fan et al. (2022), Qu & Zou (2022) tried to develop variations of HNN models like fully Hyperbolic GAN on Lorentz Model space, van Spengler et al. (2023) proposed a fully hyperbolic CNN architecture on Poincaré Ball model. Very recently, Bdeir et al. (2023) presented a fully convolutional neural network on the Lorentz Model to perform complex computer vision tasks, where they generalized fundamental components of CNNs and proposed novel formulations of convolutional layer, batch normalization, and Multinomial Logistic Regression (MLR) classifier. Moreover, hyperbolic graph neural networks can also accomplish recommendation tasks. There are numerous recommender systems such as graph neural collaborative filtering Sun et al. (2021), Yang et al. (2022), social network enhanced network system Wang et al. (2021), knowledge graph enhanced recommender system Chen et al. (2022), and session-based recommender system Guo et al. (2022a), Li et al. (2021).

Batch Normalization in Hyperbolic Neural Networks

Batch Normalization Ioffe (2015) restricts the internal departure of neuron outputs by normalizing the outputs produced by the activations at each layer. This adds stability to the training procedure and speeds up the training phase. Several attempts have been made to transcend the normalization of conventional neural networks in the hyperbolic setup. The general framework of Riemannian Batch Normalization Lou et al. (2020), however, suffers from slower computation and iterative update of the Frechét centroid, which does not arise from Gyrovector Group properties. Additionally, Bdeir et al. (2023) proposed an efficient batch normalization algorithm based on the Lorentz model, utilizing the Lorentz centroid and a mathematical rescaling operation.

Numerical Stability of Hyperbolic Neural Networks

Training of Hyperbolic Neural Networks developed on the Lorentz Model can lead to instability and floating point error due to rounding since the volume of the Lorentz model grows exponentially with respect to radius. Sometimes, people work with these floating-point representations in 64-bit precision with a higher memory cost. Mishne et al. (2023),Guo et al. (2022b),Mathieu et al. (2019) proposed some versions of feature clipping and Euclidean reparameterization to mitigate these issues. However, they largely overlooked some critical aspects, such as defining a fully hyperbolic convolutional layer or classifiers like MLR, which are essential for various computer vision tasks. In this paper, we fully address this gap by developing a novel architecture from the ground up, along with the theory of its universal consistency.

4 Preliminaries

This section discusses the preliminaries of Riemannian Manifolds and Hyperbolic Geometry which would underpin the introduction of our proposed framework.

4.1 Riemannian manifold, Tangent space, and Geodesics

Roughly speaking, an n-dimensional Manifold $\mathcal{M}$ is a topological space that locally resembles $\mathbb{R}^n$ Tu (2017). For each point $x \in \mathcal{M}$, we can define the *Tangent Space* $T_x(\mathcal{M})$ as the first order linear approximation of $\mathcal{M}$ at x. We call $\mathcal{M}$ as a *Reimannian Manifold* if there is a collection of metrics $g := \{g_x : T_x(\mathcal{M}) \times T_x(\mathcal{M}) \to \mathbb{R}, x \in \mathcal{M}\}$ at every point of $\mathcal{M}$ do Carmo (1992). This metric induces a distance function between two points $p, q \in \mathcal{M}$ joined by a piecewise smooth curve $\gamma : [a, b] \to \mathcal{M}$ with $\gamma(a) = p, \gamma(b) = q$ and the distance between p and q is defined as $L(\gamma) := \int_a^b g_{\gamma(t)}(\gamma'(t), \gamma'(t))^{1/2} dt$. The notion of *Geodesic* between two such points is meant to be that curve γ for which $L(\gamma)$ attains the minimum and that $L(\gamma)$ is referred as the *Geodesic Distance* between p and q in that case. Given such a Riemannian Manifold $\mathcal{M}$ and two linearly independent vectors u and v at $T_x(\mathcal{M})$, we define the sectional curvature at x as $k_x(u, v) := \frac{g_x(R(u,v)v,u)}{g_x(u,u)g_x(v,v)-g_x(u,v)^2}$, where

R being the Riemannian curvature tensor defined as $R(u,v)w := \nabla_u \nabla_v w - \nabla_v \nabla_u w - \nabla_{(\nabla_u v - \nabla_v u)} w$ [$\nabla_u v$ is the directional derivative of v in the direction of u, which is also known as the *Rimannian Connection* on $\mathcal{M}$.]

We use these notions to define an n-dimensional model *Hyperbolic Space* as the connected and complete Riemannian Manifold with a constant negative sectional curvature. There are various models in use for Hyperbolic Spaces, such as Poincaré Ball Model, Poincaré half Space Model, Klein-Beltrami Model, Hyperboloid Model, etc, but the celebrated *Killing-Hopf Theorem* Lang (1995) asserts that for a particular curvature and dimension, all the model hyperbolic spaces are isometric. This allows us to develop our architecture uniquely (without worrying much about performance variations) over a particular model space, where we choose to work with the Poincaré Ball model for our convenience. Here, we have briefly mentioned the critical algebraic operations on this model required for our purpose.

4.2 Poincaré Ball Model

For a particular curvature $k(<0)[c=-k]$, an $n-$ dimensional Poincaré Ball model contains all of its points inside the Ball of radius $1/\sqrt{c}$ embedded in $\mathbb{R}^n$ Lee (2006). The geodesics in this model are circular arcs perpendicular to the spherical surface of radius $1/\sqrt{c}$. The geodesic distance between two points p and q (where $\|p\|, \|q\| < 1/\sqrt{c}$) is defined as

$$d(p,q) := 2\sinh^{-1}\left(\sqrt{2\frac{\|p-q\|^2}{c(\frac{1}{c}-\|p\|^2)(\frac{1}{c}-\|q\|^2)}}\right),$$

From now on, we will denote $\mathbb{D}_c^n$ as the $n-$ dimensional Poincaré Ball with curvature $-c$.

4.3 Gyrovector Space

The concept of Gyrovector Space, introduced by Abraham A. Ungar [see Ungar (2022)], serves as a framework for studying vector space structures within Hyperbolic Space. This abstraction allows for defining special addition and scalar multiplications based on weakly associative gyrogroups. For a detailed geometric formalism of these operations, Vermeer's work Vermeer (2005) provides an in-depth exploration.

In this context, we will briefly Balluss Möbius Gyrovector Addition and Mobius Scalar Multiplication on the Poincar'e Ball. Due to isometric transformations between hyperbolic spaces of different dimensions, the same additive and multiplicative structures can be obtained for other model hyperbolic spaces (refer Fu et al. (2019)). Utilizing Möbius addition and multiplication is essential when evaluating intrinsic metrics like the Davies-Bouldin Score or Calinski-Harabasz Index to assess the performance of our proposed algorithm.

1. **Möbius Addition:** For two points u and v in the Poincaré Ball, the Möbius addition is defined as: $u \oplus_c v := \frac{(1+2c<u,v>+c\|v\|^2)u+(1-c\|u\|^2)v}{1+2c<u,v>+c^2\|u\|^2\|v\|^2}$, where c is the negative of the curvature for the Poincaré Ball.
2. **Möbius Scalar Multiplication:** For $r \in \mathbb{R}$, $c > 0$ and u in the Poincaré Ball, the scalar multiplication is defined as: $r \otimes_c u := \frac{1}{\sqrt{c}} \tanh\left(r \tanh^{-1}(\sqrt{c}\|u\|)\right) \frac{u}{\|u\|}$ This addition and scalar multiplication satisfy the axioms pertaining to the Gyrovector Group [see Ungar (2022)].

4.4 Exponential & Logarithmic Maps

For any $x \in \mathbb{D}_c^n$, the $\exp_x^c : T_x(\mathbb{D}_c^n) \subseteq \mathbb{R}^n \to \mathbb{D}_c^n$ translates a point in the tangent space of the Poincare Ball and projects it on the Poincaré Ball along the unit speed geodesic starting from $x \in \mathbb{D}_c^n$ in the direction $v \in T_x(\mathbb{D}_c^n)$. The Logarithmic map does precisely the opposite, i.e., $\log_x^c : \mathbb{D}_c^n \to T_x(\mathbb{D}_c^n) \subseteq \mathbb{R}^n$, projecting a point from the Poincaré Ball back to the tangent space at $x \in \mathbb{D}_c^n$ along the reverse of the geodesic traced by the Exponential Map. Their formulations are explicitly given as follows:

$$\exp_x^c(v) := x \oplus_c \left(\tanh\left(\sqrt{c}\frac{\lambda_x^c\|v\|}{2}\right)\frac{v}{\sqrt{c}\|v\|}\right)$$

and

$$\log_x^c(y) := \frac{2}{\sqrt{c}\lambda_x^c} \tanh^{-1}\left(\sqrt{c}\| -x \oplus_c y\|\right) \frac{-x \oplus_c y}{\| -x \oplus_c y\|},$$

for $y \neq x$ and $v \neq 0$ and the Poincaré conformal factor $\lambda_x^c := \frac{2}{(1-c\|x\|^2)}$.

5 Proposed Method

In this section, we will unravel the design strategy of expansive Hyperbolic Deep Convolutional Neural Networks (eHDCNN). Let us first define the hyperbolic convolution operation on Poincaré Ball. Assume two functions f and g from $\mathbb{R}^n \to \mathbb{R}$, we define the convolution between f and g as:

$$f \star g(x) := \int_{\mathbb{R}^n} f(z)g(x-z)dz.$$

Analogously, we define hyperbolic convolution using logarithmic and exponential maps.

Definition 1 (Hyperbolic Convolution (Continuous Version))**.** *For $x \in \mathbb{D}_c^n$, we define the convolutions of two real-valued functions f, g on $\mathbb{R}^n$ as a map from $: \mathbb{D}_c^n \to \mathbb{D}_c^1$ as*

$$f \star g(x) := \exp_0^c\left[\int_{\mathbb{D}_c^n} f(\log_0^c(z))g(\log_0^c(-z \oplus_c x))\lambda(z)\right], \tag{3}$$

where $\lambda(z) := \frac{dz}{1-c\|z\|^2}$.

Remark 1. *Note that for two real-valued functions f, g, their hyperbolic convolution is a map $h : \mathbb{D}_c^n \to \mathbb{D}_c^1$. We want to keep the range of the output function of the convolution in $\mathbb{D}_c^1$, since in the deep convolutional setup we will again convolute the output with some other filters.*

Definition 2 (Hyperbolic Expansive and Contractive Convolution (Ballrete Version))**.** *Let $\boldsymbol{w} := \{w_j\}_{j=-\infty}^{\infty}$ be an infinite dimensional vector whose elements are in $\mathbb{R}$ with finitely many non-zero entries in $\boldsymbol{w}$. Explicitly we assume $w_j \neq 0$ for $0 \leq i \leq s$. Among two widely used types of 1-D convolutions in $\mathbb{R}^n$, we talk about only the expansive and contractive type convolutions. Let $\boldsymbol{v} = \{v_1, ..., v_n\} \in \mathbb{D}_c^n$. We define the Hyperbolic Expansive Convolution $(*_h)$ and the Hyperbolic Contractive Convolution $(\star_h)$ in the following way:*

Let $\boldsymbol{v'} := \log_0^c(\boldsymbol{v}) = (v_1', v_2', ..., v_n') \in \mathbb{T}_0^c(\mathbb{D}_c^n) \subseteq \mathbb{T}_0^c(\mathbb{R}^n)$, i.e. $\boldsymbol{v'}$ is an element of the tangent bundle at 0 of $\mathbb{D}_c^n$.

1. ***Hyperbolic Expansive Convolution:***$(\boldsymbol{w} * \boldsymbol{v'}) = \sum_{l=1}^{n} w_{j-l}v_l'$ *for $j = 1, 2, ..., n+s$. Therefore $(\boldsymbol{w} * \boldsymbol{v'}) \in \mathbb{R}^{n+s}$. We apply the* exp *map to put it back in $\mathbb{D}_c^{n+s}$. Finally, we define*

$$\boldsymbol{w} *_h \boldsymbol{v} := \exp_0^c(\boldsymbol{w} * \log_0^c(\boldsymbol{v})). \tag{4}$$

2. ***Hyperbolic Contractive Convolution:*** *The usual contractive convolution for $\boldsymbol{w}$ and $\boldsymbol{v'}$ is defined as, $\boldsymbol{w} \star \boldsymbol{v'} = \sum_{l=j-s}^{j} w_{j-l}v_l, j = s+1, ..., n$. We define $\star_h$ between $\boldsymbol{w}$ and $\boldsymbol{v}$ as*

$$\boldsymbol{w} \star_h \boldsymbol{v} := \exp_0^c(\boldsymbol{w} \star \log_0^c(\boldsymbol{v})), \tag{5}$$

which lies in $\mathbb{D}_c^{n-s}$.

Remark 2. *Note that for $c = 0$, we will retrieve the usual sparse Toeplitz operators of dimensions $n \times (n+s)$ and $n \times (n-s)$ from those two cases.*

Having Ballussed all the necessary terminologies, we are finally set to define the complete architecture of the Hyperbolic Deep Convolutional Neural Network (HDCNN).

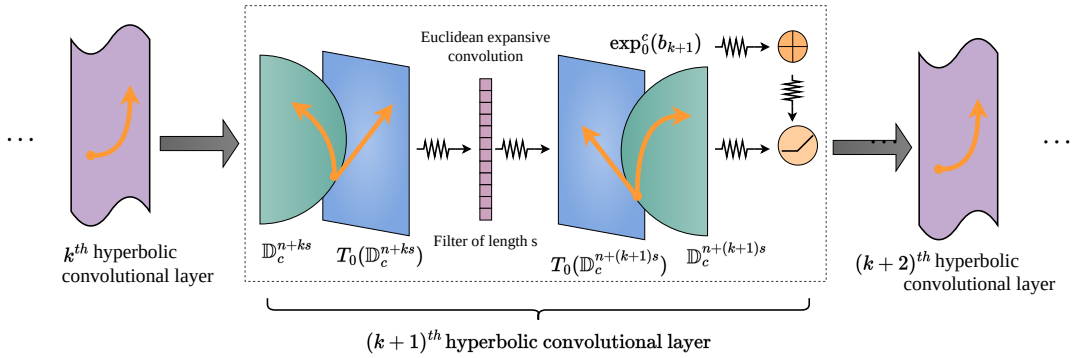


Figure 2: The complete workflow of expansive hyperbolic 1-D convolutional layer on Poincarê Ball is presented. (best view in digital format)

Definition 3 (Hyperbolic Deep Convolutional Neural Network (HDCNN))**.** *Let $L \in \mathbb{N}$ be the number of hidden layers in the network. For a given set of filters $\{\boldsymbol{w_k}\}_{k=1}^{L}$ and set of compatible bias vectors $\{\boldsymbol{b_k}\}_{k=1}^{L}$ and a vector $\boldsymbol{a_L} = \{a_1, a_2, ..., a_{n_L}\}$ [Note that these vectors all lie in Euclidean Spaces of Appropriate Dimensions]. Let $\sigma(t) := \max\{0, t\}$ be the ReLU, acting component-wise for the multidimensional operation. We also assume the dimension of the output layer is n_L and $\boldsymbol{h}_L(x) := \{h^1(x), h^2(x), ..., h^{n_L}(x)\} \in \mathbb{D}_c^{n_L}$. The HDCNN is defined as:*

$$\boldsymbol{h}_k(x) = \sigma(\boldsymbol{w}_k \circ \boldsymbol{h}_{k-1}(x) \oplus_c \exp_0^c(\boldsymbol{b}_k)), \tag{6}$$

where $\boldsymbol{h}_k(x)$ is the output from the $k-th$ hidden layer for $k \in \{1, 2, ..., L-1\}$ and $h_k(x) \in \mathbb{D}_c^{n+ks}$, where $\circ$ can be either $*_h$ or $\star_h$ as defined in Definition 2, $\boldsymbol{h}_0(x) = x$ and the final output is given as:

$$h_L(x) = \exp_0^c\left[\boldsymbol{a}_L \cdot \log_0^c(\boldsymbol{h}_L(x))\right]. \tag{7}$$

The transformation of a vector, lying as a geodesic on Poincaré Ball, is shown in Figure 2 through hyperbolic convolution performed in an intermediate layer. This transformation projects the vector as another geodesic in a higher dimensional Poincaré Ball to the subsequent layer.

Remark 3. *If we restrict our focus only to the Expansive case (eHDCNN), note that the dimension of the input to each hidden layer is getting bigger by s units every time. More explicitly, if we have started with $x \in \mathbb{D}_c^n$, and $\boldsymbol{w}_1$ is the first filter of length s, then $\boldsymbol{h}_1(x) \in \mathbb{D}_c^{n+s}$, which is the input dimension of the second hidden layer. Iteratively, the input dimension of the $k-th$ hidden layer is as same as the dimension of $\boldsymbol{h}_{k-1}$, which lies in $\mathbb{D}_c^{n+(k-1)s}$. Finally, when we reach the output layer, the output dimension will be $n + Ls$, i.e., $n_L = n + Ls$. To make the Mö bius addition and the Möbius multiplications compatible, we need to have $\boldsymbol{a}_L \in \mathbb{R}^{n+Ls}$. Also note that for $c = 0$, this architecture is reduced to the HDCNN architecture described in Lin et al. (2022b).*

6 Theoretical Analyses

We will now provide the proof for universal consistency following the framework established in Lin et al. (2022b). While we will appropriately generalize the results to the hyperbolic setting, it is first necessary to define some statistical terminologies to comprehend the mechanism of eHDCNN.

We consider a dataset $\mathcal{D} = \{z_i\}_{i=1}^m = \{x_i, y_i\}_{i=1}^m$, where the samples are assumed to be independent and identically distributed according to a Borel probability measure ρ on the space $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$. Here, $x_i \in \mathcal{X} \subseteq \mathbb{D}_c^n$ and $y_i \in \mathcal{Y} \subseteq \mathbb{D}_c^1$. We assume $\mathcal{X}$ is a compact set for this Ballussion. The goal is to learn a function $f_{\mathcal{D}} : \mathcal{X} \to \mathbb{R}^1$ that minimizes the following *Hyperbolic Generalization Error (HGE)*:

$$\mathcal{E}(f) := \int_{\mathcal{Z}} (f(x) - \log_0^c(y))^2 d\rho. \tag{8}$$

Remark 4. *The reason behind taking the* log *of* $y \in \mathcal{Y}$ *is that, the logarithm function will project back* $y \in \mathcal{Y}$ *to* $\mathbb{T}_0(\mathbb{D}_c^1) \subseteq \mathbb{R}^1$*. Hence, taking the difference between two real numbers will make sense. Also, if* $c \to 0$*, we will return the usual generalization error on the Euclidean Spaces.*

Lemma 1. *The Hyperbolic Regression Function (HRF)* $f_\rho(x) := \int_{\mathcal{Y}} \log_0^c(y) d\rho(y|x)$*, defined by the means of conditional distribution* $\rho(\cdot|x)$ *of* ρ *at* $x \in \mathcal{X}$ *minimizes the HGE.*

The next lemma will deduce what we aim to minimize.

Lemma 2. *For any* $f : \mathbb{D}_c^n \to \mathbb{R}^1$*, we have*

$$\mathcal{E}(f) - \mathcal{E}(f_\rho) = \|f - f\rho\|_{L^2_{\rho_{\mathcal{X}}}},$$

where $\rho_{\mathcal{X}}(x) := \int_{\mathcal{Y}} \rho(x, y) d\mathcal{Y}(y)$*, for each* $x \in \mathcal{X}$*, the marginal distribution of* ρ *on* $\mathcal{X}$*.*

The estimator that minimizes the hyperbolic generalization error is that estimator which minimizes the empirical error over the class of all functions expressed by our eHDCNN architecture. Hence the corresponding estimator or the *Empirical Risk Minimizer (ERM)* is defined as:

$$f_{\mathcal{D},L,s} := \arg \min_{f \in \mathcal{H}_{L,s}} \epsilon_{\mathcal{D}}(f), \tag{9}$$

where

$$\mathcal{E}_{\mathcal{D}}(f) := \frac{1}{m} \sum_{i=1}^{m} (f(x_i) - \log_0^c(y_i))^2$$

denotes the empirical risk (HERM) associated with the function f and for the filters $\boldsymbol{w_k}$ for $k \in \{1, 2, ..., L\}$ of length $s_k = d + ks$ and

$$\mathcal{H}_{L,s} := \{h_L(x), \boldsymbol{w}_k, \boldsymbol{b}_k \in \mathbb{R}^{d+ks}, k = 1, 2, ..., L\}$$

is the set of all hyperbolic outputs produced by the eHDCNN defined by 7.

To establish consistency, we must demonstrate that as the sample size $m \to \infty$, the sequence of estimators converges to the true value. Formally, a sequence of estimators is *strongly universally consistent* if it converges almost surely to the underlying parameter. In regression, this implies that empirical error estimators—derived via empirical risk minimization—converge to the generalization error over the space of square-integrable functions (a Hilbert space) with respect to the conditional output distribution. In the hyperbolic setting, we formalize this notion as follows:

Definition 4. *A sequence of Hyperbolic Regression Estimators (HRE)* $(\{f_m\}_{m=1}^{\infty})$ *built through ERM is said to be strongly universally consistent if it satisfies the condition:*

$$\lim_{m \to \infty} \mathcal{E}(f_m) - \mathcal{E}(f_\rho) = 0$$

almost surely, for every Borel probability distribution λ *such that* $\log_0^c(\mathcal{Y}) \in L^2(\lambda_{(\mathcal{Y}|x)})$*.*

The main result we will be going to prove here will be the following Theorem, which will prove the strong universal consistency of eHDCNN when the Hyperbolic Empirical Risk is minimized. The following Theorem considers a sequence of eHDCNNs as the universal approximators of continuous functions, where the depth of the network has been taken as a sequence depending upon the sample size of our dataset.

Theorem 1. *Suppose* $L = L_m \to \infty$*,* $M = M_m \to \frac{1}{\sqrt{c}}$*,* $m^{-\theta} M_m^2 \left[1 + \frac{1}{M_m \sqrt{c}} \tanh^{-1}(M_m \sqrt{c})\right]^2 \to 0$ *[constrained truncation on the power of sample size] and*

$$\frac{\left(\frac{1}{\sqrt{c}} \tanh^{-1}(M_m \sqrt{c})\right)^4 L_m^2 (L_m + d) \log(L_m)}{m^{1-2\theta}} \times \log\left(\left(\frac{1}{\sqrt{c}} \tanh^{-1}(M_m \sqrt{c})\right) m\right) \to 0, \tag{10}$$

hold for $\theta \in (0, 1/2)$ and input filter length as $2 \leq s \leq d$. Then $\pi_{M_m} f_{D,L_m,s}$ is strongly universally consistent, where $\pi_M(l) := \min\{M, |l|\} \cdot sign(l)$ is the well-known truncation operator.

Remark 5. *If we put $\lim c \to 0$ in Theorem 1, we get back Theorem 1 in Lin et al. (2022b). Therefore, Theorem 1 is a more generalized version, which is reduced to its Euclidean version for curvature 0.*

Remark 6. *When we intend to perform the convergence analysis of a series in mathematical analysis, we first consider the partial sum of the series up to a certain term (let's say up to the $k-$th term) and then try to observe the behavior of the series by letting $k \to \infty$. This idea generates the involvement of the truncation operator in Theorem 1. Note that instead of taking $M_m \to \infty$ [which is used in Lin et al. (2022b)], we have made $M_m \to \frac{1}{\sqrt{c}}$ (letting our samples lie close to the boundary of the Poincaré Ball, whose radius is $\frac{1}{\sqrt{c}}$). As $M_m \to \frac{1}{\sqrt{c}}$, $\tanh^{-1}(M_m\sqrt{c}) \to \infty$, so does $M_m\left(\frac{1}{M_m\sqrt{c}}\tanh^{-1}(M_m\sqrt{c})\right)$. It will ease our work for giving an upper bound on the covering number of $\mathcal{H}_{L,s}$ in terms of the truncation limit. Our adoption of the truncation operator is motivated by the widespread application of this operator in proving the universal consistency of various learning algorithms Györfi et al. (2002), Lin et al. (2022b).*

Apart from the truncation operator in Theorem 1, several constraints are involved which are crucial to guarantee universal consistency. The constraint on depth $L_m \to \infty$ appears naturally as it is necessary for the universal approximation used in Lemma 7. The growth of the truncation limit concerning sample size m is given by $m^{-\theta}M_m^2\left[1 + \frac{1}{M_m\sqrt{c}}\tanh^{-1}(M_m\sqrt{c})\right]^2 \to 0$ instead of $M_m^2 m^{-\theta} \to 0$ [given in Lin et al. (2022b)] to incorporate the growth restriction of sample error in term of two increasing univariate functions $h_1(M_m)h_2(m^{-1})$, where $h_1(x) = x^2\left[1 + \frac{1}{x\sqrt{c}}\tanh^{-1}(x\sqrt{c})\right]^2$ and $h_2(x) = x^\theta, \theta > 0$. Finally, the constraint in equation 10 will ensure the absolute difference between the generalization error and empirical error goes to 0, by enforcing the condition that the combined growth effect of the pseudo-dimension and the metric entropy of the class $\mathcal{H}_{L,s}$ is outpaced by the number of input samples, which will be used to prove Lemma 6.

Remark 7. *Theorem 1 only demonstrates the universal consistency of the eHDCNN architecture for one-dimensional convolution. The primary restriction comes from the infeasibility of the convolutional factorization that appeared in Zhou (2020) [also described in Lin et al. (2022b)]. Since the analysis in the hyperbolic set-up also relies on the universal approximation for the conventional eDCNN, the question of universal consistency remains open for two or higher-dimensional eHDCNN structures.*

We now dive into proving Theorem 1. Our main ingredient will be a version of Concentration Inequality [Theorem 11.4,Györfi et al. (2002)] after suitably adjusting the upper bound of the metric entropy concerning pseudo-dimension [Lemma 4, Lin et al. (2022b)]. Although our approach is similar to Lin et al. (2022b) to some extent, we have been able to derive a stronger version of Lemma 6 in Lin et al. (2022b) as presented in the proof of Lemma 6 in this paper, showing that the truncated empirical error converges to the truncated generalization error much faster in the case of hyperbolic convolution compared to the traditional Euclidean one. This will be established once we present our experimental results in terms of different curvatures (curvature 0 denotes the experiment has been done using eDCNN).

To prove Theorem 1 we divide our works into three parts as demarcated in Lin et al. (2022b) and will develop the appropriate hyperbolic versions of the corresponding results. We begin with expanding the bounds on the covering number for the class of functions defined in 7. We first need several terminologies.

Let ν be a probability measure on $\mathcal{X} \in \mathbb{D}_c^n$. For a function $f : \mathcal{X} \to \mathbb{R}$, we set

$$\|f\|_{L^p(\nu)} := \left(\int_{\mathcal{X}} |f(x)|^p \nu(x) d\mathcal{X}(x)\right)^{1/p}.$$

Denote by $L^p(\nu)$ the set of all functions with $\|f\|_{L^p(\nu)} < \infty$. For $\mathcal{A} \subseteq L^p(\nu)$, we denote $\mathcal{N}(\epsilon, \mathcal{A}, \|\cdot\|_{L^p(\nu)})$ the covering number of $\mathcal{A}$ in $L^p(\nu)$, which is the least number of Ball s of radius ϵ needed to cover up $\mathcal{A}$ with respect to the $\|\cdot\|_{L^p(\nu)}$ metric. In particular we denote $\mathcal{N}_p(\epsilon, \mathcal{A}, x_1^m) := \mathcal{N}_p(\epsilon, \mathcal{A}, \|\cdot\|_{L^p(\nu_m)})$, where ν_m is the emperical measure for the dataset $x_1^m := \{x_1, x_2, ..., x_m\} \in \mathcal{X}^m$. Further we define$\mathcal{M}(\epsilon, \mathcal{A}, \|\cdot\|_{L^p(\nu)})$ to be the $\epsilon-$packing number of $\mathcal{A}$ with respect to the $\|\cdot\|_{L^p(\nu)}$ norm, which is the largest integer N such that given any subset $\{g_1, g_2, ..., g_N\}$ of $\mathcal{A}$ satisfies $\|g_i - g_j\| \geq \epsilon$ for all $1 \leq i < j \leq N$.

Next, we will mention lemma 9.2 from Györfi et al. (2002), which expresses a relation involving inequalities among the covering and packing numbers.

Lemma 3. *Let $\mathcal{G}$ be a class of functions from $\mathcal{X} \to \mathbb{R}$ and ν be a probability measure on $\mathcal{X}$. For $p \geq 0$ and $\epsilon > 0$, we have*

$$\mathcal{M}(2\epsilon, \mathcal{G}, \|\cdot\|_{L^p(\nu)}) \leq \mathcal{N}(\epsilon, \mathcal{G}, \|\cdot\|_{L^p(\nu)}) \leq \mathcal{M}(\epsilon, \mathcal{G}, \|\cdot\|_{L^p(\nu)}).$$

In particular,

$$\mathcal{M}_p(2\epsilon, \mathcal{G}, x_1^m) \leq \mathcal{N}_p(\epsilon, \mathcal{G}, x_1^m) \leq \mathcal{M}_p(\epsilon, \mathcal{G}, x_1^m).$$

Next, we have to derive an estimate of the upper bound of the Packing number for the pseudo dimension.

Since the Lemma 2, 3, and 4 from Capacity Estimates in Appendix A of Lin et al. (2022b) are taken from results proved on general metric spaces, we will just state Lemma 4 from Lin et al. (2022b) without proof in the context of hyperbolic space, which we will use later.

Lemma 4. *For $0 < \epsilon \leq M$ and c^* being an absolute constant, we have*

$$\log_2 \sup_{x_m^1 \in \mathcal{X}^m} \mathcal{N}_1\left(\epsilon, \pi_M \mathcal{H}_{L,s}, x_1^m\right) \leq c^* L^2 (Ls + d) \log(L(s+d)) \log \frac{M}{\epsilon}.$$

We define the hyperbolic version of the generalization error (HGE) as

$$\mathcal{E}_{\pi_M}(f) := \int_{\mathcal{Z}} \left(f(x) - \log_0^c(y_M)\right)^2 d\rho,$$

and the *Hyperbolic Empirical Error (HEE)* (truncated) as

$$\mathcal{E}_{\pi_M, D}(f) := \frac{1}{m} \sum_{i=1}^{m} \left(f(x_i) - \log_0^c(y_{i,M})\right)^2,$$

where $l_M := \min\{M, |l|\} \cdot sign(l)$, the well known truncation operator.

We now provide a convergence criterion for the HEE estimates to the HGE estimate. We will use a hyperbolic version of the concentration inequality as given in Lemma 5, Lin et al. (2022b).

A more generalized version of Theorem 11.4 Györfi et al. (2002) can be presented as follows:

Lemma 5. *We assume $|y| \leq B$ and $B \geq \frac{1}{\sqrt{c}}$. For a set of functions $\mathcal{F}$ from $f : \mathcal{X} \to \mathbb{R}$ satisfying $|f(x)| \leq B$ and for all $m \geq 1$, we have*

$$\begin{aligned}
&\mathbb{P}[\exists f \in \mathcal{F} : \epsilon(f) - \epsilon(f_\rho) - (\epsilon_D(f) - \epsilon_D(f_\rho)) \geq \epsilon(\alpha + \beta + \epsilon(f) - \epsilon(f_\rho))] \\
&\leq 14 \sup_{x_1^m \in \mathcal{X}^m} \mathcal{N}_1\left(\frac{\beta\epsilon}{20B}, \mathcal{F}, x_1^m\right) \exp\left(-\frac{\epsilon^2(1-\epsilon)\alpha m}{214(1+\epsilon)B^4}\right),
\end{aligned}$$

where $\alpha, \beta > 0$ and $\epsilon \in (0, 1/2)$.

Based on Lemma 5, the following Lemma will lay out the convergence criterion of the Truncated HEE estimates.

Lemma 6. *When $m^{-\theta} M_m^2 \left[1 + \frac{1}{M_m\sqrt{c}} \tanh^{-1}(M_m\sqrt{c})\right]^2 \to 0$ and equation 10 holds for $\theta \in (0, 1/2)$, then we have*

$$\lim_{m\to\infty} \mathcal{E}_{\pi_{M_m}}(\pi_{M_m} f_{D,L,s}) - \mathcal{E}_{\pi_{M_m}, D}(\pi_{M_m} f_{D,L,s}) = 0$$

holds almost surely.

We are finally in a position to prove Theorem 1; we will give our final lemma, which will complete the proof for universal consistency.

Lemma 7. *Let $\Omega \subseteq \mathbb{D}_c^d$ be compact and $2 \leq s \leq d$. Then for any $f \in \mathcal{C}(\Omega)$, there exist a sequence of filters $\boldsymbol{w}$ and bias vectors $\boldsymbol{b}$ of appropriate dimensions and $f_L^{\boldsymbol{w},\boldsymbol{b}} \in \mathcal{H}_{L,s}$ such that*

$$\lim_{L\to\infty} \|f - f_L^{\boldsymbol{w},\boldsymbol{b}}\|_{\mathcal{C}(\Omega)} = 0.$$

Remark 8. *We notice from the proof of Lemma 6 that the truncated HEE estimates converge much faster to the corresponding HGE than their Euclidean equivalents. This property gives the eHDCNN architecture an edge over the eDCNN for faster training with much fewer training iterations needed. Roughly speaking, since each layer is taking input from a Poincaré Ball, which in turn expresses the complex representation of the data to the next layer even before the information gets carried out to the next layer directly from the previous layer, the architecture is very quick to learn the internal representation of the data. This will be evident from our simulation results, showing the ascendancy of our architecture over its Euclidean version to achieve lower error rates much faster for certain regression problems.*

7 Experiments & Results

We will demonstrate the efficacy of eHDCNN by conducting experiments on synthetic and real-world datasets. Our Python-based implementation is available at https://github.com/kushalbose92/eHDCNN.

7.1 Synthetic Datasets

We will construct two regression tasks based on the following functions,

$$f(x) = \frac{\sin(\|x\|_2)}{\|x\|_2}, \qquad g(x) = \frac{\sqrt{\|x\|_2}}{1+\sqrt{\|x\|_2}}.$$

We used the regression model $y = h(x) + \epsilon$ (where h can be either f or g) to generate the training samples, where $\epsilon \sim \mathcal{N}(0, 0.01)$ and $x \sim \text{unif}([-1,1]^{10})$. A fixed set of 800/200 samples for the train/test split is used for the experiment, except that the test data are taken without the Gaussian noise. We have used a filter size of length 8 and the number of layers 4. We have trained our model over 100 iterations for 800 training samples and recorded the mean RMSE. We repeat the experiments for six different sets of curvatures. Refer to Figures 3(a) and 3(b) for the detailed illustration.

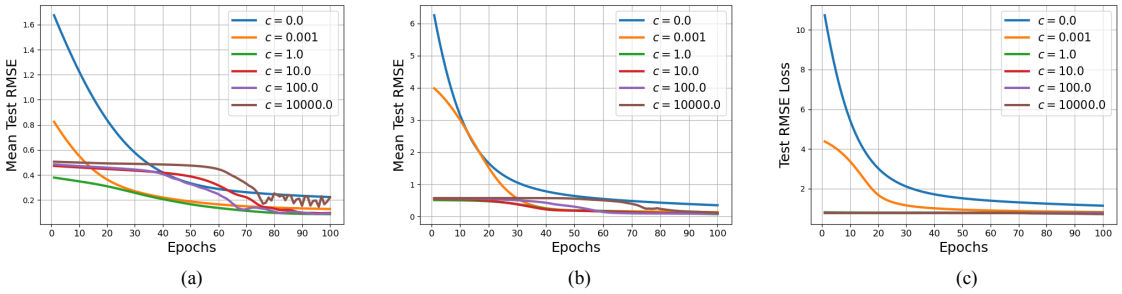
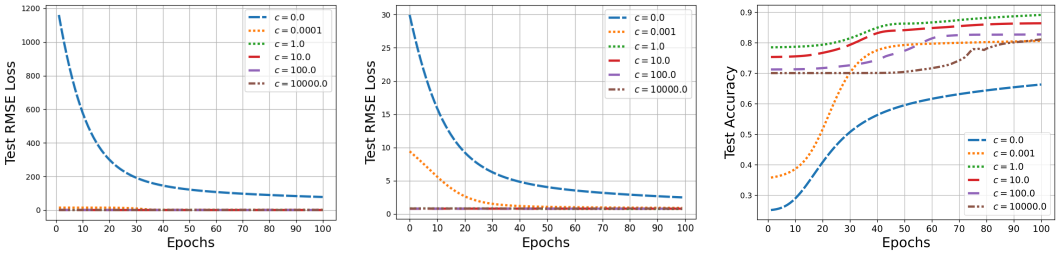


Figure 3: The performance analysis of eHDCNN with varying space curvatures (a) for $f(x)$ and (b) for $g(x)$, and (c) House price prediction is demonstrated. The Root Mean Square Error (RMSE) decreases faster with increasing curvature, justifying the utility of applying hyperbolic convolution. (best view in digital format)

The curves are evidence of the faster convergence of test RMSE loss during the entire training process, which validates the Remark 8. The loss curves are much steeper when the curvatures are more significant than zero compared to the same as the Euclidean counterpart. One point should be noted that the performance of eHDCNN started to deteriorate with the higher value of curvature. The phenomenon can be attributed to the contraction of Poincaré Ball with a very high curvature. Thus, the loss curves seem to be overlapping. Yet, the performance is commendable when eHDCNN is trained in the hyperbolic space of low curvature.



(a) (b) (c)

Figure 4: The performance analysis of eHDCNN with varying space curvatures (a) for Superconductivity (b) for Wave Energy, and (c) test accuracy for WISDM are demonstrated. The Root Mean Square Error (RMSE) decreases faster for both (a) and (b) with increasing curvature. On the contrary, test accuracy increases in (c), justifying the utility of employing hyperbolic convolution. (best view in digital format)

Table 1: The details of four real-world datasets are presented.

Dataset	Superconductivity	Wave Energy Converters	House Price Prediction	WISDM
No of samples	288000	21263	545	1073120
No. of features	81	81	12	3
No. of classes	-	-	-	6
Target task	Regression	Regression	Regression	Classification

7.2 Real-world Datasets

We considered four real-world datasets to showcase the effectiveness of eHDCNN. The details of the datasets and the hyperparameters are provided respectively in Table 1 and 2.

Table 2: The complete details of hyperparameters for four real-world datasets are presented to reproduce the results.

Hyperparameters	Superconductivity	Wave Energy Converters	House Price Prediction	WISDM
No of layers	4	4	4	4
length of input filter	8	8	8	9
Noise	No	No	No	No
Learning Rate	0.01	0.01	0.01	0.01
Weight decay	0.0005	0.0005	0.0005	0.0005
Train/test split	0.80	0.80	0.80	0.70
No of samples	288000	21263	545	1073120
Input dimension	81	81	12	240
Batch Size	128	128	Full	128

7.2.1 Regression Task

We include 3 real-world regression datasets to demonstrate the performance of eHDCNN over the prevailing DCNN. We deploy the same eHDCNN architecture with 4 layers, and the length of the input filter is 8 for all three regression tasks. We split the entire dataset into 80% samples for training and the rest 20% samples for testing. We record the standardized test RMSE over the number of iterations during the training phase.

House Price Prediction

We consider the widely available house price prediction dataset Wang & Zhao (2022) to solve the regression task. This dataset consists of 545 samples with 12 input features such as area, number of bedrooms, furnishing

status, air conditioning, etc. At first, we standardize the entire data after numerically encoding its categorical column. We have trained our model using 4 layers and with an input filter length of 8. The test RMSE has been plotted against training iterations for six different curvatures in 3(c), where the curvature 0 means that the test RMSE has been plotted based on the eDCNN model.

Superconductivity

As described in Hamidieh (2018), this dataset contains 21263 samples, each with 81 features like mean atomic mass, entropy atomic mass, mean atomic radius, entropy valence etc, along with the output feature as the critical temperature in the 82nd column. We split the dataset into $80:20$ for our training and testing purposes. We will train our model with a mini-batch of size 128 in each training iteration. The test RMSE has been plotted against the number of training iterations for six different curvatures in 4 (a), where the curvature 0 indicates that the test RMSE has been taken based on the eDCNN model.

Wave-Energy Converters

As described in Mehdipour et al. (2024), this dataset contains 288000 samples, each with 81 features. This data set consists of positions and absorbed power outputs of wave energy converters (WECs) in four real wave scenarios from the southern coast of Australia (Sydney, Adelaide, Perth, and Tasmania). We split the dataset into $80:20$ for our training and testing purposes. Similar to the Superconductivity dataset, we will train our model with a mini-batch of size 128 each time. For the test RMSE plot against the number of training epochs, we refer to 4 (b).

7.2.2 Classification Task

The only dataset we include for solving classification tasks is WISDM.

WISDM

We have applied eHDCNN on the WISDM, a well-adopted Human Activity Recognition (HAR) dataset Kwapisz et al. (2011). As it is described in Lin et al. (2022c), this dataset has six types of human activities such as cycling, jogging, sitting, standing, going upstairs and downstairs, with the corresponding accelerations along x, y, and z axes at different timestamps and several user id ranging from 1 to 36. We have used the user IDs from 1 to 28 for training and the rest for testing. We have put 80 consecutive timestamps for each of the six classes together to make our input dimension $80 \times 3 = 240$. After this conversion, our training dataset has 10172 samples, and the test dataset has 3242 number of samples. Our experiment is carried out on a network with 4 layers with input filter length as 9. We have trained our model with a mini-batch of size 128 in each epoch.

Discussion

We run experiments on the House Price Prediction, Superconductivity, Wave-Energy Converters, and WISDM, where plots can be seen respectively in Figures 3(c), 4(a), 4(b), and 4(c). Test RMSE loss is the metric for the first three datasets, and test accuracy is the metric for the last one. The plots elucidate that the corresponding metric performs better when the curvature increases than the Euclidean variant. The better performance underscores the efficacy of hyperbolic architecture dominates over its Euclidean counterpart. One common point is that performance further degrades when the value of the curvature lies in a very high range. It occurs due to the shrinkage of the Poincaré Ball with a very high value of the curvature.

8 Ablation Studies

We conduct an ablation study to study the effect of the filter length and number of hidden layers of the eHDCNN. The experiment is performed on Superconductivity. The filter length and number of layers are chosen respectively from the sets $s = \{6, 7, 8, 9\}$ and $L = \{3, 4, 5, 6\}$. We run experiments for each pair

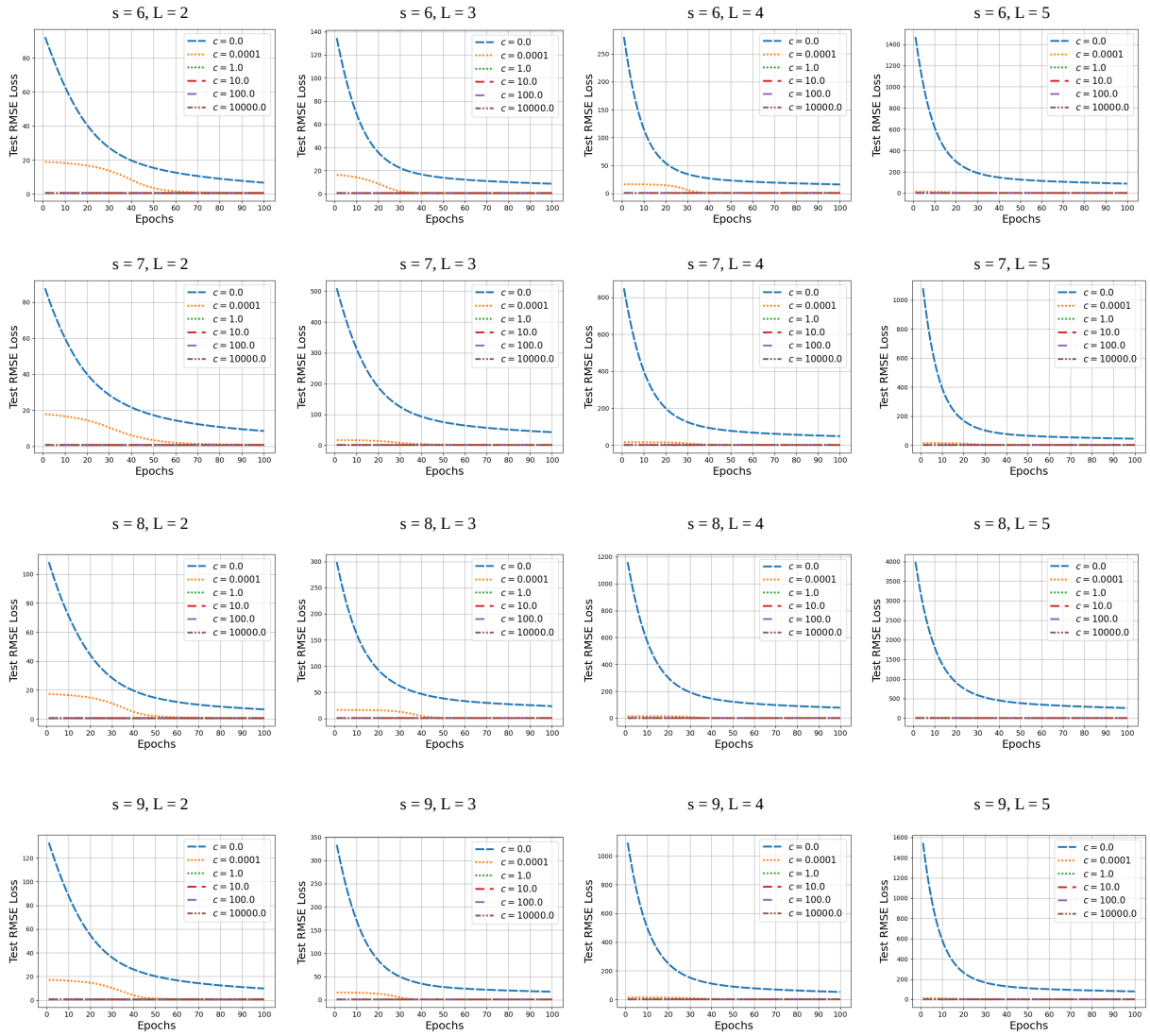


Figure 5: Various experiments were performed on the Superconductivity dataset by varying filter length and number of convolutional layers of the eHDCNN architecture.

of (s, L), and vary curvatures of the Poincaré Ball. The test RMSE curves are plotted and all results are presented in Figure 5. It can be observed that the test RMSE slowly decreases during the initial epochs of training of the eHDCNN. If we increase the number of layers or the length of the input filter, the respective error rates seem to be more stable and converge faster for the eHDCNN. This emphasizes the stability of our proposed architecture during training and is a clear indication of the fact that it requires a much lesser number of training iterations compared to the conventional eDCNN architecture for convergence.

9 Conclusion & Future Works

In this paper, we have identified the limitations of Euclidean spaces in providing meaningful information for training conventional DCNNs. We demonstrated the superiority of hyperbolic convolutions by treating the output of each layer as elements of the Poincaré Ball, projecting them onto the Tangent Space for expansive convolution, and then mapping them back to a higher-dimensional Poincaré Ball to capture complex

hierarchical structures to the next layer. Our primary contribution is the proof of universal consistency by defining regression and error estimators in the hyperbolic space, drawing an analogy to Euclidean space. This is the first known result to explore the statistical consistency of architectures developed beyond the Euclidean domain. Furthermore, our simulation results validate our theoretical justification, showing why eHDCNN is more adept at capturing complex representations, as noted in Remark 8. We anticipate that our findings will significantly accelerate the growth of deep learning spanning across the hyperbolic regime.

References

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*, pp. 4453–4462, 2022.

Ahmad Bdeir, Kristian Schwethelm, and Niels Landwehr. Fully hyperbolic convolutional neural networks for computer vision. *arXiv preprint arXiv:2303.15919*, 2023.

Yankai Chen, Menglin Yang, Yingxue Zhang, Mengchen Zhao, Ziqiao Meng, Jianye Hao, and Irwin King. Modeling scale-free graphs with hyperbolic geometry for knowledge-aware recommendation. In *Proceedings of the fifteenth ACM international conference on web search and data mining*, pp. 94–102, 2022.

Hanane Djeddal, Liticia Touzari, Anastasios Giovanidis, Chi-Dung Phung, and Stefano Secci. Hyperbolic k-means for traffic-aware clustering in cloud and virtualized rans. *Computer Communications*, 176:258–271, 2021.

M.P. do Carmo. *Riemannian Geometry*. Mathematics (Boston, Mass.). Birkhäuser, 1992. ISBN 9783764334901. URL `https://books.google.co.in/books?id=uXJQQgAACAAJ`.

Xiran Fan, Chun-Hao Yang, and Baba C Vemuri. Nested hyperbolic spaces for dimensionality reduction and hyperbolic nn design. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 356–365, 2022.

Kun Fu, Zhuo Chen, Yue Zhang, and Xian Sun. Enhanced feature representation in detection for optical remote sensing images. *Remote Sensing*, 11(18):2095, 2019.

Octavian Ganea, Gary Becigneul, and Thomas Hofmann. Hyperbolic neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL `https://proceedings.neurips.cc/paper_files/paper/2018/file/dbab2adc8f9d078009ee3fa810bea142-Paper.pdf`.

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*, pp. 11–20, 2022a.

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*, pp. 11–20, 2022b.

L. Györfi, M. Kohler, A. Krzyzak, and H. Walk. *A Distribution-Free Theory of Nonparametric Regression*. Springer Series in Statistics. Springer New York, 2002. ISBN 9780387954417. URL `https://books.google.co.in/books?id=0vmb9oGBlo0C`.

Kam Hamidieh. A data-driven statistical model for predicting the critical temperature of a superconductor. *Computational Materials Science*, 2018. URL `https://api.semanticscholar.org/CorpusID:55069173`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Sergey Ioffe. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Valentin 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*, pp. 6418–6428, 2020.

Jennifer R Kwapisz, Gary M Weiss, and Samuel A Moore. Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, 12(2):74–82, 2011.

Serge Lang. *Differential and Riemannian manifolds*. Springer Science & Business Media, 1995.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

John M Lee. *Riemannian manifolds: an introduction to curvature*, volume 176. Springer Science & Business Media, 2006.

Yicong Li, Hongxu Chen, Xiangguo Sun, Zhenchao Sun, Lin Li, Lizhen Cui, Philip S. Yu, and Guandong Xu. Hyperbolic hypergraphs for sequential recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, pp. 988–997, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450384469. doi: 10.1145/3459637.3482351. URL `https://doi.org/10.1145/3459637.3482351`.

Fangfei Lin, Bing Bai, Kun Bai, Yazhou Ren, Peng Zhao, and Zenglin Xu. Contrastive multi-view hyperbolic hierarchical clustering. *arXiv preprint arXiv:2205.02618*, 2022a.

Shao-Bo Lin, Kaidong Wang, Yao Wang, and Ding-Xuan Zhou. Universal consistency of deep convolutional neural networks. *IEEE Transactions on Information Theory*, 68(7):4610–4617, 2022b.

Shao-Bo Lin, Kaidong Wang, Yao Wang, and Ding-Xuan Zhou. Universal consistency of deep convolutional neural networks. *IEEE Transactions on Information Theory*, 68(7):4610–4617, 2022c.

Teng Long and Nanne van Noord. Cross-modal scalable hyperbolic hierarchical clustering. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 16655–16664, 2023.

Aaron Lou, Isay Katsman, Qingxuan Jiang, Serge Belongie, Ser-Nam Lim, and Christopher De Sa. Differentiating through the fréchet mean. In *International conference on machine learning*, pp. 6393–6403. PMLR, 2020.

Emile Mathieu, Charline Le Lan, Chris J Maddison, Ryota Tomioka, and Yee Whye Teh. Continuous hierarchical representations with poincaré variational auto-encoders. *Advances in neural information processing systems*, 32, 2019.

Hossein Mehdipour, Erfan Amini, Seyed Taghi Omid Naeeni, Mehdi Neshat, and Amir H Gandomi. Optimization of power take-off system settings and regional site selection procedure for a wave energy converter. *Energy Conversion and Management: X*, 22:100559, 2024.

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.

Yoshihiro Nagano, Shoichiro Yamaguchi, Yasuhiro Fujita, and Masanori Koyama. A wrapped normal distribution on hyperbolic space for gradient-based learning. In *International Conference on Machine Learning*, pp. 4693–4702. PMLR, 2019.

Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. *Advances in neural information processing systems*, 30, 2017a.

Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. *Advances in neural information processing systems*, 30, 2017b.

Maximillian Nickel and Douwe Kiela. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *International conference on machine learning*, pp. 3779–3788. PMLR, 2018.

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

Frederic Sala, Chris De Sa, Albert Gu, and Christopher Ré. Representation tradeoffs for hyperbolic embeddings. In *International conference on machine learning*, pp. 4460–4469. PMLR, 2018.

Rik Sarkar. Low distortion delaunay embedding of trees in hyperbolic plane. In *International symposium on graph drawing*, pp. 355–366. Springer, 2011.

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

K Simonyan. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

I Sonata, Y Heryadi, L Lukas, and A Wibowo. Autonomous car using cnn deep learning algorithm. In *Journal of Physics: Conference Series*, volume 1869, pp. 012071. IOP Publishing, 2021.

Jianing Sun, Zhaoyue Cheng, Saba Zuberi, Felipe Pérez, and Maksims Volkovs. Hgcf: Hyperbolic graph convolution networks for collaborative filtering. In *Proceedings of the Web Conference 2021*, pp. 593–601, 2021.

Loring W Tu. *Differential geometry: connections, curvature, and characteristic classes*, volume 275. Springer, 2017.

Abraham Ungar. *A gyrovector space approach to hyperbolic geometry*. Springer Nature, 2022.

Jussi Väisälä. Gromov hyperbolic spaces. *Expositiones Mathematicae*, 23(3):187–231, 2005.

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.

J Vermeer. A geometric interpretation of ungar's addition and of gyration in the hyperbolic plane. *Topology and its Applications*, 152(3):226–242, 2005.

Hao Wang, Defu Lian, Hanghang Tong, Qi Liu, Zhenya Huang, and Enhong Chen. Hypersorec: Exploiting hyperbolic user and item representations with multiple aspects for social-aware recommendation. *ACM Transactions on Information Systems (TOIS)*, 40(2):1–28, 2021.

Yijia Wang and Qiaotong Zhao. House price prediction based on machine learning: A case of king county. In *2022 7th International Conference on Financial Innovation and Economic Development (ICFIED 2022)*, pp. 1547–1555. Atlantis Press, 2022.

Jiudong Yang and Jianping Li. Application of deep convolution neural network. In *2017 14th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, pp. 229–232. IEEE, 2017.

Menglin Yang, Min Zhou, Jiahong Liu, Defu Lian, and Irwin King. Hrcf: Enhancing collaborative filtering via hyperbolic geometric regularization. In *Proceedings of the ACM Web Conference 2022*, pp. 2462–2471, 2022.

Ding-Xuan Zhou. Universality of deep convolutional neural networks. *Applied and computational harmonic analysis*, 48(2):787–794, 2020.

A Appendix: Proofs

We provide the detailed proofs and derivations of the Lemmas and Theorem presented in Section 6.

Lemma 1

The Hyperbolic Regression Function (HRF) $f_\rho(x) := \int_{\mathcal{Y}} \log_0^c(y) d\rho(y|x)$, defined by the means of conditional distribution $\rho(\cdot|x)$ of ρ at $x \in \mathcal{X}$ minimizes the *Hyperbolic Generalization Error (HGE).*

Proof. The HGE can be written in terms of conditional expectation in the following way:

$$\begin{aligned}\mathcal{E}(f) &= \int_{\mathcal{Z}} (f(x) - \log_0^c(y))^2 d\rho \\ &= \mathbb{E}_{\mathcal{X},\mathcal{Y}}[f(\mathcal{X}) - \log_0^c(\mathcal{Y})]^2\end{aligned}$$

Now for any function $g : \mathcal{X} \to \mathbb{R}^1$, we write

$$\begin{aligned}\mathcal{E}(g) =& \mathbb{E}_{\mathcal{X}} \left[\mathbb{E}_{\mathcal{Y}|\mathcal{X}} \left[(g(\mathcal{X}) - \mathbb{E}[\log_0^c(\mathcal{Y})|\mathcal{X}] + \mathbb{E}[\log_0^c(\mathcal{Y})|\mathcal{X}] - \log_0^c(\mathcal{Y}))^2 |\mathcal{X}\right]\right] \\ =& \mathbb{E}_{\mathcal{X}} \left[\mathbb{E}_{\mathcal{Y}|\mathcal{X}} \left[(g(\mathcal{X}) - \mathbb{E}[\log_0^c(\mathcal{Y})|\mathcal{X}])^2 |\mathcal{X}\right]\right] + \mathbb{E}_{\mathcal{X}} \left[\mathbb{E}_{\mathcal{Y}|\mathcal{X}} \left[(\mathbb{E}[\log_0^c(\mathcal{Y})|\mathcal{X}] - \log_0^c(\mathcal{Y})|\mathcal{X})^2 |\mathcal{X}\right]\right] \\ &+ 2\mathbb{E}_{\mathcal{X}} \left[\mathbb{E}_{\mathcal{Y}|\mathcal{X}} \left[(g(\mathcal{X}) - \mathbb{E}[\log_0^c(\mathcal{Y})|\mathcal{X}]) (\mathbb{E}[\log_0^c(\mathcal{Y})|\mathcal{X}] - \log_0^c(\mathcal{Y})) |\mathcal{X}\right]\right].\end{aligned}$$

The cross term in the last expression is 0, since

$$\mathbb{E}_{\mathcal{X}} \left[\mathbb{E}_{\mathcal{Y}|\mathcal{X}} \left[(\mathbb{E}[\log_0^c(\mathcal{Y})|\mathcal{X}] - \log_0^c(\mathcal{Y}))\right]\right] = 0.$$

Therefore, the expression for HGE is reduced to

$$\mathcal{E}(g) = \mathbb{E}_{\mathcal{X}} \left[\mathbb{E}_{\mathcal{Y}|\mathcal{X}} \left[(g(\mathcal{X}) - \mathbb{E}[\log_0^c(\mathcal{Y})|\mathcal{X}])^2 |\mathcal{X}\right]\right] + \mathbb{E}_{\mathcal{X}} \left[\mathbb{E}_{\mathcal{Y}|\mathcal{X}} \left[(\mathbb{E}[\log_0^c(\mathcal{Y})|\mathcal{X}] - \log_0^c(\mathcal{Y})|\mathcal{X})^2 |\mathcal{X}\right]\right],$$

which attains minimum when $g(x) = \mathbb{E}\left[\log_0^c(\mathcal{Y})|x\right]$ for each $x \in \mathcal{X}$. Alternately, we write for each $x \in \mathcal{X}$

$$g(x) = \int_{\mathcal{Y}} \log_0^c(y) d\rho(y|x).$$

□

Lemma 2

For any $f : \mathbb{D}_c^n \to \mathbb{R}^1$, we have

$$\mathcal{E}(f) - \mathcal{E}(f_\rho) = \|f - f\rho\|_{L^2_{\rho_{\mathcal{X}}}},$$

where $\rho_{\mathcal{X}}(x) := \int_{\mathcal{Y}} \rho(x, y) d\mathcal{Y}(y)$, for each $x \in \mathcal{X}$, the marginal distribution of ρ on $\mathcal{X}$.

Proof. Following the proof of Lemma 1, we can write

$$\begin{aligned}\mathcal{E}(f) - \mathcal{E}(f_\rho) &= \mathbb{E}_{\mathcal{X}} \left[\mathbb{E}_{\mathcal{Y}|\mathcal{X}} \left[(f(x) - \mathbb{E}[\log_0^c(\mathcal{Y})|\mathcal{X}])^2 |\mathcal{X}\right]\right] \\ &= \mathbb{E}_{\mathcal{X},\mathcal{Y}}[\left[(f(x) - \mathbb{E}[\log_0^c(\mathcal{Y})|\mathcal{X}])^2 |\mathcal{X}\right] \\ &= \int_{\mathcal{X}} \int_{\mathcal{Y}} (f(x) - f_\rho(x))\rho(x, y) d\mathcal{X}(x) d\mathcal{Y}(y) \\ &= \int_{\mathcal{X}} (f(x) - f_\rho(x))^2 \int_{\mathcal{Y}} \rho(x, y) d\mathcal{Y}(y) d\mathcal{X}(x) \\ &= \int_{\mathcal{X}} (f(x) - f_\rho(x))^2 \rho_{\mathcal{X}}(x) d\mathcal{X}(x) \\ &= \|f - f_\rho\|_{L^2_{\rho_{\mathcal{X}}}}.\end{aligned}$$

□

Lemma 3

Let $\mathcal{G}$ be a class of functions from $\mathcal{X} \to \mathbb{R}$ and ν be a probability measure on $\mathcal{X}$. For $p \geq 0$ and $\epsilon > 0$, we have

$$\mathcal{M}(2\epsilon, \mathcal{G}, \|\cdot\|_{L^p(\nu)}) \leq \mathcal{N}(\epsilon, \mathcal{G}, \|\cdot\|_{L^p(\nu)}) \leq \mathcal{M}(\epsilon, \mathcal{G}, \|\cdot\|_{L^p(\nu)}).$$

In particular,

$$\mathcal{M}_p(2\epsilon, \mathcal{G}, x_1^m) \leq \mathcal{N}_p(\epsilon, \mathcal{G}, x_1^m) \leq \mathcal{M}_p(\epsilon, \mathcal{G}, x_1^m).$$

Proof. The same proof mentioned in Lemma 9.2 Györfi et al. (2002), can be applied to any general metric space M instead of $\mathbb{R}^d$. In particular M can be $\mathcal{X}$. This shows the lemma is unaltered in the case of a compact subset in a hyperbolic space. □

Lemma 6

When $m^{-\theta} M_m^2 \left[1 + \frac{1}{M_m\sqrt{c}} \tanh^{-1}(M_m\sqrt{c})\right]^2 \to 0$ and 10 holds for $\theta \in (0, 1/2)$, then we have

$$\lim_{m\to\infty} \mathcal{E}_{\pi_{M_m}}(\pi_{M_m} f_{D,L,s}) - \mathcal{E}_{\pi_{M_m},D}(\pi_{M_m} f_{D,L,s}) = 0$$

holds almost surely.

Proof. We have $|\pi_M f, D, L, s| \leq M$ and $|\log_0^c(y_M)|, |\log_0^c(y_{i,M})| \leq \frac{1}{\sqrt{c}} \tanh^{-1}(M\sqrt{c})$ [The last two inequalities follow from the fact that $\tanh^{-1}$ is increasing on $(-1, 1)$]. A little computation will show that

$$|\mathcal{E}_{\pi_M}(\pi_M f_{D,L,s})| \leq M^2 \left[1 + \frac{1}{M\sqrt{c}} \tanh^{-1}(M\sqrt{c})\right]^2.$$

This leads us to derive that

$$|\mathcal{E}_{\pi_M}(\pi_M f_{D,L,s}) - \mathcal{E}_{\pi_M,D}(\pi_M f_{D,L,s})| \leq 2M^2 \left[1 + \frac{1}{M\sqrt{c}} \tanh^{-1}(M\sqrt{c})\right]^2.$$

Putting $\alpha = \beta = 1$, in Lemma 5 and $\epsilon = m^{-\theta}$ we get that

$$\begin{aligned}\mathcal{E}_{\pi_M}(\pi_M f_{D,L,s}) - \mathcal{E}_{\pi_M}(f_\rho) - \left(\mathcal{E}_{\pi_{M,D}}(\pi_M f_{D,L,s}) - \mathcal{E}_{\pi_{M,D}}(f_\rho)\right) \\ \leq 2m^{-\theta}\left[1 + M^2\left(1 + \frac{1}{M\sqrt{c}} \tanh^{-1}(M\sqrt{c})\right)\right]^2\end{aligned}$$

holds with probability at least

$$1 - 14 \sup_{x_1^m \in \mathcal{X}^m} \mathcal{N}_1\left(\frac{1}{20\frac{1}{\sqrt{c}}\tanh^{-1}(M\sqrt{c})m^\theta}, \mathcal{F}, x_1^m\right) \exp\left(-\frac{m^{1-2\theta}}{428(1+\epsilon)\left(\frac{1}{\sqrt{c}}\tanh^{-1}(M\sqrt{c})\right)^4}\right)$$

From lemma Lemma 4 we write

$$\begin{aligned}&\sup_{x_1^m \in \mathcal{X}^m} \mathcal{N}_1\left(\frac{1}{20\frac{1}{\sqrt{c}}\tanh^{-1}(M\sqrt{c})m^\theta}, \mathcal{F}, x_1^m\right) \exp\left(-\frac{m^{1-2\theta}}{428(1+\epsilon)\left(\frac{1}{\sqrt{c}}\tanh^{-1}(M\sqrt{c})\right)^4}\right) \\ &\leq \exp\left(c^* \log\left(20\left(\frac{1}{\sqrt{c}}\tanh^{-1}(M\sqrt{c})\right)^2 m^\theta\right) L_m^2(d + sL_m)\log(L_m(s+d)) - c'\right),\end{aligned}$$

where $c' = \frac{m^{1-2\theta}}{428\left(\frac{1}{\sqrt{c}}\tanh^{-1}(M\sqrt{c})\right)^4}$. The conditions of Theorem 1 indicates that the RHS of the last inequality goes to 0 as $m \to \infty$. Combining together with the strong law of large numbers we get

$$\begin{aligned}\mathcal{E}_{\pi_{M_m}}\left(\pi_{M_m} f_{D,L_m,s}\right) - \mathcal{E}_{\pi_{M_m},D}\left(\pi_{M_m} f_{D,L_m,s}\right) &\leq 2m^{-\theta}\left[1 + M\left(1 + \frac{1}{M\sqrt{c}}\tanh^{-1}(M\sqrt{c})\right)\right]^2 \\ &\leq 8M^2 m^{-\theta} \to 0\end{aligned}$$

as $m \to \infty$ holds almost surely, completing the proof of Lemma 6. □

Lemma 7

Let $\Omega \subseteq \mathbb{D}_c^d$ be compact and $2 \leq s \leq d$. Then for any $f \in \mathcal{C}(\Omega)$, there exist a sequence of filters $\boldsymbol{w}$ and bias vectors $\boldsymbol{b}$ of appropriate dimensions and $f_L^{\boldsymbol{w},\boldsymbol{b}} \in \mathcal{H}_{L,s}$ such that

$$\lim_{L\to\infty} \|f - f_L^{\boldsymbol{w},\boldsymbol{b}}\|_{\mathcal{C}(\Omega)} = 0.$$

Proof. Define $g(y) := f(\exp_0^c(y))$ for $y \in \log_0^c(\Omega)$. Then, by Theorem 1 Zhou (2020), we know that there exists $g_L^{\boldsymbol{w},\boldsymbol{b}}$ [where $g_L^{\boldsymbol{w},\boldsymbol{b}}$ lies in the free parameter space of the DCNN], such that

$$\lim_{L\to\infty} \|g - g_L^{\boldsymbol{w},\boldsymbol{b}}\|_{\mathcal{C}(\log_0^c(\Omega))} = 0.$$

We now define $f_L^{\boldsymbol{w},\boldsymbol{b}}(x) := g_L^{\boldsymbol{w},\boldsymbol{b}}(\log_0^c(x))$ for $x \in \mathbb{D}_c^d$. Now it is easy to verify that

$$\lim_{L\to\infty} \|f - f_L^{\boldsymbol{w},\boldsymbol{b}}\|_{\mathcal{C}(\Omega)} = \lim_{L\to\infty} \|g \circ \log_0^c - g_L^{\boldsymbol{w},\boldsymbol{b}} \circ \log_0^c\|_{\mathcal{C}(\log_0^c(\Omega))} = 0,$$

since the $\log_0^c$ [hence its inverse $\exp_0^c$] is global diffemorphism from $\mathbb{D}_c^d \to \mathbb{R}^d$ [from $\mathbb{R}^d \to \mathbb{D}_c^d$]. □

Theorem 1

Suppose $L = L_m \to \infty$, $M = M_m \to \frac{1}{\sqrt{c}}$, $m^{-\theta} M_m^2 \left[1 + \frac{1}{M_m\sqrt{c}}\tanh^{-1}(M_m\sqrt{c})\right]^2 \to 0$ [constrained truncation on the power of sample size] and

$$\frac{\left(\frac{1}{\sqrt{c}}\tanh^{-1}(M_m\sqrt{c})\right)^4 L_m^2(L_m + d)\log(L_m)}{m^{1-2\theta}} \times \log\left(\left(\frac{1}{\sqrt{c}}\tanh^{-1}(M_m\sqrt{c})\right) m\right) \to 0,$$

(10) hold for $\theta \in (0, 1/2)$ and input filter length as $2 \leq s \leq d$. Then $\pi_{M_m} f_{D,L_m,s}$ is strongly universally consistent, where $\pi_M(l) := \min\{M, |l|\} \cdot sign(l)$ is the well-known truncation operator.

Proof. Since $M_m \to \frac{1}{\sqrt{c}}$, we have $M_m \times \left(\frac{1}{M_m\sqrt{c}}\tanh^{-1}(M_m\sqrt{c})\right) \to \infty$ as $m \to \infty$. We also have $\mathbb{E}[(\log_0^c(y))^2] < \infty$, i.e. $f_\rho \in L^2(\rho_{\mathcal{X}})$. By Lemma 7, we say that there exists a big enough L_ϵ so that $f_{L_\epsilon}^{w,b} \in \mathcal{H}_{L_\epsilon,s}$ with

$$\|f_\rho - f_{L_\epsilon}^{w,b}\|_{L^2(\rho_x x)}^2 \leq \left[\limsup_{x\in\mathcal{X}} \|f_\rho(x) - f_{L_\epsilon}^{w,b}(x)\|\right]^2 = \left[\|f_\rho - f_{L_\epsilon}^{w,b}\|_{\mathcal{C}(\mathcal{X})}\right]^2 \leq \epsilon,$$

where the second inequality follows from the fact that $\rho_{\mathcal{X}}$ being a Borel Probability measure on $\mathcal{X}$.

By triangle inequality, we write

$$\begin{aligned}\mathcal{E}(\pi_M(f_{D,L,s})) - \mathcal{E}(f_\rho) \leq & (\epsilon\mathcal{E}(\pi_M(f_{D,L,s})) - (1+\epsilon)\mathcal{E}(\pi_M(f_{D,L,s}))) \\ & + (1+\epsilon)\left(\mathcal{E}_{\pi_M}(\pi_M(f_{D,L,s}))) - \mathcal{E}_{\pi_M,D}(\pi_M(f_{D,L,s}))\right. \\ & + (1+\epsilon)\left(\mathcal{E}_{\pi_M,D}(\pi_M(f_{D,L,s})) - \mathcal{E}_{\pi_M,D}(f_{D,L,s})\right) \\ & + (1+\epsilon)(\mathcal{E}_{\pi_M,D}(f_{D,L,s})) - (1+\epsilon)^2(\mathcal{E}_D(f_{D,L,s})) \\ & + (1+\epsilon)^2\left(\mathcal{E}_D(f_{D,L,s}) - \mathcal{E}_D(f^{w,b}_{L_\epsilon})\right) + (1+\epsilon)^2\left(\mathcal{E}_D(f^{w,b}_{L_\epsilon}) - \mathcal{E}(f^{w,b}_{L_\epsilon})\right) \\ & + (1+\epsilon)^2\left(\mathcal{E}(f^{w,b}_{L_\epsilon}) - \mathcal{E}(f_\rho)\right) + \left((1+\epsilon)^2 - 1\right)\mathcal{E}(f_\rho) \\ = & \sum_{i=1}^{8} B_i.\end{aligned}$$

We will use an inequality, which we will require throughout the rest of the steps:

$$(s+t)^2 \leq (1+\epsilon)s^2 + (1+1/\epsilon)t^2 \tag{11}$$

for $s, t, \epsilon > 0$.

We will bound each of the B_i to prove the universal consistency as done in Part 3 of Appendix A in Lin et al. (2022b).

We will start with B_1 as,

$$\begin{aligned}B_1 &= \epsilon\mathcal{E}(\pi_M(f_{D,L,s})) - (1+\epsilon)\mathcal{E}(\pi_M(f_{D,L,s})) \\ &= \int_{\mathcal{Z}} |\pi_M(f_{D,L,s}(x)) - (\log_0^c(y_M)) + (\log_0^c(y_M)) - (\log_0^c(y))|^2 d\rho \\ &- (1+\epsilon)\int_{\mathcal{Z}} |\pi_M(f_{D,L,s}) - (\log_0^c(y_M))|^2 d\rho \\ &\leq (1+(1/\epsilon))\int_{\mathcal{Z}} |\log_0^c(y) - \log_0^c(y_M)|^2 d\rho.\end{aligned}$$

But we have $M = M_m \to \frac{1}{\sqrt{c}}$ as $m \to \infty$. Since $\epsilon > 0$ is arbitrary, we get $B_1 \to 0$ as $m \to \infty$.

By Lemma 7 and the constraints in the statement of Theorem 1 we get

$$B_2 \to 0 \text{ as } m \to \infty.$$

By the definition of the truncation operator, we get,

$$B_3 = \frac{1}{m}\sum_{i=1}^{m} |\pi_M(f_{D,L,s}(x_i)) - (\log_0^c(y_{i,M}))|^2 - \frac{1}{m}\sum_{i=1}^{m} |f_{D,L,s}(x_i) - (\log_0^c(y_{i,M}))|^2 \leq 0.$$

By the Strong Law of Large Numbers and inequality 11 we have,

$$\begin{aligned}B_4 &\leq (1+\epsilon)(1+1/\epsilon)\frac{1}{m}\sum_{i=1}^{m} |\log_0^c(y_i) - \log_0^c(y_{i,M})|^2 \\ &\to (1+\epsilon)(1+1/\epsilon)\int_{\ddagger} |\log_0^c(y) - \log_0^c(y_M)|^2 d\rho\end{aligned}$$

as $m \to \infty$ almost surely. By the fact that $M_m \to \frac{1}{\sqrt{c}}$ as $m \to \infty$, we get

$$B_4 \to 0.$$

Since $f_{D,L}$ is the estimator of Emperical Risk Minimizer, we obtain

$$B_5 = (1+\epsilon)^2 \left(\frac{1}{m} \sum_{i=1}^{m} |f_{D,L}(x_i) - \log_0^c(y_i)|^2 - \frac{1}{m} \sum_{i=1}^{m} |f_{L_\epsilon}^{w,b}(x_i) - \log_0^c(y_i)|^2 \right) \leq 0.$$

Again by the Strong Law of Large Numbers, we have

$$B_6 \to 0$$

almost surely.

For B_7 we have

$$B_7 = (1+\epsilon)^2 \|f_{L_\epsilon} - f\rho\|_{L^2_{\rho_{\mathcal{X}}}}^2.$$

By Lemma 7, we get

$$B_7 \leq (1+\epsilon)^2 \epsilon.$$

Also, we have

$$B_8 \leq ((1+\epsilon)^2 - 1) \int_{\mathcal{Z}} |f_\rho(x) - \log_0^c(y)|^2 d\rho = \epsilon(\epsilon+2) \int_{\mathcal{Z}} |f_\rho(x) - \log_0^c(y)|^2 d\rho.$$

Summing up all the terms from B_1 to B_8, we get

$$\limsup_{m\to\infty} \mathcal{E}(\pi_M(f_{D,L,s})) - \mathcal{E}(f_\rho) \leq (1+\epsilon)^2\epsilon + \epsilon(2+\epsilon) \int_{\mathcal{Z}} |f_\rho(x) - \log_0^c(y)|^2 d\rho$$

holds almost surely. As $\epsilon > 0$ is arbitrary, we can write

$$\limsup_{m\to\infty} \mathcal{E}(\pi_M(f_{D,L,s})) - \mathcal{E}(f_\rho) = 0.$$

This completes the proof of the universal consistency of eHDCNN. □