INDIAN STATISTICAL INSTITUTE

MASTERS THESIS

Universally Consistent Hyperbolic Deep Neural Networks

Author:
Sagar GHOSH

Supervisor:
Dr. Swagatam DAS

A thesis submitted in partial fulfillment to the requirements for the degree of ***Master of Technology*** *(****Computer Science****)*

in the

Computer and Communication Sciences Division

June 1, 2025

Declaration of Authorship

I, Sagar GHOSH, declare that this thesis titled, "Universally Consistent Hyperbolic Deep Neural Networks" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this Institute.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: Sagar Ghosh

Date: 01.06.2025

CERTIFICATE

This is to certify that the dissertation entitled **"Universally Consistent Hyperbolic Deep Neural Networks"** submitted by **Mr. Sagar Ghosh** to Indian Statistical Institute, Kolkata, in partial fulfillment to the requirements for the degree of **Master of Technology in Computer Science** is a bona fide record of work carried out by him under my supervision and guidance. The dissertation has fulfilled all the requirements as per the regulations of this institute and, in my opinion, has reached the standard needed for submission.

Dr. Swagatam Das
Professor,
Electronics and Communication Sciences Unit,
Indian Statistical Institute,
Kolkata-700108, INDIA.

"... What we mean by words body, soul, mind: I don't fathom, but I shall always observe the universe quietly, without words, the current of the cosmos's awareness flows towards you "

"Naibedya": ***Rabindranath Tagore***

Acknowledgements

I am profoundly grateful to everyone who has extended their support and guidance throughout my M.Tech. journey, playing a vital role in both my academic growth and personal well-being. Though words may fall short in conveying the depth of my appreciation, I will do my best to express it here.

First and foremost, I would like to express my heartfelt gratitude to my advisor, Prof. Swagatam Das, whose unwavering support and insightful mentorship have been instrumental in shaping my research interests and honing my critical thinking skills. His patience, kindness, and encouragement—especially during times of limited progress or uncertainty—have been invaluable and deeply motivating.

I am also thankful to the faculty members of the institute for their quiet yet impactful presence, whose lasting influence has helped shape my academic journey in subtle but meaningful ways.

I feel fortunate to have had the opportunity to regularly collaborate with Mr. Kushal Bose, Senior Research Fellow in the Electronics and Communication Sciences Unit. Our discussions significantly deepened my understanding and broadened my perspective in this field. I also wish to thank the members of the Machine Learning Research Group—your regular interactions, insights, and camaraderie have been an integral part of my learning experience.

Above all, I extend my sincere gratitude to my parents, whose unwavering support has been the foundation of all that I have achieved. I also fondly remember my grandparents and their enduring love, which continues to inspire and guide me.

Abstract

Universally Consistent Hyperbolic Deep Neural Networks

by Sagar GHOSH

The ubiquitous pertinence of Deep Neural Networks has made it pivotal in modern Computer Science Applications, ranging from Computer Vision to Pattern Recognition and Machine Translation. Although these deep architectures are primarily based on Euclidean Spaces, Hyperbolic Neural Networks (HNN) gained traction in recent times to tackle more complex non-Euclidean data having inherent hierarchical structures. These HNN architectures have shown commendable improvements in test results on tree or graph-like data by exploiting the inherent exponential metric distances of hyperbolic spaces, making them more suitable to embed non-Euclidean data. Although HNNs surpass their conventional Euclidean counterparts by commendable margins, little to no theory is known behind their surprising results. We try to bridge this gap in this thesis by analyzing the universal consistency results on two types of HNNs: one is Convolutional and the other one is Transformer.

We start by defining the corresponding architectures on hyperbolic spaces, suitable for our theoretical analyses. We will also define several statistical terminologies in this context, required for our theoretical justification. The primary contribution of this work is the introduction of universal consistency of HNNs. Universal consistency refers to the ability of a model to approximate any target function within a given class, under the assumption that the model has sufficient complexity (capacity). The thesis rigorously proves that hyperbolic neural networks, when trained under sufficient regularity constraints, can achieve universal consistency, ensuring they are capable of learning complex relationships across a broad range of tasks.

Additionally, we present empirical results showcasing the superior performance of HNNs on benchmark datasets, including those with hierarchical or non-Euclidean structures. These results highlight the potential of hyperbolic neural networks to outperform traditional Euclidean-based models in tasks such as graph classification and representation learning.

In conclusion, this thesis establishes the universal consistency of two hyperbolic neural networks, providing a powerful framework for tackling a wide variety of machine learning problems, particularly those involving data with complex, hierarchical, or non-Euclidean relationships, paving the way for future research and applications in this area.

Contents

List of Figures

List of Tables

List of Symbols

$\mathbb{R}^d$	Euclidean Space of dimension d
$\mathbb{D}_c^d$	Poincaré Ball in dimension d with curvature -c ($c > 0$)
$T_p(\mathbb{D}_c^d)$	Tangent Space at $p \in \mathbb{D}_c^d$
$\exp_x^c(y)$	Exponential map on $\mathbb{R}^d$ starting at $x \in \mathbb{D}_c^d$, projecting $y \in T_x(\mathbb{D}_c^d)$
$\log_y^c(x)$	Logarithm Map on $\mathbb{D}_c^d$ starting at $y \in T_x(\mathbb{D}_c^d)$, projecting $x \in \mathbb{D}_c^d$
$\oplus_c$	Möbius Addition on Poincaré Disc with curvature -c
$\otimes_c$	Möbius Scaler Multiplication on Poincaré Disc with curvature -c
$*$	Euclidean Expansive Convlution: $(\boldsymbol{w} * \boldsymbol{v'}) = \sum_{l=1}^{n} w_{j-l} v'_l$ for $j = 1, 2, ..., n+s$
$\star$	Euclidean Contractive Convolution: $\boldsymbol{w} \star \boldsymbol{v'} = \sum_{l=j-s}^{j} w_{j-l} v_l, j = s+1, ..., n$
$*_h$	Hyperbolic Expansive Convolution: $\boldsymbol{w} *_h \boldsymbol{v} := \exp_0^c(\boldsymbol{w} * \log_0^c(\boldsymbol{v}))$
$\star_h$	Hyperbolic Contractive Convolution:$\boldsymbol{w} *_h \boldsymbol{v} := \exp_0^c(\boldsymbol{w} \star \log_0^c(\boldsymbol{v}))$
h_L	$\boldsymbol{h}_L(x) := \{h^1(x), h^2(x), ..., h^{n_L}(x)\} \in \mathbb{D}_c^{n_L}$
$\boldsymbol{h}_k$	$\boldsymbol{h}_k(x) = \sigma(\boldsymbol{w}_k \circ \boldsymbol{h}_{k-1}(x) \oplus_c \exp_0^c(\boldsymbol{b}_k))$
h_L	$h_L(x) = \exp_0^c\left[\boldsymbol{a}_L \cdot \log_0^c(\boldsymbol{h}_L(x))\right]$
$\mathcal{E}_D(f)$	$\frac{1}{m}\sum_{i=1}^{m}\left(f(x_i) - \log_0^c(y_i)\right)^2$
$f_{\mathcal{D},L,s}$	$\arg\min_{f \in \mathcal{H}_{L,s}} \mathcal{E}_{\mathcal{D}}(f)$
$\mathcal{H}_{L,s}$	$\{h_L(x), \boldsymbol{w}_k, \boldsymbol{b}_k \in \mathbb{R}^{d+ks}, k = 1, 2, ..., L\}$
$t_{\mathcal{H}}^{h,s,r}$	A single transformer block with number of heads h, head size s, hidden dim r
$\mathcal{T}_{\mathcal{H}}^{h,s,r}$	Class of continuous functions represented by $t_{\mathcal{H}}^{h,s,r}$ block
$t_{\mathcal{P},\mathcal{H}}^{h,s,r}$	Positionally encoded $t_{\mathcal{H}}^{h,s,r}$ block
$\mathcal{T}_{\mathcal{P},\mathcal{H}}^{h,s,r}$	Class of continuous functions represented by $t_{\mathcal{P},\mathcal{H}}^{h,s,r}$ block
$\mathcal{E}_{\pi_M,D}(f)$	$\frac{1}{t}\sum_{i=1}^{t}\|f(X_i) - \log_0^c(Y_i)\|^2$
$\mathcal{E}_{\pi_M}(f)$	$\int_{\mathcal{X}\times\mathcal{Y}} \|f(x) - \log_0^c(y)\| d\rho$

Dedicated to my parents

Chapter 1

Introduction

1.1 Objective

In recent years, neural networks have become a cornerstone of modern machine learning, driving advancements across a wide range of fields, from computer vision to natural language processing. Moreover, as data is increasingly becoming more complex and exhibiting non-Euclidean nature, Hyperbolic Neural Networks (HNN) (Ganea, Becigneul, and Hofmann, 2018) have become a popular architecture to process these types of data. Despite their success, there remains a significant gap in understanding the theoretical foundations that guarantee their performance across diverse applications. This thesis explores the concept of universally consistent hyperbolic neural networks, a framework that aims to bridge this gap by providing theoretical guarantees for the consistency of these neural network models. Universal consistency, in this context, refers to the ability of a neural network to approximate any function within a specific class to arbitrary precision, given sufficient complexity and training data. By focusing on the theoretical properties of neural networks, this work seeks to establish a deeper understanding of their capabilities, limitations, and the conditions under which they can be expected to perform reliably. We primarily focus on two important class of neural networks and their hyperbolic variants: one is Convolutional and the second one is Transformer. We begin with a brief introduction to both the networks.

1.2 Convolutional Neural Networks

The ubiquitous utility of Deep Convolutional Neural Networks (DCNNs) (LeCun et al., 1998) dominated the arena of Computer Vision (Yang and 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 focussed 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

$$f \otimes g(z) := \int_{\mathbb{R}^n} f(x)g(z-x)dx, \tag{1.1}$$

where $z \in \mathbb{R}^n$. In the discrete 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

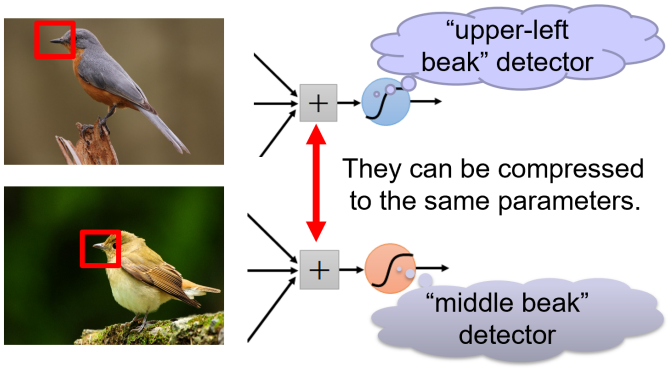


FIGURE 1.1: Visualization of filtering in CNN

forms of equations

$$(\mathbf{w} * \mathbf{v})_k := \sum_{i=1}^{n} w_{k-i} v_i, \;\; k = 1, 2, ..., n+s \tag{1.2}$$

and

$$(\mathbf{w} \star \mathbf{v})_k := \sum_{i=k-s}^{k} w_{k-i} v_i, \;\; k = s+1, s+2, ..., n \tag{1.3}$$

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, 2020a)

$$h_i(x) = r\left(\mathbf{w}_i \circ h_{i-1}(x) + b_i\right), \;\; \text{for} \;\; i = 1, 2, ..., L, \tag{1.4}$$

starting with the input as $h_0(x) = x$ and $\circ$ can be either $*$ or $\star$ as defined by equations 1.2 or 1.3 respectively. The final one-dimensional output of this network is defined as the scalar product between the output produced by the L^{th} layer with a trainable vector a_L of compatible length

$$h_o(x) := a_l \cdot h_L(x). \tag{1.5}$$

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 and Noord, 2023) can be put forth where Euclidean feature space seems unproductive, like datasets containing hierarchical structures.

The embedding learning of hierarchical datasets in the Euclidean spaces raises concerns regarding extracting meaningful information. To address this concern, the research community showed numerous endeavors (Ganea, Becigneul, and Hofmann, 2018; Nickel and Kiela, 2017a; Bdeir, Schwethelm, and Landwehr, 2023) in designing neural networks in non-Euclidean feature spaces. Hyperbolic Neural Networks (HNNs) (Ganea, Becigneul, and Hofmann, 2018) paved the way for generating embeddings in the hyperbolic space equipped with negative curvature. HNNs offer congenial feature space to understand the complex relationships and structural intricacies within the data. In this sobering context, the Hyperbolic Deep Convolutional Neural Networks (HDCNN) (Bdeir, Schwethelm, and Landwehr, 2023) successfully pursued various image-related tasks. Despite the profound impact of hyperbolic neural networks, insightful theoretical analysis is lacking in the literature. This work is motivated to bridge this widened gap by offering an extensive statistical analysis to comprehend the underlying working mechanism of hyperbolic convolution. The consistency analysis of Euclidean convolutional networks has

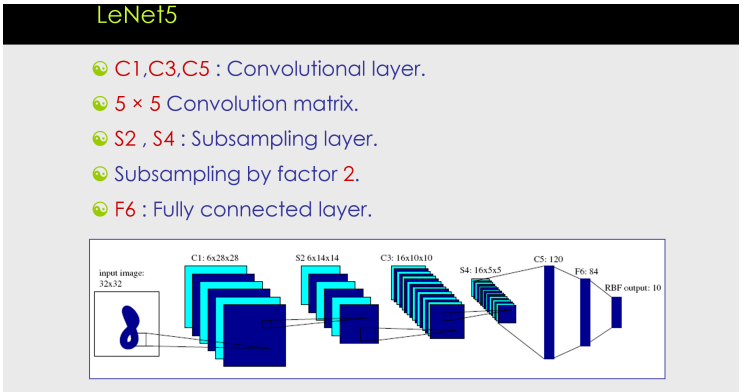


FIGURE 1.2: Typical mathematical workflow of a LeNet 5 CNN Architecture

been addressed in (Lin et al., 2022b), where the authors provided theoretical insights through explicit bounds on packing numbers and error analysis for bounded samples within the Euclidean framework. Their work primarily focuses on expansive convolutional operations. However, a similar analysis for convolution operations in hyperbolic space remains uncharted. In particular, exploring the consistency of expansive convolution in hyperbolic spaces would mark a significant advancement toward a deeper theoretical understanding of hyperbolic convolution.

In response to this gap, we have developed a theoretical foundation for the consistency of 1-D expansive Deep Hyperbolic Deep Convolutional Neural Networks (eHDCNNs). Our proposed architecture offers a hyperbolic generalization of the standard Euclidean DCNN, which is essential for defining statistical concepts and enabling subsequent theoretical analysis. We initially constructed a framework for 1-D expansive convolution on the Poincaré disc, providing a basis for our consistency analysis. Additionally, experiments conducted on synthetic and real-world datasets illustrate the efficiency of representing features in hyperbolic spaces. The results empirically support our theoretical findings, achieving lower error rates in the hyperbolic setting significantly faster than the Euclidean equivalent.

1.3 Transformer

The emergence of self-attention-based Transformer Architecture (Vaswani, 2017) has dominated the area of natural language processing (NLP), computer vision, and speech recognition. The architecture was originally proposed as a sequence-to-sequence model to perform NLP tasks such as machine translation (Wang et al., 2019), language modeling (Bouschery, Blazevic, and Piller, 2023), text generation (Zhang et al., 2023), and question answering (Shao et al., 2019). The feasibility of the Transformer to model an input sequence of tokens to an output sequence of tokens has achieved state-of-the-art performance. Its widespread applications have also advanced into other domains such as audio processing [(Dong, Xu, and Xu, 2018; Gulati et al., 2020)], chemical and biological sciences [(Rives et al., 2021; Schwaller et al., 2019)].

The sequence-to-sequence model of vanilla Transformer architecture consists of two blocks: While the encoder block consists of a self-attention layer and a token-wise feed-forward layer, the decoder block contains an additional masked attention layer. The encoder block is injected with input token embeddings along with a piece of position-encoded information, which is converted to a weighted combination of embeddings for each of the input tokens through the self-attention mechanism. The weighted embeddings are then processed by the feed-forward layer independently. To mitigate the problem of vanishing gradients, each sublayer is equipped with a residual connection (Gu and Feng, 2019) followed by a layer normalization to stabilize the parameters. Additionally, the decoder block deploys a cross-attention mechanism between the masked multi-head attention layers and token-wise feed-forward layers, while the masking is

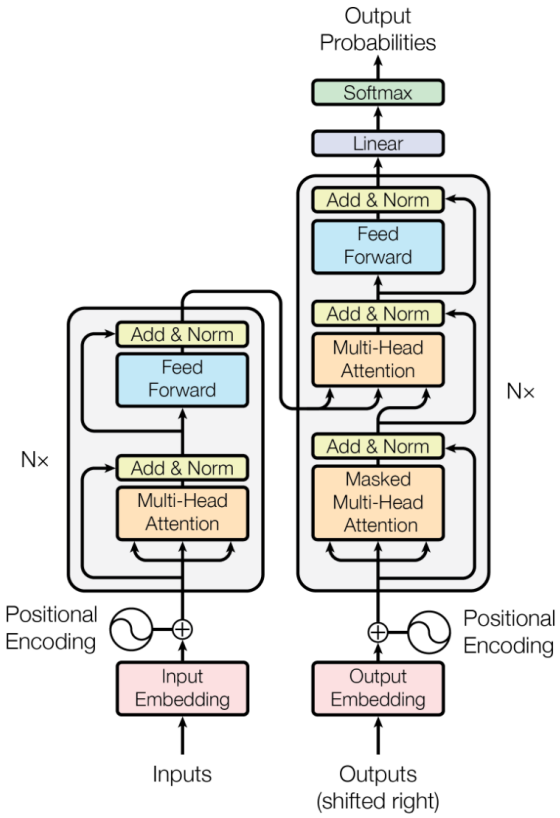


FIGURE 1.3: The Transformer Architecture

done to prevent the tokens from assuming subsequent positions. Here, we present a brief mathematical description of the attention mechanism and the token-wise feedforward mechanism.

Attention Mechanism

In each attention head of the attention layer, we have a trainable triplet of Query (Q), Key (K), and Value (V) matrices. For t number of input tokens $\{X_i\}_{i=1}^{t}$ residing in an input embedding space of dimension d and for the three matrices Q, K and V of compatible sizes, the attention is defined in the form of a dot product/multiplication followed by a component wise softmax operations:

$$\text{Attn}(Q, K, V) := \sigma\left(QK^t\right) V, \tag{1.6}$$

where σ is the component-wise softmax operator. Some representation of the dot product also involves a scaled version, which is performed by dividing the multiplying factor QK^t by $\sqrt{d}$. Here $\sigma\left(QK^t\right)$ is often referred to as the Attention Matrix. Finally, all the outputs of the attention heads are concatenated to produce the final output of the attention layer.

The Transformer Architecture in Figure 1.3 (Vaswani, 2017) mainly uses three types of attention mechanisms: The Encoder block uses a self-attention mechanism to capture the contextual relations of an input token to other tokens present in the input sequence, while the Decoder block uses mask-attention and cross-attention mechanisms for generation purposes. For detailed discussion, we refer to (Lin et al., 2022c).

Feed-Forward Mechanism

The token-wise fully connected feed-forward layer with one hidden layer processes each token independently, followed by a ReLU activation before the processed tokens are passed onto the final layer. The output can be given mathematically as

$$\mathrm{FF}(X) := U_2 \cdot \mathrm{ReLU}\left(U_1 \cdot \mathrm{Attn}(X) + b_1 \mathbb{1}_t^t\right) + b_2 \mathbb{1}_t^t, \tag{1.7}$$

where U_1, U_2 are matrices of appropriate dimension and b_1, b_2 are vectors of dimension t.

Despite the widespread applications of the Transformer Architecture, its theoretical properties are largely unknown. The most well-known result in this area can be attributed to the universal approximation property of the Transformer, which was proved by Yun and others (Yun et al., 2020). This remarkable work demonstrated that a Transformer with 2 attention heads each of size 1 in their attention layer and a hidden layer dimension of size 4 can arbitrarily approximate any sequence to sequence functions with compact support on $\mathbb{R}^{d\times t}$ (t is the number of input tokens and d is their embedding dimension), when the input tokens are injected with their positional information. Their work mathematically formalized the notion of contextual embeddings and proved that the multi-head attention mechanism can effectively compute the contextual embeddings of the input tokens, whereas the value mapping ability of the feed-forward network enables the Transformer Architecture to approximate any permutation equivariant functions, further by injecting the positional information one can remove the restriction of permutation equivariance.

While the universal approximation property of the Transformer is a key step in understanding the functions it can represent, the question of universal consistency remains open. It's also crucial to define what universal consistency means in this context. Given a labeled sequence-to-sequence task, we can define generalization error as the expected output conditioned on the input. Under suitable truncation constraints, the empirical error can be shown to converge to this generalization error as the sample size grows. Previously the issue of universal consistency has been addressed in the context of convolutional neural network in (Lin et al., 2022b), by considering a capacity estimation using a relationship between metric entropy and pseudo-dimension estimate, followed by an error analysis on bounded samples, and finally involving the universal approximation property in the case of a $1-$ dimensional convolution operation. We will modify the lemmas and theorems accordingly and address this issue of universal consistency of the Transformer architecture by suitably developing a regularity condition, ensuring the convergence of the empirical error to the generalization error. Advancing a step further, we propose an analogous hyperbolic version of the entire Transformer Architecture based on Poincaré Disc and prove its universal consistency as well.

1.4 Thesis Organization

Having described these architectures briefly, the rest of the Thesis is organized as follows:

- **Chapter 2:** We will provide a brief literature review
- **Chapter 3:** We discuss all the Mathematical Preliminaries
- **Chapter 4:** Universal Consistency of the Hyperbolic Convolutional neural Network
- **Chapter 5:** Universal Consistency of the Hyperbolic Transformer
- **Chapter 6:** Experimental Results
- **Chapter 7:** Conclusion and Future Works

Chapter 2

Literature Review

2.1 Works Related to Hyperbolic Neural Networks and CNN

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 and Kiela, 2017b),(Nickel and 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 discrete data (Sala et al., 2018),(Sarkar, 2011).

Fully Connected Hyperbolic Neural Network

In 2018 (Ganea, Becigneul, and Hofmann, 2018), and in 2020 (Shimizu, Mukuta, and Harada, 2020) independently developed the structure of Hyperbolic Neural Networks on Poincaré Disc by utilizing the gyrovector space structure. They defined the generalized notions of different layers like fully connected, convolutional, or attention layers. (Fan, Yang, and Vemuri, 2022), (Qu and Zou, 2022) tried to develop variations of HNN models like fully Hyperbolic GAN on Lorentz Model space, (Spengler, Berkhout, and Mettes, 2023) proposed a fully hyperbolic CNN architecture on Poincaré Disc model. Very recently, (Bdeir, Schwethelm, and Landwehr, 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 recommending 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, Schwethelm, and Landwehr, 2023) proposed an efficient batch normalization algorithm based on the Lorentz model, utilizing the Lorentz centroid and a mathematical re-scaling 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 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.

2.2 Works Related to Transformer and Universal Approximation

Attention Mechanisms and It's Imperatives

Following the self-attention mechanism described in (Vaswani, 2017), numerous attempts have been made to devise various types of Attention Modules and their direct implementations to find contextual meaning in NLP tasks. The observation, that trained Transformer produces an Attention Matrix with a considerbale sparsity across data points (due to most of the tokens do not carry any contextual linking with most other tokens), gave rise to the concept of Sparse Attention Mechanism (Child et al., 2019), for which the components of the Attention Matrix are not stored where there is no contextual relation among tokens. Following that, a number of Sparse Attention Mechanisms have been proposed till date: for example, Band Attention and Global Attention [see Star Transformer (Qipeng et al., 2019)] , Inter Global Node Attention [see LongFormer, (Beltagy, Peters, and Cohan, 2020)], External Global Node Attention [see BigBird, (Zaheer et al., 2020), (Ho et al., 2019)]. A block based splitting of queries and keys followed by a Sinkhorn normalization procedure to produce a doubly-stochastic assignment based Attention Matrix [see Sparse Sinkhorn Attention (Tay et al., 2020)] can also facilitate the model to model locality.

Neural Networks and Universal Approximations

One of the most classical results in the theory of Neural Networks is the universal approximation theorems [see (Cybenko, 1989), (Hornik, 1991)]. They proved that any neural network with one hidden layer with sufficiently long width can arbitrarily approximate continuous functions of compact support. Several other works focused on bounding the depth of the hidden layer [see (Lu et al., 2017), (Lin and Jegelka, 2018)]. Hanin and Selke (Clark, 2019) proved that any neural network can approximate a scalar valued continuous function with hidden layer size $s + 1$, provided the input layer has a size of s. Moreover, Zhou (Zhou, 2020b) proved the universal approximation property in the context of a one-dimensional convolutional neural network. In 2019, Yun and others (Yun et al., 2020) showed that Transformers having residual connection and softmax activations in their Attention Layer can act as universal approximators of sequence to sequence functions of compact support, provided the input tokens are injected with positional

information. Additionally, Sannai (Sannai, Takai, and Cordonnier, 2019) showed the universality of permutation equivariant functions using fully connected deep neural networks with ReLU activations.

Chapter 3

Mathematical Preliminaries

We will discuss the preliminaries of Riemannian Manifolds and Hyperbolic Geometry which would underlay the foundation of our proposed frameworks.

Riemannian manifold, Tangent space, and Geodesics: Roughly speaking, an *n-dimensional Manifold* $\mathcal{M}$ is a topological space that locally looks like $\mathbb{R}^n$ (Tu, 2017). For each $x \in \mathcal{M}$, one defines the *Tangent Space* $T_x(\mathcal{M})$ as the first order linear approximation of $\mathcal{M}$ at x. $\mathcal{M}$ is called a *Reimannian Manifold* if for every point $x \in \mathcal{M}$, there is a collection of metrics $g := \{g_x : T_x(\mathcal{M}) \times T_x(\mathcal{M}) \to \mathbb{R}, x \in \mathcal{M}\}$ (Carmo, 1992). The distance function on this space is induced by these collections of Metrics, which is a 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 measured as $L(\gamma) := \int_a^b g_{\gamma(t)}(\gamma'(t), \gamma'(t))^{1/2} dt$. The *Geodesic* between two such points is that curve γ for which $L(\gamma)$ is the minimum and that $L(\gamma)$ is referred to as the *Geodesic Distance* between them. Given such a Riemannian Manifold $\mathcal{M}$ and two linearly independent vectors u and v at $T_x(\mathcal{M})$, we also 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 $R(u, v)w := \nabla_u \nabla_v w - \nabla_v \nabla_u w - \nabla_{(\nabla_u v - \nabla_v u)} w$ [where $\nabla_u v$ is the directional derivative of v in the direction of u, also known as the *Rimannian Connection* on $\mathcal{M}$.]

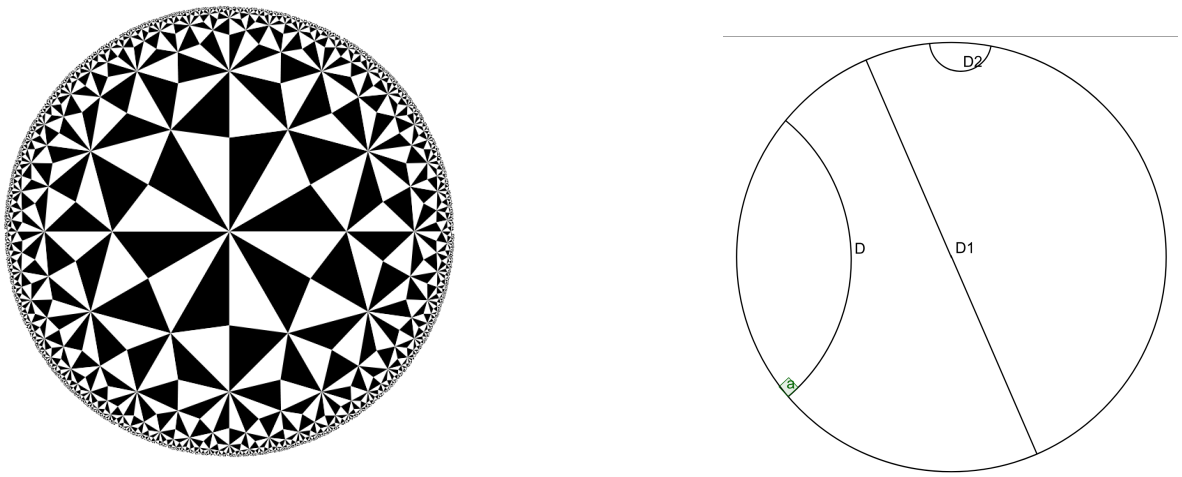
Following these notations, an n-dimensional model *Hyperbolic Space* is defined as the connected and complete Riemannian Manifold with a constant negative sectional curvature. To date, various model Hyperbolic Spaces are proposed in theory, such as Poincaré Disc Model, Poincaré half Space Model, Klein-Beltrami Model, Hyperboloid Model, etc. Still, the celebrated *Killing-Hopf Theorem* (Lang, 1995) states that all the model hyperbolic spaces are isometric provided they have the same dimension and curvature. We will use this theorem to develop our proposed algorithm uniquely (without introducing any performance variations) over a particular model space, where we conveniently pick the Poincaré Disc model. Now we briefly discuss the critical algebraic operations on this model space, which will be required for our purpose.

Poincaré Disc Model For a particular curvature $k(< 0)[c = -k]$, an $n-$ dimensional Poincaré Disc model contains can be thought of as 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 this ball. The geodesic distance between two points p and q (where $\|p\|, \|q\| < 1/\sqrt{c}$) is defined as

$$d(p, q) := 2 \sinh^{-1} \left(\sqrt{2 \frac{\|p - q\|^2}{c(\frac{1}{c} - \|p\|^2)(\frac{1}{c} - \|q\|^2)}} \right). \tag{3.1}$$

In what follows, we denote $\mathbb{D}_c^n$ as the $n-$ dimensional Poincaré Disc with curvature $-c$.

Gyrovector Space The concept of Gyrovector Space serves as a framework for studying vector space structures within Hyperbolic Space, introduced by Ungar [see (Ungar, 2022)]. This abstraction allows for defining special addition and scalar multiplications based on weakly associative



(A) The Poincaré Disc Model (B) Geodesics on the Poincaré Disc

FIGURE 3.1: The geometry of the Poincaré Disc Model: (A) Black regions and white regions have equal volumes, (B) The geodesic curves on the Disc

gyrogroups. We refer to Vermeer's work (Vermeer, 2005) for a detailed geometric formalism of these operations.

In this context, we briefly discuss Möbius Gyrovector Addition and Möbius Scalar Multiplication on the Poincar'e Disc. Due to isometry, transformations between hyperbolic spaces of the same dimensions, the same additive and multiplicative structures can be obtained for other model hyperbolic spaces (Ungar, 2022).

1. **Möbius Addition:** For two points p and q in the Poincaré Disc, the Möbius addition is defined as:
$$p \oplus_c q := \frac{(1 + 2c < p, q > +c\|q\|^2)p + (1 - c\|p\|^2)q}{1 + 2c < p, q > +c^2\|p\|^2\|q\|^2}, \tag{3.2}$$
where c is the negative of the curvature of the Poincaré Disc.

2. **Möbius Scalar Multiplication:** For $r \in \mathbb{R}$, $c > 0$ and p in the Poincaré Disc, the scalar multiplication is defined as:
$$r \otimes_c p := \frac{1}{\sqrt{c}} \tanh\left(r \tanh^{-1}(\sqrt{c}\|p\|)\right) \frac{p}{\|p\|}. \tag{3.3}$$
This addition and scalar multiplication satisfy the Gyrovector Group Axioms[see (Ungar, 2022)].

Fréchet Centroid: For a set of m points $\{x_1, x_2, ..., x_m\} \in \mathbb{D}_c^n$, we define the Fréchet centroid as a generalized notion of the Euclidean Centroid, defined as
$$FC(x_1, x_2, ..., x_m) := \frac{1}{m} \otimes_c (x_1 \oplus_c (x_2 \oplus_c ... (x_{m-1} \oplus_c x_m))) . \tag{3.4}$$

Exponential & Logarithmic Maps: For any $p \in \mathbb{D}_c^n$, the $\exp_p^c : T_p(\mathbb{D}_c^n) \subseteq \mathbb{R}^n \to \mathbb{D}_c^n$ translates a point from the tangent space of the Poincare Disc and projects it on the Poincaré Disc along the unit speed geodesic starting from $p \in \mathbb{D}_c^n$ in the direction $v \in T_x(\mathbb{D}_c^n)$. The Logarithmic map does exactly the opposite, i.e., $\log_p^c : \mathbb{D}_c^n \to T_p(\mathbb{D}_c^n) \subseteq \mathbb{R}^n$, projecting a point from the

Poincaré Disc back to the tangent space at $p \in \mathbb{D}_c^n$ along the reverse of the geodesic traced by the Exponential Map. Their explicit formulations are given as follows:

$$\exp_p^c(q) := p \oplus_c \left(\tanh\left(\sqrt{c} \frac{\lambda_p^c \|q\|}{2} \right) \frac{q}{\sqrt{c}\|q\|} \right) \tag{3.5}$$

and

$$\log_p^c(z) := \frac{2}{\sqrt{c}\lambda_p^c} \tanh^{-1}\left(\sqrt{c} \| -p \oplus_c z \| \right) \frac{-p \oplus_c z}{\| -p \oplus_c z \|}, \tag{3.6}$$

for $z \neq p$ and $q \neq 0$ and the Poincaré conformal factor $\lambda_p^c := \frac{2}{(1-c\|p\|^2)}$.

Gromov Hyperbolicity: For any metric space (X, d), the *Gromov Product* of two points x, y with respect to a third point w is defined as

$$(x, y)_w := \frac{1}{2}(d(x, w) + d(y, w) - d(x, y)) \tag{3.7}$$

and we say X is $\delta-$ hyperbolic iff for any tuple (x, y, z, w) of four points in X, we have

$$(x, z)_w \geq \min((x, y)_w, (y, z)_w) - \delta. \tag{3.8}$$

It can be proved that if Equation 3.8 is satisfied for one base point w, then it is satisfied for all base points up to a constant multiple of 2 (Coornaert, Delzant, and Papadopoulos, 2006). Therefore, we can conveniently remove the base point from the definition of the Gromov Product. Although this form was originally introduced by Gromov himself, there is an equivalent definition of the same easier for implementation purposes, which was introduced by Rips (Bridson and Haefliger, 2013). It reduces the four-point definition to an arbitrary geodesic triangle $[x, y, z] \in X$. According to Rips, such a triangle is said to be $\delta-$ slim if δ is the minimum positive value such that any side can be contained in the union of δ neighborhoods of the other two sides, and X is $\delta-$ hyperbolic if any triangle in X is $\delta-$ slim. Rips also showed that there exists a constant a such that (Bridson and Haefliger, 2013) X is $\delta-$ hyperbolic as defined by Gromov if and only if X is $a \cdot \delta$ hyperbolic as defined by Rips, and we call this δ as the *Gromov Hyperbolicity Index (GHI)* of X. However, according to both definitions, a space with lower GHI will be more hyperbolic (for lower GHI, the sides of any geodesic triangle will be closer to each other, indicating more negative bendness/curvature), compared to a space with higher GHI.

CAT(0) Space: A geodesic metric space $(\mathcal{X}, d)$ is called a CAT(0) Space if for any $p, q, r \in \mathcal{X}$ and for any length minimizing geodesic $\gamma : [0, 1] \to \mathcal{X}$ with $\gamma(0) = p$ and $\gamma(1) = q$,

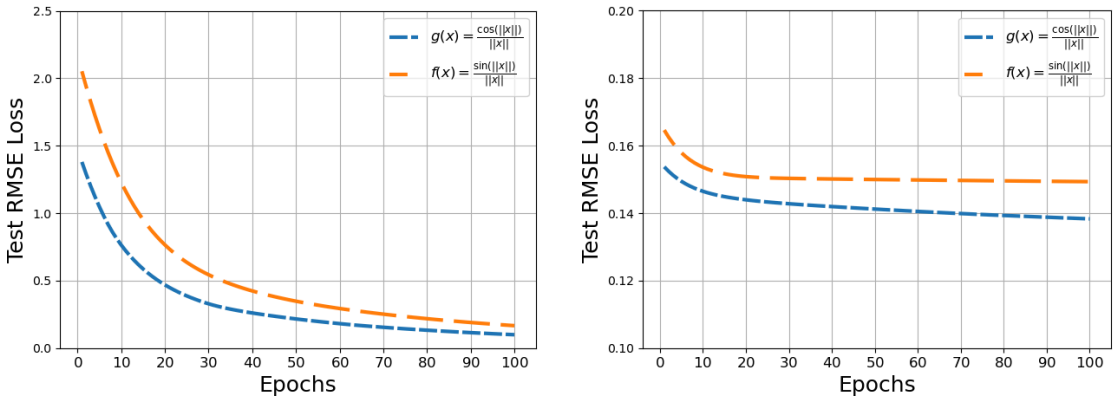
$$d^2(r, \gamma(t)) \leq (1-t)d^2(r, p) + td^2(r, q) - (1-t)td^2(p, q)$$

holds for all $t \in (0, 1)$. Any complete Riemannian Manifold with non-positive sectional curvature is a CAT(0) Space [see (Bacák, 2014)]. Therefore, any Hyperbolic Spaces, in particular, the Poincaré Discs are also CAT(0) Spaces.

Chapter 4

Universal Consistency of Hyperbolic Deep Convolutional Neural Network

In this chapter, we will prove the universal consistency of $1-$ Dimensional Expansive Hyperbolic Convolutional Neural Network. We will begin with a motivating example which will demonstrate the efficacy of our architecture over the conventional neural network architecture.



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

FIGURE 4.1: 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.

4.1 Motivating Example

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 4.1. Experiments

are conducted for two different curvatures $c = 0$ (Euclidean space) and $c = 1$. We considered unit radius Poincaré Disc as the hyperbolic space. Assuming the point set in discrete 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.

Now we will begin our theoretical analyses. At first we will mathematically describe our architecture, which heavily relies on Riemannian Geometry, described in the Chapter Mathematical Preliminaries [3]. Then we will state what universal consistency means in the hyperbolic set-up and in the final part of this chapter, we will derive the consistency analyses in detail.

4.2 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 Disc. 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 $f, g : \mathbb{R}^n \to \mathbb{R}$ as*

$$f \star g(x) := \exp_0^c \left[\int_{\mathbb{D}_c^n} f(\log_0^c(z)) g(\log_0^c(-z \oplus_c x)) \lambda(z) \right], \tag{4.1}$$

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 $\mathbb{D}_c$, since in the deep convolutional setup we will again convolute the output with some other filters.*

Definition 2 (Hyperbolic Expansive and Contractive Convolution (Discrete 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*

$$\boldsymbol{w} *_h \boldsymbol{v} := \exp_0^c(\boldsymbol{w} * \log_0^c(\boldsymbol{v})). \tag{4.2}$$

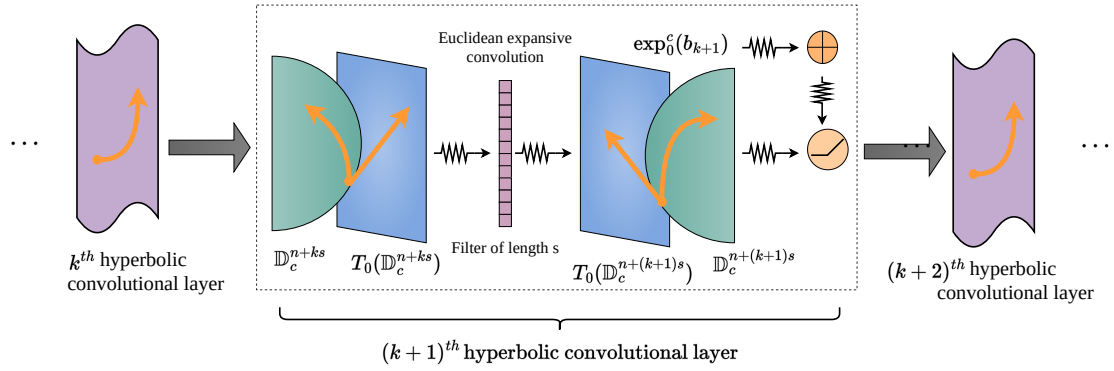


FIGURE 4.2: The complete workflow of expansive hyperbolic 1-D convolutional layer on Poincarê disc is presented. (best view in digital format)

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*

$$\boldsymbol{w} \star_h \boldsymbol{v} := \exp_0^c(\boldsymbol{w} \star \log_0^c(\boldsymbol{v})), \tag{4.3}$$

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

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:*

$$\boldsymbol{h}_k(x) = \sigma(\boldsymbol{w}_k \circ \boldsymbol{h}_{k-1}(x) \oplus_c \exp_0^c(\boldsymbol{b}_k)), \tag{4.4}$$

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:

$$h_L(x) = \exp_0^c\left[\boldsymbol{a}_L \cdot \log_0^c(\boldsymbol{h}_L(x))\right]. \tag{4.5}$$

The transformation of a vector, lying as a geodesic on Poincaré Disc, is shown in Figure 4.2 through hyperbolic convolution performed in an intermediate layer. This transformation projects the vector as another geodesic in a higher dimensional Poincaré Disc 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).

4.3 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 discussion. The goal is to learn a function $f_{\mathcal{D}} : \mathcal{X} \to \mathbb{R}^1$ that minimizes the following *Hyperbolic Generalization Error (HGE)*:

$$\mathcal{E}(f) := \int_{\mathcal{Z}} (f(x) - \log_0^c(y))^2 d\rho. \tag{4.6}$$

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 4. *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.*

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

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

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

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

The cross term in the last expression is 0, since

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

Therefore, the expression for HGE is reduced to

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

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 $x \in \mathcal{X}$

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

□

The next lemma will deduce what we aim to minimize.

Lemma 5. *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 4, we can write

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

□

The estimator that minimizes the hyperbolic generalization error is the estimator that 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:

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

where

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

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 4.5.

Now, to verify the consistency, we need to show that when the sample size $m \to \infty$, the sequence of estimators converges to the real estimate. This is formally defined as follows: A sequence of estimators for a parameter is said to be strongly universally consistent if it converges almost surely to the true value of the parameter. In the case of a regression problem, we say that a sequence of empirical error estimators, built through empirical risk minimizers, is strongly

universally consistent if they approach the generalization error over the class of outputs belonging to the Hilbert Space of square-integrable functions with respect to the distribution measure of the output conditioned on the input variable. Therefore in the hyperbolic setup, we define it in the following way:

Definition 6. *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 7. *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*

$$\frac{\left(\frac{1}{\sqrt{c}} \tanh^{-1}(M_m\sqrt{c})\right)^4 L_m^2 (L_m + d)\log(L_m)}{m^{1-2\theta}} \times \log\left(\left(\frac{1}{\sqrt{c}} \tanh^{-1}(M_m\sqrt{c})\right) m\right) \to 0, \tag{4.7}$$

hold for $\theta \in (0, 1/2)$ *and input filter length as* $2 \le s \le d$. *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 7, we get back Theorem 1 in (Lin et al., 2022b). Therefore, Theorem 7 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 7. 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 Disc, 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 7, 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 12. 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 4.7 will ensure the absolute difference between the generalization error and empirical error goes to* 0, *which will be used to prove Lemma 11.*

Remark 7. *Theorem 7 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, 2020a) [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 7. 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 11 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 7 we divide our works into three parts as demarcated in (Lin et al., 2022b), namely

1. **capacity Estimation**
2. **Finite Sample Error Bound**
3. **Universal Consistency**

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 4.5. We first need several terminologies.

4.3.1 Capacity Estimate for a class of functions expressed by 1- dimensional HCNN

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 balls 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 8. *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. □

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

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

4.3.2 Finite Sample Error Bound

We begin by defining 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 10. *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 10, the following Lemma will lay out the convergence criterion of the Truncated HEE estimates.

Lemma 11. *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 4.7 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 10 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 9 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 7 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

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

as $m \to \infty$ holds almost surely, completing the proof of Lemma 11.

□

4.3.3 Universal Consistency

We are finally in a position to prove Theorem 7; we will give our final lemma, which will complete the proof for universal consistency.

Lemma 12. *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 11 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 Disc, 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.

Proof. Define $g(y) := f(\exp_0^c(y))$ for $y \in \log_0^c(\Omega)$. Then, by Theorem 1 (Zhou, 2020a), 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 $\mathbb{R}^d \to \mathbb{D}_c^d$]. □

Proof of Theorem 7

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 12, 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

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

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

$$(a+b)^2 \leq (1+\epsilon)a^2 + (1+1/\epsilon)b^2 \tag{4.8}$$

for $a, b, \epsilon > 0$.

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

We will start with S_1 as,

$$\begin{aligned} S_1 &= \epsilon(\pi_M(f_{D,L,s})) - (1+\epsilon)(\pi_M(f_{D,L,s})) \\ &= \int_{\mathcal{Z}} |\pi_M(f_{D,L,s}(x)) - (\log_0^c(y_M)) + (\log_0^c(y_M)) - (\log_0^c(y))|^2 d\rho \\ &- (1+\epsilon)\int_{\mathcal{Z}} |\pi_M(f_{D,L,s}) - (\log_0^c(y_M))|^2 d\rho \\ &\leq (1+(1/\epsilon))\int_{\mathcal{Z}} |\log_0^c(y) - \log_0^c(y_M)|^2 d\rho. \end{aligned}$$

But we have $M = M_m \to \frac{1}{\sqrt{c}}$ as $m \to \infty$. Since $\epsilon > 0$ is arbitrary, we get $S_1 \to 0$ as $m \to \infty$.

By Lemma 12 and the constraints in the statement of Theorem 7 we get

$$S_2 \to 0 \text{ as } m \to \infty.$$

By the definition of the truncation operator, we get,

$$S_3 = \frac{1}{m}\sum_{i=1}^{m} |\pi_M(f_{D,L,s}(x_i)) - (\log_0^c(y_{i,M}))|^2 - \frac{1}{m}\sum_{i=1}^{m} |f_{D,L,s}(x_i) - (\log_0^c(y_{i,M}))|^2 \leq 0.$$

By the Strong Law of Large Numbers and inequality 4.8 we have,

$$\begin{aligned} S_4 &\leq (1+\epsilon)(1+1/\epsilon)\frac{1}{m}\sum_{i=1}^{m} |\log_0^c(y_i) - \log_0^c(y_{i,M})|^2 \\ &\to (1+\epsilon)(1+1/\epsilon)\int_{\ddagger} |\log_0^c(y) - \log_0^c(y_M)|^2 d\rho \end{aligned}$$

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

$$S_4 \to 0.$$

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

$$S_5 = (1+\epsilon)^2 \left(\frac{1}{m}\sum_{i=1}^{m} |f_{D,L}(x_i) - \log_0^c(y_i)|^2 - \frac{1}{m}\sum_{i=1}^{m} |f_{L_\epsilon}^{w,b}(x_i) - \log_0^c(y_i)|^2 \right) \leq 0.$$

Again by the Strong Law of Large Numbers, we have

$$S_6 \to 0$$

almost surely.

For S_7 we have

$$S_7 = (1+\epsilon)^2 \| f_{L_\epsilon} - f\rho \|^2_{L^2_{\rho_{\mathcal{X}}}}.$$

By Lemma 12, we get

$$S_7 \leq (1+\epsilon)^2 \epsilon.$$

Also, we have

$$S_8 \leq ((1+\epsilon)^2 - 1) \int_{\mathcal{Z}} |f_\rho(x) - \log_0^c(y)|^2 d\rho = \epsilon(\epsilon+2) \int_{\mathcal{Z}} |f_\rho(x) - \log_0^c(y)|^2 d\rho.$$

Summing up all the terms from S_1 to S_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. □

Source: The main source of this result can also be found at (Ghosh, Bose, and Das, 2024).

Chapter 5

Universal Consistency of Hyperbolic Transformer

5.1 Proposed Architecture and Problem Statement

Before diving into the problem at hand, let us first generalize the transformer architecture to adapt to a hyperbolic regime. This primarily calls for the transformations $\mathcal{A}(\cdot)$ and $\mathcal{F}(\cdot)$ to be operationally amenable to observations sampled from $\mathbb{D}_c^d$. In particular, let us assume that the input X (to the encoder) hails from $\mathbb{D}_c^{d\times t}$, i.e., a matrix of order $d \times t$ whose columns are embedded words or token vectors after getting mapped to the Poincaré Disc $\mathbb{D}_c^d$ through the Exponential Map, where d is the embedding dimension. As such, the encoder block of the corresponding transformer must constitute a sequence-to-sequence function, mapping $\mathbb{D}_c^{d\times t} \to \mathbb{R}^{d\times t}$, similar to that considered by (Yun et al., 2020). The adoption, mathematically, turns out as the following

$$\mathcal{A}^c(X) := X \oplus_c \exp_0^c \left[\sum_{j=1}^{h} U_f^j U_v^j \log_0^c(X) \sigma \left(\left(U_k^j \log_0^c(X) \right)^T U_q^j \log_0^c(X) \right) \right] \text{ and} \tag{5.1}$$

$$\mathcal{F}^c(X) := \log_0^c \left[\mathcal{A}^c(X) \oplus_c \exp_0^c \left[U_2 \text{ReLU} \left(U_1 \log_0^c \left(\mathcal{A}^c(X) \right) \right) + l_1 \mathbb{1}_m^T \right] \oplus_c \exp_0^c (l_2 \mathbb{1}_m^T) \right], \tag{5.2}$$

where the input X consisting of t tokens and $\{U_f^j\}_{j=1}^h \in \mathbb{R}^{d\times s}$ are the concatenating matrices. Corresponding to each head index j, the query, key, and value matrices are denoted respectively as $U_v^j, U_k^j, U_q^j \in \mathbb{R}^{s\times d}$, such that $U_2 \in \mathbb{R}^{d\times r}$, $U_1 \in \mathbb{R}^{r\times d}$ given $l_1 \in \mathbb{R}^r, l_2 \in \mathbb{R}^d$. Notably, both the curvature-corrected general self-attention (5.1) and the feed-forward map (5.2) remain equipped with a skip-connection, here, through an Möbius Addition operation. We denote any such resultant attention block consisting of a $\mathcal{A}^c$ layer with number of heads h, each with head size s; and a $\mathcal{F}^c$ layer with hidden layer size r, $t_{\mathcal{H}}^{h,s,r} : \mathbb{D}_c^{d\times t} \to \mathbb{R}^{d\times t}$. The collection of these sequence-to-sequence functions results in the following generalized hyperbolic attention architecture that accommodates the curvature of the underlying space.

$$\mathcal{T}_{\mathcal{H}}^{h,s,r} := \{ f : \mathbb{D}_c^{d\times t} \to \mathbb{R}^{d\times t} \mid f \text{ is a finite composition blocks } t_{\mathcal{H}}^{h,s,r} \}, \tag{5.3}$$

To this end, we also address the extension of a fixed positional encoding (E) to our hyperbolic architecture. We adopt the original sinusoidal positional encoding from (Vaswani, 2017), apply the exponential map to project these vectors into $\mathbb{D}_c^d$, and combine them with the input tokens using Möbius addition. The class of functions which can be expressed by $\mathcal{T}_{\mathcal{H}}^{h,s,r}$ attached with a hyperbolic positional encoding is represented as

$$\mathcal{T}_{\mathcal{P},\mathcal{H}}^{h,s,r} := \{ f_{\mathcal{P}}(X) := f(X \oplus_c E) \mid f \in \mathcal{T}_{\mathcal{H}}^{h,s,r} \text{ and } E \in \mathbb{D}_c^{d\times t} \}. \tag{5.4}$$

5.2 Theoretical Analysis

The setup underlying our problem begins with a dataset $\mathcal{D} := \{(X_i, Y_i)\}_{i=1}^t$, $t \in \mathbb{N}_+$ drawn independently at random following the unknown Borel Probability Measure ρ on $\mathcal{Z} := \mathcal{X} \times \mathcal{Y}$. While X_is are taken as the input observations and Y_is are the target outputs that we aim to model based on observed outputs from the $\mathcal{F}^c$ layer. Each X_i consists of t many tokens embedded in $\mathbb{D}_c^d$, i.e. $X_i := \{X_{i,1}, X_{i,2}, ..., X_{i,t}\} \in \mathcal{X} \subseteq \mathbb{D}_c^{d\times t}$ and the output tokens $Y_i := \{Y_{i,1}, Y_{i,2}, ..., Y_{i,t}\} \in \mathcal{Y} \subseteq \mathbb{D}_c^{d\times t}$. In our study, conforming to real scenarios such as word tokens based on a predefined dictionary, we assume $\mathcal{X}, \mathcal{Y}$ are taken from a compact subset $\mathcal{R}$ of $\mathbb{D}_c^{d\times t}$. The goal of statistical modeling based on transformers is to learn a sequence-to-sequence link function $f : \mathbb{D}_c^{d\times t} \to \mathbb{R}^{d\times t}$ that models the behavior of Y, given an explanatory variable X. Given that the model is well-specified, it boils down to minimizing the following estimation error

$$\mathcal{E}(f) := \int_{\mathcal{X}\times\mathcal{Y}} \|f(x) - \log_0^c(y)\|^2 d\rho. \tag{5.5}$$

Observe that under L^2 loss, it becomes equivalent to solving the ordinary functional regression. The next result proves an immediate guarantee that a benchmark optimizer exists for (5.5).

Lemma 13. *There exists an optimal solution to (5.5), given by the Hyperbolic Regression Function (HRF) $f_\rho(x) := \int_{\mathcal{Y}} \log_0^c(y) d\rho(y|x)$, where $\rho(\cdot|x)$ is the conditional distribution corresponding to ρ given $x \in \mathcal{X}$.*

Proof. Given the L^2 estimation error

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

for any function $g : \mathcal{X} \to \mathbb{R}^1$, $\mathcal{E}(g)$ can be written in terms of conditional expectation as

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

The cross term in the last expression is 0, since $\mathbb{E}_{\mathcal{X}}\left[\mathbb{E}_{\mathcal{Y}|\mathcal{X}}\left[\left(\mathbb{E}[\log_0^c(\mathcal{Y})|\mathcal{X}] - \log_0^c(\mathcal{Y})\right)\right]\right] = 0$. Therefore, the expression for $\mathcal{E}(g)$ is reduced to

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

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

□

It is quite intuitive to see that the expected response, conditioned on input tokens, would minimize the discrepancy in modeling. As such, our task of encapsulating the output law based on transformers must aim to approximate f_ρ universally. Observe that in an empirical setup, the problem (5.5) restricted to sequence-to-sequence functions $f : \mathbb{D}_c^{d\times t} \to \mathbb{R}^{d\times t}$ incurs the realized risk

$$f_D^{h,s,r} := \arg\min_{f\in\mathcal{T}_{\mathcal{P},\mathcal{H}}^{h,s,r}} \mathcal{E}_D(f), \tag{5.6}$$

where $\mathcal{E}_D(f) := \frac{1}{t}\sum_{i=1}^{t} \|f(X_i) - \log_0^c(Y_i)\|_{L^2}^2$ is the empirical estimation error. In the case when the input tokens X_is and the corresponding output labels Y_is are supported on $\mathbb{R}^d$, we define $\mathcal{E}_D(f) := \frac{1}{t}\sum_{i=1}^{t} \|f(X_i) - Y_i\|_{L^2}^2$ As a first step towards exploring the universal consistency of the minimizers, let us observe the large sample ($t \to \infty$) behaviour of the corresponding errors.

Definition 14 (Universal Strong Consistency (Györfi et al., 2006)). *A sequence of Regression Estimators (HRE) $(\{f_m\}_{m=1}^{\infty})$ obtained through Empirical Risk Minimization is said to be strongly universally consistent if it satisfies the condition*

$$\lim_{t\to\infty} \mathcal{E}_D(f_t) - \mathcal{E}(f_\rho) = 0$$

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

The following theorem provides a statistical guarantee that our optimal estimates $f_D^{h,s,r}$ indeed achieve strong consistency in the sense of Definition 14, under judicial choices of the model parameters.

Theorem 15. *Given an arbitrary $\theta \in (0, 1/2d)$ is arbitrary, number of input tokens t, and any fixed input embedding dimension d, if the following conditions hold*

(i) the sequence of the truncation parameters $M_t = M \to \frac{1}{\sqrt{c}}$ as $t \to \infty$,

(ii) $t^{-\theta} M_t^2 \left[1 + \frac{1}{M_t\sqrt{c}} \tanh^{-1}(M_t\sqrt{c})\right]^2 \to 0$, and

(iii) $\frac{P\log(Q)}{t^{1-2\theta d}} \to 0$, where

$$P := \left(\frac{1}{\sqrt{c}} \tanh^{-1}(M_t\sqrt{c})\right)^4 d\log(d)$$
$$Q := dt\left(\frac{1}{\sqrt{c}} \tanh^{-1}(M_t\sqrt{c}) t^{\theta}\right)^{dt} \log\left(dt\left(2\frac{1}{\sqrt{c}} \tanh^{-1}(M_t\sqrt{c})\right)^{dt}\right),$$

then, $\pi_{M_t} f_D^{2,1,4}$ is strongly universally consistent, such that $\pi_w(u) := \max\{w, |u|\}$ is a truncation operator.

The universal consistency of the typical Euclidean transformer can be derived as a special case of the theorem. In particular, by setting the limit $\lim c \to 0$ we get the following corollary.

Corollary 15.1 (Consistency in Euclidean Regime). *Given $\theta \in (0, 1/2d)$, the number of input tokens t, and input embedding dimension d, if the following conditions hold*

(i) The sequence of the truncation parameters $M_t = M \to \infty$ as $t \to \infty$,

(ii) $t^{-\theta} M_t^2 \to 0$ as $t \to \infty$, and

(iii) $\frac{P\log(Q)}{t^{1-2\theta d}} \to 0$, where $P = M_t^4 d\log(d)$ and $Q = dt(M_t t^{\theta})^{dt} \log\left(dt\left(2M_t t^{\theta}\right)^{dt}\right)$,

then, $\pi_{M_t} \tilde{f}_D^{2,1,4}$ is strongly universally consistent.

Remark 9. ***(Implication of using the truncation parameter M_t in Theorem 15)****In mathematical analysis, convergence of a series is typically studied by examining its partial sums; specifically, by analyzing the behavior of the sum up to the $t-th$ term for some t, and then by observing the limit as $t \to \infty$. This approach underpins the introduction of the truncation operator in Theorem 15. Unlike the approach in (Lin et al., 2022b), which takes $M_t \to \infty$, we consider the limit $M_t \to 1/\sqrt{c}$, effectively positioning our samples near the boundary of the Poincaré disc $\mathbb{D}_c^d$, whose radius is $1/\sqrt{c}$. As $M_t \to 1/\sqrt{c}$, the*

term $\tanh^{-1}(M_t\sqrt{c}) \to \infty$, *and consequently,* $M_t\left(\frac{1}{M_t\sqrt{c}}\tanh^1(M_t\sqrt{c})\right) \to \infty$. *This facilitates the derivation of an upper bound on the covering number of* $\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}$ *in terms of the truncation limit. Our use of the truncation operator is further motivated by its established role in proving the universal consistency of various learning algorithms (Györfi et al., 2006).*

The forthcoming discussion builds towards the proof of the theorem, compartmentalized into three phases, and along the way, explores the complexity and approximation capacity of the class of functions $\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}$. We call the class of functions defined by $\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}$ as the *HyT* block.

5.2.1 Capacity Estimates for a Class of Functions Charecterized by a HyT Block

To establish a covering argument to bound the metric entropy of the transformer class, let us review some basic notions. Observe that for $\rho \in \mathscr{P}(\mathcal{X})$ and a function $f : \mathcal{X} \to \mathbb{D}^K_c$, the L^p-norm is given as

$$\|f\|_{L^p(\rho)} := \left[\int_{\mathcal{X}} \left[\log^c_0(f(x))\right]^p d\rho\right]^{1/p},$$

where $v^p = \sum_{i=1}^n |v_i|^p$ for a vector $v = [v_1, v_2, ..., v_n] \in \mathbb{R}^n$, $p \geq 1$. By $L^p(\rho)$ we denote the set of all functions $f : \mathcal{X} \to \mathbb{D}^K_c$ satisfying $\|f\|_{L^p(\rho)} < \infty$. For any set of functions $T \subseteq L^p(\rho)$, we denote $\mathcal{N}(\epsilon, T, \|\cdot\|_{L^p(\rho)})$ as the *ϵ-covering number* of T with respect to (w.r.t.) the $\|\cdot\|_{L^p(\rho)}$, i.e., the smallest natural number n_0 such that the space T can be covered by n_0 many balls of radius ϵ w.r.t. $\|\cdot\|_{L^p(\rho)}$. By letting $v^m_1 := \{v_1, v_2, ..., v_m\} \in \mathcal{X}^m$, we define $\mathcal{N}_p(\epsilon, T, v^m_1) = \mathcal{N}(\epsilon, T, \|\cdot\|_{L^p(\rho_m)})$, where ρ_m is the empirical distributions of $v^m_1 \in \mathcal{X}^m$. We also define $\mathcal{M}(\epsilon, T, \|\cdot\|_{L^p(\rho)})$ to be the *ϵ-packing number* which is the largest integer n_p such that a subset $\{f_1, f_2, ..., f_{n_p}\}$ of T satisfying $\|\log^c_0(f_j) - \log^c_0(f_k)\|_{L^p(\rho)} \geq \epsilon$ for all $1 \leq j < k \leq n_p$. We denote $\mathcal{M}_p(\epsilon, T, v^m_1) = \mathcal{M}(\epsilon, T, \|\cdot\|_{L^p(\rho_m)})$. The following Lemma recalls the relationship between the ϵ-packing number and the ϵ-covering number.

Lemma 16 ((Györfi et al., 2006)). *For a class of functions T on $\mathcal{X}$, equipped with a probability measure ρ, we have for $p \geq 1$ and $\epsilon > 0$; $\mathcal{M}(2\epsilon, T, \|\cdot\|_{L^p(\rho)}) \leq \mathcal{N}(\epsilon, T, \|\cdot\|_{L^p(\rho)}) \leq \mathcal{M}(\epsilon, T, \|\cdot\|_{L^p(\rho)})$. Given $v^m_1 \in \mathcal{X}^m$, we have in particular $\mathcal{M}_p(2\epsilon, T, v^m_1) \leq \mathcal{N}_p(\epsilon, T, v^m_1) \leq \mathcal{M}_p(\epsilon, T, v^m_1)$.*

Now, let us generalize the definition of the Pseudo-Dimension (Pdim) for a class of real-valued functions to the same for those that take values in a Poincaré Disc. It becomes imperative to find a complexity bound for the transformer class.

Definition 17 (Pseudo-Dimension). *For $U \subseteq L^p(\rho)$, where $\rho \in \mathscr{P}(\mathcal{X})$, by the Pseudo-Dimension of U, we indicate the largest integer l (denoted as $Pdim(U)$) for which there exists $(x_1, x_2, ..., x_l, \eta^1_1, \eta^1_2, ..., \eta^1_l, ..., \eta^K_1, \eta^K_2, ..., \eta^K_l) \in \mathcal{X}^l \times \mathbb{R}^{Kl}$ such that for any $(a_1, a_2, ..., a_l) \in \{0,1\}^l$, there exists some $f \in U$ so that for all $i \in \{1, 2, ..., l\}$, $\log^c_0(f(x_i)) > \eta_i = [\eta^i_1, \eta^i_2, ..., \eta^i_l]^T \iff a_i = 1$.*

Previously, (Lin et al., 2022b) provided an upper bound on the ϵ-packing number for a class of real-valued functions using the findings of (Haussler, 1992). Here, we generalize the framework in Lemma 18, and in Theorem 19 using the concept of Pseudo-Dimension, which is immediately beneficial due to its avoidance of the curse of dimensionality based on the indifference to the Hausdorff (ambient) dimension.

Lemma 18. *Let F be a family of functions on $\mathcal{X}$ (equipped with a probability measure ρ) to $\mathbb{D}^K_c$ satisfying $\log^c_0(f) \in [-M, M]^K$, for all $f \in F$, where $M \in (0, 1/\sqrt{c})$. Let $r := [r_{ij}]_{m\times K}$ be a random matrix where r_{ij} follows $\mathcal{U}[-M, M]$ i.i.d. Also, let $v := \{v_1, ..., v_m\}$ be a random vector in $\mathcal{X}^m$, where each v_i is drawn independently at random following ρ. Then for all $\epsilon > 0$,*

$$\mathbb{E}\left(|sign((\log^c_0(F))|_v - r)|\right) \geq \mathcal{M}\left(\epsilon, F, \|\cdot\|_{L^p(\rho)}\right)\left(1 - \mathcal{M}(\epsilon, F, \|\cdot\|_{L^p(\rho)})e^{-m\left(\frac{\epsilon}{2M}\right)^K}\right), \text{ where}$$

$\log_0^c(F)|_v := \{\log_0^c(f)_v : f \in F\} = \{(\log_0^c(f(v_1)), \log_0^c(f(v_2)), ..., \log_0^c(f(v_m))) : f \in F\}$.

Proof. We take G to be an ϵ-separated subset of F with $|G| = \mathcal{M}(\epsilon, F, \|\cdot\|_{L^p(\rho)})$. Now,

$$\begin{aligned}
&\mathbb{E}\left(|\text{sign}((\log_0^c(F))|_v - r)|\right) \\
&\geq \mathbb{E}\left(|\text{sign}((\log_0^c(G))|_v - r)|\right) \\
&\geq \mathbb{E}(|\{f \in G : \text{sign}(\log_0^c(f)|_v - r) \neq \text{sign}(\log_0^c(g)|_v - r) \text{ for all } g \in G, g \neq f\}|) \\
&= \sum_{f \in G} \mathbb{P}(\text{sign}(\log_0^c(f)|_v - r) \neq \text{sign}(\log_0^c(g)|_v - r) \text{ for all } g \in G, g \neq f) \\
&= \sum_{f \in G} (1 - \mathbb{P}(\exists g \in G, g \neq f : \text{sign}(\log_0^c(f)|_v - r) = \text{sign}(\log_0^c(g)|_v - r))) \\
&\geq \sum_{f \in G} (1 - |G| \max_{g \in G, g \neq f} \mathbb{P}(\text{sign}(\log_0^c(f)|_v - r) = \text{sign}(\log_0^c(g)|_v - r))). \qquad (5.7)
\end{aligned}$$

If f and g are ϵ-separated in G; both $\log_0^c(f), \log_0^c(g) \in [-M, M]^K$, and r_i is drawn at random from the uniform distribution on $[-M, M]^K$, then the probability that $\text{sign}(\log_0^c(f)|_{v_i} - r_i) \neq \text{sign}(\log_0^c(g)|_{v_i} - r_i)$ will be at least $(\epsilon/2M)^K$, for $1 \leq i \leq m$. Hence, we get

$$\mathbb{P}(\text{sign}(\log_0^c(f)|_v - r) = \text{sign}(\log_0^c(g)|_v - r)) \leq \left(1 - \left(\frac{\epsilon}{2M}\right)^K\right) \leq e^{-m\left(\frac{\epsilon}{2M}\right)^K}. \qquad (5.8)$$

Putting the bounds in (5.7) and (5.8) together and by noting that $|G| = \mathcal{M}(\epsilon, F, \|\cdot\|_{L^p(\rho)})$, we get

$$\mathbb{E}\left(|\text{sign}((\log_0^c(F))|_v - r)|\right) \geq \mathcal{M}\left(\epsilon, F, \|\cdot\|_{L^p(\rho)}\right)\left(1 - \mathcal{M}(\epsilon, F, \|\cdot\|_{L^p(\rho)})e^{-m\left(\frac{\epsilon}{2M}\right)^K}\right).$$

This concludes the proof. □

We now use Lemma 18 to derive an upper bound on the ϵ-covering number of F.

Theorem 19. *For a family of functions F with $Pdim(F) = d$, we have for any $0 < \epsilon \leq 2M$*

$$\mathcal{M}(\epsilon, F, \|\cdot\|_{L^p(\rho)}) < 2\left[2eK\left(\frac{2M}{\epsilon}\right)^K \log\left(2eK\left(\frac{2M}{\epsilon}\right)^K\right)\right]^d.$$

Proof. By Sauer's Lemma, we can write

$$|\text{sign}(\log_0^c(F)|_v - r)| \leq \left(\frac{emK}{d}\right)^d,$$

for all $m \geq d$ and $r \in [-M, M]^{mK}$. Given $|G| = \mathcal{M}(\epsilon, F, \|\cdot\|_{L^p(\rho)})$, we get for all $m \geq d$,

$$\left(\frac{emK}{d}\right)^d \geq |G|\left(1 - |G|e^{-m\left(\frac{\epsilon}{2M}\right)^K}\right). \qquad (5.9)$$

It is easy to verify the claim of Theorem 19 if $\left(2M\epsilon^{-1}\right)^K \log(2|G|) < d$ simply by noting that $\epsilon \leq 2M$. So, to prove our claim, we can assume that $\left(2M\epsilon^{-1}\right)^K \log(2|G|) \geq d$. Hence, $m \geq \left(2M\epsilon^{-1}\right)^K \log(2|G|) \geq d$. Simplifying the term on the right in inequality (5.9), we get

$$\left(1 - |G|e^{-m\left(\frac{\epsilon}{2M}\right)^K}\right) \geq \frac{1}{2}.$$

Hence, using Sauer's Lemma, we have

$$\left(\frac{emK}{d}\right)^d \geq \frac{1}{2}|G|. \tag{5.10}$$

This is true for all $m \geq \left(\frac{2M}{\epsilon}\right)^K \log(2|G|) \geq d$. In particular, this is true for $m = \left(\frac{2M}{\epsilon}\right)^K \log(2|G|)$. Replacing m by $\left(\frac{2M}{\epsilon}\right)^K \log(2|G|)$ in (5.10), we get

$$|G| < 2\left(2eK\left(\frac{2M}{\epsilon}\right)^K \log\left(2eK\left(\frac{2M}{\epsilon}\right)^K\right)\right)^d,$$

completing the proof of Theorem 19. □

Now we calculate the number of trainable parameters in a single functional block of $\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}$, which consists of a $\mathcal{A}^c$ layer with parameters h and s, and a $\mathcal{F}^c$ layer with parameter r, as defined in (5.1) and (5.2) respectively. For each attention head, the query, key, value, and the output projection matrices from the concatenated attention heads have sd parameters each, totaling $4sd$ parameters. For h attention heads, this leads to a total of $4sdh$ trainable parameters for the $\mathcal{A}^c$ layer. Similarly, the $\mathcal{F}^c$ layer with a hidden dimension r has $2dr$ many trainable parameters. Finally, the trainable input embedding matrix adds another dv parameters, v being fixed to the vocabulary size. Putting everything together, for l blocks of $f \in \mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}$, there is a total of $n_{param} = l(4sdh + 2dr + dv) + \mathcal{O}(d)$, many trainable parameters.

Now coming to the number of actual neurons, in the $\mathcal{A}^c$ layer, in each attention head, there are d input neurons and s output neurons for each of the query, key, and value matrix computations. Furthermore, the matrix projecting the concatenating heads to the output demands another set of d neurons per head. Therefore the $\mathcal{A}^c$ layer requires a total of $4(d + sh)$ neurons. Similarly the $\mathcal{F}^c$ layer has $2d + r$ neurons (d input neurons, r hidden layer neurons, and d output neurons). Therefore, the total number of neurons or computational units required for l blocks consisting of a single $f \in \mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}$ in each block, is $n_{neuron} = l\,(6d + sh + r)$.

Remark 10. *Following the architecture of (Yun et al., 2019), if we consider the setup $h = 2, s = 1$, and $r = 4$ then n_{param} and n_{neuron} respectively reduces to $C_1 d$ and $C_2 d$, where C_1, C_2 are constants greater than 0.*

Now let us provide an upper bound on the Pseudo-Dimension of a truncated class of functions in $\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}$, which gets us to the next lemma.

Lemma 20. *An upper bound for the Pseudo-Dimension of $\pi_M(\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}})$ can be given as*

$$Pdim\left(\pi_M(\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}})\right) \lesssim n_{param} \log_2(n_{neuron}), \tag{5.11}$$

where $\pi_M(\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}) := \{f \in \mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}} : \log_0^c(f) \in [-M, M]^{d\times t}\}$.

Proof. By combining Theorem 7 from (Bartlett et al., 2019) and Theorem 14.1 from (Anthony and Bartlett, 2009) we get that,

$$\text{Pdim}\left(\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}\right) \leq C n_{param} \log_2(n_{neuron}),$$

for some constant $C > 0$. But $\pi_M\left(\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}\right) \subseteq \mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}$. Hence

$$\text{Pdim}\left(\pi_M\left(\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}\right)\right) \leq \text{Pdim}\left(\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}\right),$$

completing the proof of Lemma 20. □

Next, we give an upper bound on the covering number of $\pi_M\left(\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}\right)$, which enables us to estimate the capacity of any function in $\pi_M\left(\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}\right)$ w.r.t. the L^2 norm.

Lemma 21. *There exists an absolute constant C such that for any $0 \leq \epsilon \leq M$,*

$$\log \sup_{X_1^t \in \mathcal{X}^t} \mathcal{N}\left(\epsilon, \pi_M\left(\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}\right), X_1^t\right) \leq C n_{param} \log n_{neuron} \log\left[dt\left(\frac{2M}{\epsilon}\right)^{dt} \log\left(dt\left(\frac{2M}{\epsilon}\right)^{dt}\right)\right],$$

where $X_1^t := \{X_1, X_2, ..., X_t\}$ are the input tokens.

Proof. For a sequence of t tokens $X_1^t \in \mathbb{D}_c^{d\times t}$, we can write from Lemma 16

$$\mathcal{N}\left(\epsilon, \pi_M\left(\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}\right), X_1^t\right) \leq \mathcal{M}\left(\epsilon, \pi_M\left(\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}\right), X_1^t\right). \tag{5.12}$$

By writing $K = dt$, Theorem 19 enables us to write

$$\begin{aligned}\log\left(\mathcal{M}\left(\epsilon, \pi_M\left(\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}\right), X_1^m\right)\right) \leq & C_1 \text{Pdim}\left(\pi_M\left(\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}\right)\right) \\ & \times \log\left[dt\left(\frac{2M}{\epsilon}\right)^{dt} \log\left(dt\left(\frac{2M}{\epsilon}\right)^{dt}\right)\right].\end{aligned} \tag{5.13}$$

However, from Lemma 20 we have,

$$\text{Pdim}\left(\pi_M(\mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}})\right) \leq C n_{param} \log(n_{neuron}). \tag{5.14}$$

As such, combining inequalities (5.12), (5.13), and (5.14), we get the result (21), completing the proof. □

Corollary 21.1. *In what follows, combining Lemma 21 and Remark 10, we get the capacity estimate of $\pi_M\left(\mathcal{T}^{2,1,4}_{\mathcal{P},\mathcal{H}}\right)$ as,*

$$\log \sup_{X_1^t \in \mathcal{X}^t} \mathcal{N}\left(\epsilon, \pi_M\left(\mathcal{T}^{2,1,4}_{\mathcal{P},\mathcal{H}}\right), X_1^t\right) \lesssim d\log(d) \log\left[dt\left(\frac{2M}{\epsilon}\right)^{dt} \log\left(dt\left(\frac{2M}{\epsilon}\right)^{dt}\right)\right].$$

Remark 11. *Corollary 21.1 establishes an upper bound on the metric entropy of the covering number for a truncated subset of $\mathcal{T}^{2,1,4}_{\mathcal{P},\mathcal{H}}$. This, in turn, yields an upper bound on the capacity of the bounded function class represented by $\mathcal{T}^{2,1,4}_{\mathcal{P},\mathcal{H}}$. We use this result in Lemma 23 to deduce the convergence rate of the empirical error rates.*

5.2.2 Finite Sample Error Bound

Now, we aim to show that the empirical error sequences corresponding to the finite and bounded samples converge to the universal error, obtained due to the empirical risk minimization as described in (5.6). To this end, we provide an upper bound on the difference between the truncated empirical errors and the truncated form of the universal error. Thus, for any $f \in \mathcal{T}^{h,s,r}_{\mathcal{P},\mathcal{H}}$, we define $\mathcal{E}_{\pi_M}(f) := \int_{\mathcal{X}\times\mathcal{Y}} \|f(x) - \log_0^c(y_M)\|^2 d\rho$ and $\mathcal{E}_{D\pi_M}(f) := \frac{1}{t}\sum_{i=1}^t \|f(X_i) - \log_0^c(Y_{i,M})\|^2$, where $\pi_M(z) := z_M := \min\{M, |z|\} \cdot \text{sign}(z)$, for any $z \in \mathbb{R}$ and for z in higher dimensional spaces, the truncation is done component-wise.

Lemma 22 ((Györfi et al., 2006)). *We consider $\|y\| \leq B$ and $B \geq \frac{1}{\sqrt{c}}$. For a class of functions $f \in U$ from $\mathbb{D}_c^{d\times t} \to \mathbb{R}^{d\times t}$ satisfying $\|f(x)\| \leq B$, we have*

$$\begin{aligned} &Pr\Big[\exists f \in U : \mathcal{E}(f) - \mathcal{E}(f_\rho) - (\mathcal{E}_D(f) - \mathcal{E}_D(f_\rho)) \geq \epsilon(\alpha + \beta + \mathcal{E}(f) - \mathcal{E}(f_\rho))\Big] \\ &\leq 14 \sup_{X_1^t \in \mathcal{X}^t} \mathcal{N}_1\left(\frac{\beta\epsilon}{20B}, U, X_1^t\right) \exp\left(-\frac{\epsilon^2(1-\epsilon)\alpha t}{214(1+\epsilon)B^4}\right), \end{aligned} \tag{5.15}$$

for all $t \geq 1$ and $\alpha, \beta > 0$ and $\epsilon \in (0, 1/2d)$.

Now we use Lemma 22 to derive the finite sample error bound in Lemma 23.

Lemma 23. *Given $t^{-\theta} M_t^2 \left[1 + \frac{1}{M_t\sqrt{c}} \tanh^{-1}(M_t\sqrt{c})\right]^2 \to 0$, and condition (iii) in Theorem 15 holds for $\theta \in (0, 1/2d)$, then*

$$\lim_{t\to\infty} \mathcal{E}_{\pi_{M_t}}\left(\pi_{M_t} f_D^{2,1,4}\right) - \mathcal{E}_{\pi_{M_t},D}\left(\pi_{M_t} f_D^{2,1,4}\right) = 0, \ a.s.$$

Proof. Because $(\pi_M f_D^{h,s,r}), Y_M, Y_{i,M} \in \left[-\frac{1}{\sqrt{c}} \tanh^{-1}(M\sqrt{c}), \frac{1}{\sqrt{c}} \tanh^{-1}(M\sqrt{c})\right]^{dt}$, we get,

$$\left\|\mathcal{E}_{\pi_M}\left(\pi_M f_D^{h,s,r}\right) - \mathcal{E}_{\pi_M}\left(\pi_M f_D^{h,s,r}\right)\right\| \leq 8d\left(\frac{1}{\sqrt{c}} \tanh^{-1}(M\sqrt{c})\right)^2.$$

By putting $\alpha = \beta = 1$ and $\epsilon = t^{-\theta}$ in Lemma 22 we get,

$$\begin{aligned} &\left(\mathcal{E}_{\pi_M}\left(\pi_M f_D^{h,s,r}\right) - \mathcal{E}_{\pi_M}(f_\rho)\right) - \left(\mathcal{E}_{\pi_M}(\pi_M f_D^{h,s,r}) - \mathcal{E}_{\pi_M}(\pi_M f_\rho)\right) \\ \leq &8d\left(\frac{1}{\sqrt{c}} \tanh^{-1}(M\sqrt{c})\right)^2 t^{-\theta}, \end{aligned} \tag{5.16}$$

holds with probability at most

$$14 \sup_{X_1^t} \mathcal{N}_1\left(\left(\frac{t^{-\theta}}{2M}\right)^K \frac{1}{20}, \pi_M U, X_1^t\right) \exp\left(-\frac{t^{1-2d\theta}}{214 \times (3M^4)^d}\right).$$

We now use Corollary 21.1 to derive that,

$$\sup_{X_1^t} \mathcal{N}_1\left(\left(\frac{t^{-\theta}}{2M}\right)^d \frac{1}{20}, \pi_M U, X_1^t\right) \exp\left(-\frac{t^{1-2d\theta}}{214 \times (3M^4)^d}\right) \leq \exp\left[A \log(B)\right] \times \exp\left(-\frac{t^{1-2d\theta}}{214 \times (3M^4)^d}\right),$$

where

$$\begin{aligned} A &:= Cd\log(d), \\ B &:= dt\left(2M' t^\theta\right)^{dt} \log\left(dt\left(2M' t^\theta\right)^{dt}\right), \text{ and} \\ M' &:= \frac{1}{\sqrt{c}} \tanh^{-1}(M\sqrt{c}). \end{aligned}$$

If the conditions of Theorem 15 hold true for $(h, s, r) = (2, 1, 4)$ we get

$$\lim_{t\to\infty} \sup_{X_1^t} \mathcal{N}_1 \left(\left(\frac{t^{-\theta}}{2M} \right)^d \frac{1}{20}, \pi_M U, X_1^t \right) \exp\left(- \frac{t^{1-2d\theta}}{214 \times (3M^4)^d} \right) = 0.$$

Finally, combining this with the Strong Law of Large Numbers, we have,

$$\mathcal{E}_{\pi_{M_t}}(\pi_{M_t} f_D^{2,1,4}) - \mathcal{E}_{\pi_{M_t},D}(\pi_{M_t} f_D^{2,1,4}) = 0$$

holds almost surely, completing the proof of Lemma 23. □

Remark 12. *From Lemma 23 (see proof in Appendix), it is evident that for a finite truncation parameter M, the rate of convergence of the empirical error rates to the universal error is $\mathcal{O}(t^{-\theta})$ for some $\theta \in (0, 1/2d)$, with probability at least $1 - C^* P \log(Q) \exp\left(-\frac{t^{1-2d\theta}}{214\times(3M^4)^d}\right)$ (P, Q defined in Theorem 15), for some constant $C^* > 0$. As a consequence, for $t \to \infty$, $\hat{\mathcal{E}}(f)$ converges to the $\mathcal{E}(f_\rho)$ almost surely, by the second Borel-Cantelli Lemma.*

5.2.3 Step 3: Universal Approximations of Transformers

The next lemma will be our final module to prove Theorem 15, which will generalize the universal approximation property mentioned in (Yun et al., 2019) for the hyperbolic setting.

Lemma 24. *For any given $\epsilon > 0$ and a function $g : \mathbb{D}_c^{d\times t} \to \mathbb{R}^{d\times t}$, there exists $f \in \mathcal{T}_{\mathcal{P},\mathcal{H}}^{2,1,4}$ such that,*

$$\delta_2(f, g) := \left[\int_X \|f(x) - g(x)\|_2^2 dx \right]^{1/2} < \epsilon. \tag{5.17}$$

Proof. We define $h := g \circ \exp_0^c$. Then h is a continuous map from $\mathbb{R}^{d\times t} \to \mathbb{R}^{d\times t}$. Now g is compactly supported and $\exp_0^c : \mathbb{R}^{d\times t} \to \mathbb{D}_c^{d\times t}$ is a global diffeomorphism. Thus, h is a continuous map from $\mathbb{R}^{d\times t} \to \mathbb{R}^{d\times t}$ with compact support. Now, by Theorem 3 in (Yun et al., 2019), we know that there exists a $b \in T_{\mathcal{P},\mathcal{H}}^{2,1,4}$ such that

$$\left[\int_Y \|h(y) - b(y)\|_2^2 dy \right]^{1/2} < \frac{\epsilon}{C}, \tag{5.18}$$

where $C = \sup_{x\in\mathcal{R}\subsetneq\mathbb{D}_c^{d\times t}} \|\log_0^c(x)\|$. We have $C < \infty$, since $\mathcal{R}$ is a compact subset of $\mathbb{D}_c^{d\times t}$ and $\log_0^c$ is a global diffeomorphism on $\mathcal{R}$. We will now construct the target function f from b. Define $f := b \circ \log_0^c$. Then $f \in \mathcal{T}_{\mathcal{P},\mathcal{H}}^{2,1,4}$. We also note that

$$\begin{aligned}
\delta_2(f, g) &= \left[\int_X \|f(x) - g(x)\|_2^2 dx \right]^{1/2} \\
&= \left[\int_X \|h \circ \log_0^c(x) - b \circ \log_0^c(x)\|_2^2 dx \right]^{1/2} \\
&= \left[\int \|(h - b)(\log_0^c(x))\|_2^2 dx \right]^{1/2} \\
&\leq C \left[\int_Y \|h(y) - b(y)\|_2^2 dy \right]^{1/2} \leq \epsilon,
\end{aligned}$$

where the last inequality is followed from Equation (5.18). □

Proof of Theorem 15

Proof. Since we know that $\mathbb{E}[(\log_0^c(y))^2] < \infty$, by Lemma 24, for all $\epsilon > 0$ there exists $f \in \mathcal{T}^{2,1,4}_{\mathcal{P},\mathcal{H}}$ such that $\delta_2(f, f_\rho) < \epsilon$.

By the triangle inequality, we bound the following difference into a sum of 8 terms,

$$\begin{aligned}
\mathcal{E}(\pi_M(f_D^{2,1,4})) - \mathcal{E}(f_\rho) \leq & \epsilon\mathcal{E}\left(\pi_M(f_D^{2,1,4})\right) - (1+\epsilon)\mathcal{E}(\pi_M(f_D^{2,1,4})) \\
& + (1+\epsilon)\left(\mathcal{E}_{\pi_M}(\pi_M(f_D^{2,1,4}))\right) - \mathcal{E}_{\pi_M,D}(\pi_M(f_D^{2,1,4})) \\
& + (1+\epsilon)\left(\mathcal{E}_{\pi_M,D}(\pi_M(f_D^{2,1,4})) - \mathcal{E}_{\pi_M,D}(f_D^{2,1,4})\right) \\
& + (1+\epsilon)(\mathcal{E}_{\pi_M,D}(f_D^{2,1,4})) - (1+\epsilon)^2(\mathcal{E}_D(f_D^{2,1,4})) \\
& + (1+\epsilon)^2\left(\mathcal{E}_D(f_D^{2,1,4}) - \mathcal{E}_D(f)\right) + (1+\epsilon)^2\left(\mathcal{E}_D(f) - \mathcal{E}(f)\right) \\
& + (1+\epsilon)^2\left(\mathcal{E}(f) - \mathcal{E}(f_\rho)\right) + \left((1+\epsilon)^2 - 1\right)\mathcal{E}(f_\rho) \\
=: & \sum_{i=1}^{8} A_i,
\end{aligned}$$

where each of the A_i's represents a term in the summation. We will show that under the conditions of Theorem 15, each of the $A_i \to 0$. We note that, for each $q, w, \epsilon > 0$,

$$(q+w)^2 \leq (1+\epsilon)q^2 + (1+1/\epsilon)w^2. \tag{5.19}$$

We begin by showing that $A_1 \to 0$ as $t \to \infty$.

$$\begin{aligned}
A_1 &= \epsilon\mathcal{E}\left(\pi_M(f_D^{2,1,4})\right) - (1+\epsilon)\mathcal{E}(\pi_M(f_D^{2,1,4})) \\
&= \int_{\mathcal{Z}} \|\pi_M(f_D^{2,1,4}(x)) - (\log_0^c(y_M)) + (\log_0^c(y_M)) - (\log_0^c(y))\|^2 d\rho \\
&- (1+\epsilon)\int_{\mathcal{Z}} \|\pi_M(f_D^{2,1,4}) - (\log_0^c(y_M))\|^2 d\rho \\
&\leq (1+(1/\epsilon))\int_{\mathcal{Z}} |\log_0^c(y) - \log_0^c(y_M)|^2 d\rho. \quad \text{[By (5.19)]}
\end{aligned}$$

Since $\epsilon > 0$ is arbitrary, by condition (i) in Theorem 15, we have $A_1 \to 0$ as $t \to \infty$. Next, combining Lemma 24 and the conditions of Theorem 15, we get $A_2 \to 0$ as $t \to \infty$. The definition of the truncation operator results in

$$A_3 = \frac{1}{t}\sum_{i=1}^{t} \|\pi_M(f_D^{2,1,4}(X_i)) - (\log_0^c(Y_{i,M}))\|^2 - \frac{1}{t}\sum_{i=1}^{t} \|f_D^{2,1,4}(X_i) - (\log_0^c(Y_{i,M}))\|^2 \leq 0.$$

Using the Strong Law of Large Numbers and (5.19), we deduce

$$A_4 \leq (1+\epsilon)(1+1/\epsilon)\frac{1}{t}\sum_{i=1}^{t} \|\log_0^c(Y_i) - \log_0^c(Y_{i,M})\|^2 \to (1+\epsilon)(1+1/\epsilon)\int_{\mathcal{Z}} \|\log_0^c(y) - \log_0^c(y_M)\|^2 d\rho$$

as $t \to \infty$ almost surely. And the condition (i) in Theorem 15 implies $A_4 \to 0$ as $t \to \infty$. Since $f_D^{2,1,4}$ is obtained by minimizing the empirical risk, we have,

$$A_5 = (1+\epsilon)^2\left(\frac{1}{t}\sum_{i=1}^{t} \|f_D^{2,1,4}(X_i) - \log_0^c(Y_i)|^2 - \frac{1}{t}\sum_{i=1}^{t} |f(X_i) - \log_0^c(Y_i)|^2\right) \leq 0.$$

Once again, the strong law of large numbers directly indicates $A_6 \to 0$ almost surely as $t \to \infty$. We can write A_7 as $A_7 = (1+\epsilon)^2 \|f - f\rho\|^2_{L^2_{\rho_\mathcal{X}}}$, which goes to 0 by Lemma 24. Finally, for A_8, we have

$$A_8 \leq ((1+\epsilon)^2 - 1) \int_\mathcal{Z} \|f_\rho(x) - \log_0^c(y)\|^2 d\rho = \epsilon(\epsilon+2) \int_\mathcal{Z} \|f_\rho(x) - \log_0^c(y)\|^2 d\rho,$$

which definitely goes to 0 as $\epsilon > 0$ is arbitrary and the integral is bounded. This completes the proof of Theorem 15.

□

Chapter 6

Experimental Results

In this Section, we demonstrate the Experimental Results obtained by our architectures and simultaneously compare them with their Euclidean counterparts. We will do it in two separate sections: the first one will be for the Hyperbolic CNN, and the second part will be for the Hyperbolic Transformer.

6.1 Experiments with eHDCNN

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.

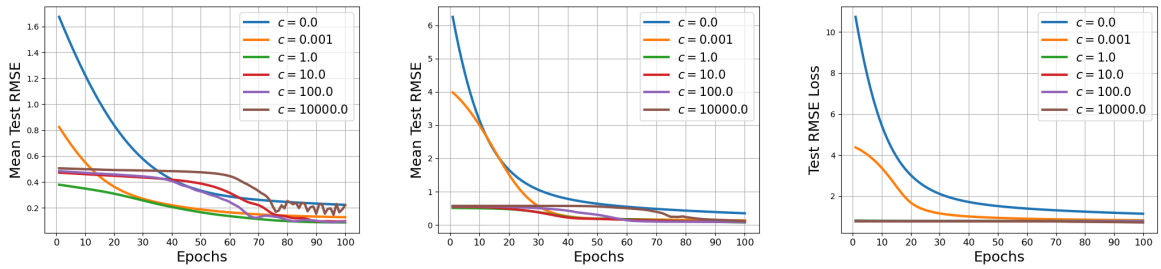


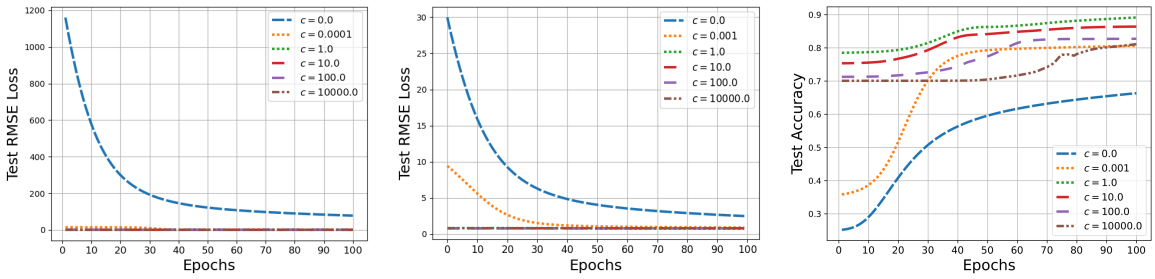
FIGURE 6.1: The performance analysis of eHDCNN with varying space curvatures (a) for $f(x)$ and (b) for $g(x)$, and (c) House price prediction is demonstrated. The Root Mean Square Error (RMSE) decreases faster with increasing curvature, justifying the utility of applying hyperbolic convolution. (best view in digital format)

6.1.1 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 $x \sim unif([-1,1]^{10})$. A fixed set of 800/200 samples for a 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 6.1(a) and 6.1(b) for the detailed illustration.



(a) (b) (c)

FIGURE 6.2: 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)

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 Disc 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.

TABLE 6.1: 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

6.1.2 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 6.1 and 6.2.

TABLE 6.2: 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

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 and 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 6.1(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 6.2 (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 6.2 (b).

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, Weiss, and Moore, 2011). As it is described in (Lin et al., 2022b), 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 6.1(c), 6.2(a), 6.2(b), and 6.2(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 higher range. It occurs due to the shrinkage of the Poincarê disc with a very high value of the curvature.

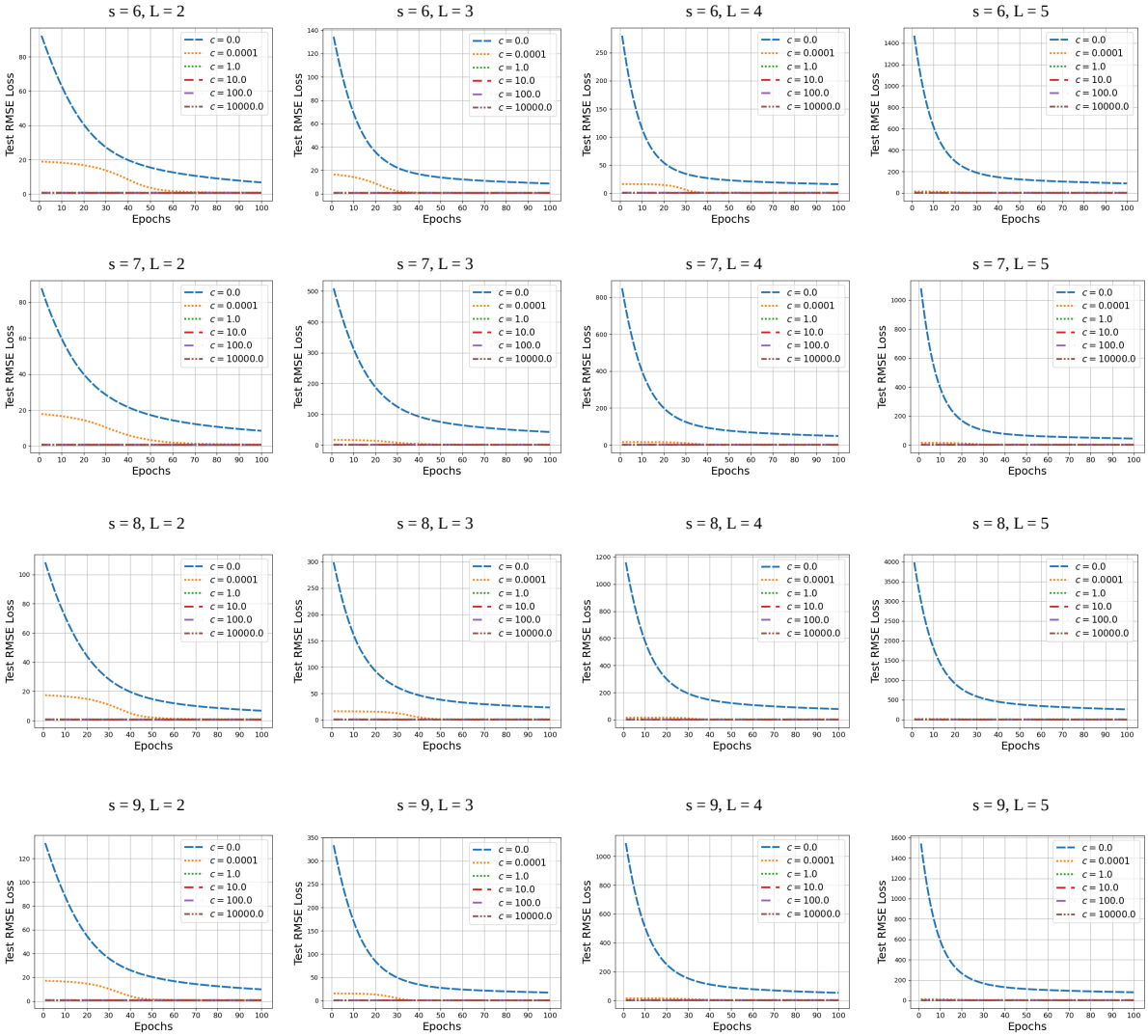


FIGURE 6.3: Various experiments were performed on the Superconductivity dataset by varying filter length and number of convolutional layers of the eHD-CNN architecture.

6.1.3 Ablation Study

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 of (s, L), and vary the curvatures of the Poincaré disc. The test RMSE curves are plotted and all results are presented in Figure 6.3. 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.

6.2 Experiments with Hyperbolic Transformer

We will discuss the results obtained from our simulation on real-world question answering datasets such as Squad, BoolQ, etc, on a BERT-like only-encoder based Transformer Model. TO validate our theoretical claims, we will run our simulation on the same model with four different curvatures: curvature 0 (Euclidean) and on Poincaré Balls with curvatures $0.0001, 1.0$, and 100 respectively. The Python-based implementation is available at
https://github.com/sagarghosh1729/Hyperbolic_BERT.git.

6.2.1 Model Description

Our model builds upon the established Euclidean transformer and its hyperbolic variants, while the Euclidean version is just a computationally efficient special case (c=0) of more general hyperbolic models. This relationship allows us to extend empirical findings from our Euclidean model to its hyperbolic counterparts. Therefore, we employ a small 1-block encoder-only network with specific dimensions: $d = 128$ (embedding dimension), $h = 2$ (number of attention heads), $s = 1$ (size of the head), and $r = 4$ (hidden layer dimension in the feed-forward block), aligning with our theoretical framework. For practical implementation, we adopt the vocabulary, pretrained wordpiece tokenizer, and sinusoidal positional embeddings from a standard BERT. We train our model as a regressor using Mean Squared Error (MSE) loss and the Adam optimizer (learning rate 0.001, weight decay 0.01). For each dataset and for a chosen curvature, we run the model only once and note the Test Root Mean Squared Error (Test RMSE) against the number of training iterations.

6.2.2 Dataset Description

1. **Squad**
   - *No of Samples:* $\sim 100k$
   - *Type:* Extractive QA type, answers span on a continuous block of text in the context.
   - *Use Cases:* Popular benchmarks for training and evaluating extractive QA models like BERT, a regression setup.
2. **Bool QA**
   - *No of Samples:* $\sim 12k$
   - *Type:* Binary question answer type
   - *Use Cases:* Simple question-answer with a classification setup.

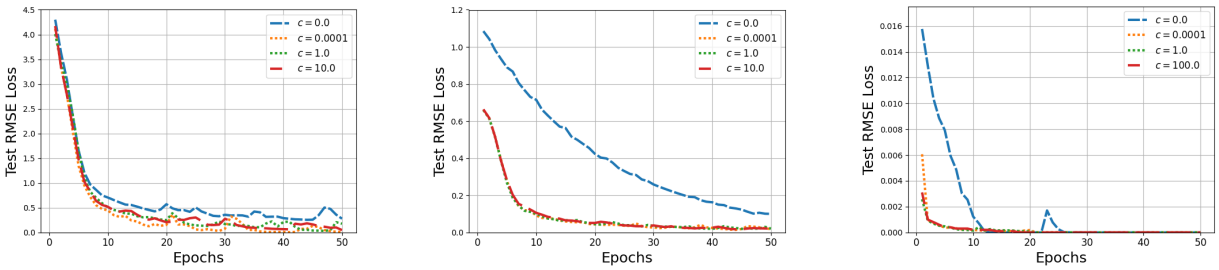
3. **Tweet QA**
    - *No of Samples:* $\sim 13k$
    - *Type:* Tweets
    - *Use Cases:* Presence of noise, more towards real-world-like data, a regression setup

6.2.3 Dataset Specific Model Descriptions

The Table 6.3 refers to the simulation-specific hyperparameters and other details.

TABLE 6.3: The complete details of hyperparameters for three real-world datasets are presented to reproduce the results.

Hyperparameter Details	Squad QA Dataset	Bool QA Dataset	Tweet QA Dataset
No of Encoders	1	1	1
Input Embedding Dimension	128	128	128
Noise	Yes	Yes	Yes
Learning Rate	0.001	0.001	0.001
Weight decay	0.0005	0.0005	0.0005
Train/test split	0.80	0.80	0.80
No of samples	$\sim 100k$	$\sim 12k$	$\sim 13k$
Batch Size	16	16	16
Max Sequence length	512	512	512
(h, s, r)	$(2, 1, 4)$	$(2, 1, 4)$	$(2, 1, 4)$
Optimizer	Adam	Adam	Adam
No of Input Tokens	$\sim 12M$	$\sim 0.4M$	$\sim 0.27M$



(A) Squad QA (B) Bool QA (C) Tweet QA

FIGURE 6.4: Test RMSE Loss over the number of training iterations of the three Question-Answering datasets shown above

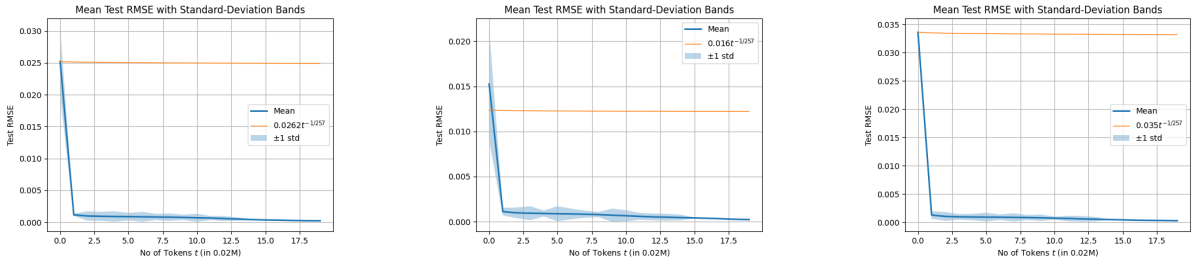
6.2.4 Discussion on Simulations

We now present and analyze the results obtained from our experiments on the three datasets introduced earlier. The accompanying figure serves as a visual reference for this discussion. Since we have framed the sequence-to-sequence translation task as a large-scale regression problem, with the L^2 generalization error as the underlying principle, our evaluation metric is the Test RMSE, plotted against the number of training iterations. For each dataset, we simulated the BERT-based encoder model using the parameter configuration $(h, s, r) = (2, 1, 4)$ across 50 training iterations, testing four different curvature values: 0.0, 0.0001, 1.0, and 10.0. The resulting Test

RMSE curves demonstrate the benefit of our hyperbolic generalization. A slight performance decline at higher curvature values is expected due to the Poincaré Disc contracting toward a single point as the curvature parameter $c \to \infty$.

6.2.5 Analyzing the Sample Complexity of Transformers

In this section, we will empirically validate our claim made in Remark 12 regarding the sample complexity on the Tweet QA Dataset. We ran the same experiment at the $20-$th epoch 10 times to note the mean Test RMSE at the token numbers in multiples of $20,000$ and the corresponding standard deviation. We refer to Figure 6.5 for the visual demonstration. It is evident that the Test RMSE curves along with the standard deviation bands always lie under the curve $y = t^{-1/257}$ (up to a constant) as the number of training tokens becomes large, empirically validating our claim made in Remark 12.



(A) $c = 0.0001$ (B) $c = 1.0$ (C) $c = 10.0$

FIGURE 6.5: Test RMSE Loss over the number of training tokens (in 0.02 Millions) of the Tweet QA dataset over the course of three different curvatures

Chapter 7

Conclusion and Future Works

7.1 Conclusion and Future Works related to eHDCNN

Conclusion

In this thesis, 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 Poincar'e Disc, projecting them onto the Tangent Space for expansive convolution, and then mapping them back to a higher-dimensional Poincaré Disc 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.

Future Works

Although this is the first work related to consistency analysis of a hyperbolic neural network, our method evolves around $1-$dimensional hyperbolic convolutions. The main bottleneck of extending similar analyses to higher dimensions is the lack of universal approximation results for CNNs beyond $2-$ dimensions. The convolution operation can be thought of as a matrix multiplication by sparse Toeplitz-type operators, factorization of which into block Toeplitz Operators becomes infeasible in higher dimensions [(Zhou, 2018) (Zhou, 2020b)]. Hence, the universal approximation property of $2-$ dimensional CNNs remains open to date, which obstructs the proof of universal consistency of similar networks beyond $2-$dimensions.

Another interesting observation from our simulations results is that: the test errors/accuracy of all the datasets, when trained with an eHDCNN architecture, converges much faster to the generalization error compared to the conventional CNNs. We believe the theoretical validation of this result could be extremely difficult (because we need to track all the parameter updation parts on each mini-batch in every epoch of the Adam Optimizer), yet this could be a fascinating result to prove.

7.2 Conclusion and Future Works related to Hyperbolic Transformer

Conclusion

We proved that Transformers and their hyperbolic variants are universally consistent when trained under sufficient regularity constraints. Our result emboldens the Transformer architecture with the confidence that when trained with a sufficient number of input tokens, eventually no model can perform better than it, provided the training is done as a sequence-to-sequence model. This theoretical guarantee places Transformers on firm ground not only in terms of empirical success, as demonstrated in language modeling (Vaswani, 2017), vision (Dosovitskiy et al., 2020), and even reinforcement learning (Chen et al., 2021), but also in terms of foundational learning theory. Universal consistency implies that, under increasing data and appropriate model scaling, the Transformer will asymptotically achieve the lowest possible generalization error, a property long considered a benchmark for statistical optimality (Györfi et al., 2006).

Building on this foundation, we conducted a detailed sample complexity analysis, quantifying how the number of training samples impacts the convergence of empirical error to generalization error. Our results show that, under suitable truncation and model capacity constraints, the convergence rate is $\mathcal{O}(t^{-1/2d})$, d being the input embedding dimension and t being the number of training tokens. This provides practical insights into how much data is needed for the Transformer to achieve near-optimal performance, aligning with recent scaling laws observed empirically.

Moreover, by extending our results to hyperbolic variants, we further generalized the theory to settings where data resides in non-Euclidean spaces. These models are particularly well-suited for capturing hierarchical or graph-structured relationships common in natural language, social networks, and phylogenetics (Kaplan et al., 2020).

Future Works

While our analysis assumes idealized training conditions and sufficient data availability, it offers theoretical backing for scaling laws observed in practice (Kaplan et al., 2020) and motivates further research into consistency under non-i.i.d. or adversarial scenarios. Future work could explore generalization in more structured output spaces, robustness under distributional shifts or adversarial attacks, and tighter bounds on the convergence rate of empirical to expected error.

In summary, our findings not only deepen the theoretical understanding of Transformer-based models but also suggest a broader principle: under suitable conditions, attention-based architectures are not only powerful in practice but also provably optimal in learning theory.

Bibliography

Anthony, Martin and Peter L Bartlett (2009). *Neural network learning: Theoretical foundations*. cambridge university press.

Atigh, Mina Ghadimi et al. (2022). "Hyperbolic image segmentation". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4453–4462.

Bacák, Miroslav (2014). *Convex analysis and optimization in Hadamard spaces*. Vol. 22. Walter de Gruyter GmbH & Co KG.

Bartlett, Peter L et al. (2019). "Nearly-tight VC-dimension and pseudodimension bounds for piecewise linear neural networks". In: *Journal of Machine Learning Research* 20.63, pp. 1–17.

Bdeir, Ahmad, Kristian Schwethelm, and Niels Landwehr (2023). "Fully Hyperbolic Convolutional Neural Networks for Computer Vision". In: *arXiv preprint arXiv:2303.15919*.

Beltagy, Iz, Matthew E Peters, and Arman Cohan (2020). "Longformer: The long-document transformer". In: *arXiv preprint arXiv:2004.05150*.

Bouschery, Sebastian G, Vera Blazevic, and Frank T Piller (2023). "Augmenting human innovation teams with artificial intelligence: Exploring transformer-based language models". In: *Journal of Product Innovation Management* 40.2, pp. 139–153.

Bridson, Martin R and André Haefliger (2013). *Metric spaces of non-positive curvature*. Vol. 319. Springer Science & Business Media.

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

Chen, Lili et al. (2021). "Decision transformer: Reinforcement learning via sequence modeling". In: *Advances in neural information processing systems* 34, pp. 15084–15097.

Chen, Yankai et al. (2022). "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.

Child, Rewon et al. (2019). "Generating long sequences with sparse transformers". In: *arXiv preprint arXiv:1904.10509*.

Clark, Kevin (2019). "What Does Bert Look At? An Analysis of Bert's Attention". In: *arXiv preprint arXiv:1906.04341*.

Coornaert, Michel, Thomas Delzant, and Athanase Papadopoulos (2006). *Géométrie et théorie des groupes: les groupes hyperboliques de Gromov*. Vol. 1441. Springer.

Cybenko, George (1989). "Approximation by superpositions of a sigmoidal function". In: *Mathematics of control, signals and systems* 2.4, pp. 303–314.

Djeddal, Hanane et al. (2021). "Hyperbolic K-means for traffic-aware clustering in cloud and virtualized RANs". In: *Computer Communications* 176, pp. 258–271.

Dong, Linhao, Shuang Xu, and Bo Xu (2018). "Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition". In: *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, pp. 5884–5888.

Dosovitskiy, Alexey et al. (2020). "An image is worth 16x16 words: Transformers for image recognition at scale". In: *arXiv preprint arXiv:2010.11929*.

Fan, Xiran, Chun-Hao Yang, and Baba C Vemuri (2022). "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.

Fu, Kun et al. (2019). "Enhanced feature representation in detection for optical remote sensing images". In: *Remote Sensing* 11.18, p. 2095.

Ganea, Octavian, Gary Becigneul, and Thomas Hofmann (2018). "Hyperbolic Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2018/file/dbab2adc8f9d078009ee3fa810bea142-Paper.pdf.

Ghosh, Sagar, Kushal Bose, and Swagatam Das (2024). "On the Universal Statistical Consistency of Expansive Hyperbolic Deep Convolutional Neural Networks". In: *arXiv preprint arXiv:2411.10128*.

Gu, Shuhao and Yang Feng (2019). "Improving multi-head attention with capsule networks". In: *Natural Language Processing and Chinese Computing: 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9–14, 2019, Proceedings, Part I 8*. Springer, pp. 314–326.

Gulati, Anmol et al. (2020). "Conformer: Convolution-augmented transformer for speech recognition". In: *arXiv preprint arXiv:2005.08100*.

Guo, Yunhui et al. (2022a). "Clipped hyperbolic classifiers are super-hyperbolic classifiers". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11–20.

— (2022b). "Clipped hyperbolic classifiers are super-hyperbolic classifiers". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11–20.

Györfi, L. et al. (2002). *A Distribution-Free Theory of Nonparametric Regression*. Springer Series in Statistics. Springer New York. ISBN: 9780387954417. URL: https://books.google.co.in/books?id=0vmb9oGBlo0C.

Györfi, László et al. (2006). *A distribution-free theory of nonparametric regression*. Springer Science & Business Media.

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

Haussler, David (1992). "Decision theoretic generalizations of the PAC model for neural net and other learning applications". In: *Information and computation* 100.1, pp. 78–150.

He, Kaiming et al. (2016). "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Ho, Jonathan et al. (2019). "Axial attention in multidimensional transformers". In: *arXiv preprint arXiv:1912.12180*.

Hornik, Kurt (1991). "Approximation capabilities of multilayer feedforward networks". In: *Neural networks* 4.2, pp. 251–257.

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

Kaplan, Jared et al. (2020). "Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models". In: *arXiv preprint arXiv:2001.08361* 1.2, p. 4.

Khrulkov, Valentin et al. (2020). "Hyperbolic image embeddings". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6418–6428.

Kwapisz, Jennifer R, Gary M Weiss, and Samuel A Moore (2011). "Activity recognition using cell phone accelerometers". In: *ACM SigKDD Explorations Newsletter* 12.2, pp. 74–82.

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

LeCun, Yann et al. (1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11, pp. 2278–2324.

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

Li, Yicong et al. (2021). "Hyperbolic hypergraphs for sequential recommendation". In: *Proceedings of the 30th ACM international conference on information & knowledge management*, pp. 988–997.

Lin, Fangfei et al. (2022a). "Contrastive multi-view hyperbolic hierarchical clustering". In: *arXiv preprint arXiv:2205.02618*.

Lin, Hongzhou and Stefanie Jegelka (2018). "Resnet with one-neuron hidden layers is a universal approximator". In: *Advances in neural information processing systems* 31.

Lin, Shao-Bo et al. (2022b). "Universal consistency of deep convolutional neural networks". In: *IEEE Transactions on Information Theory* 68.7, pp. 4610–4617.

Lin, Tianyang et al. (2022c). "A survey of transformers". In: *AI open* 3, pp. 111–132.

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

Lou, Aaron et al. (2020). "Differentiating through the fréchet mean". In: *International conference on machine learning*. PMLR, pp. 6393–6403.

Lu, Zhou et al. (2017). "The expressive power of neural networks: A view from the width". In: *Advances in neural information processing systems* 30.

Mathieu, Emile et al. (2019). "Continuous hierarchical representations with poincaré variational auto-encoders". In: *Advances in neural information processing systems* 32.

Mehdipour, Hossein et al. (2024). "Optimization of power take-off system settings and regional site selection procedure for a wave energy converter". In: *Energy Conversion and Management: X* 22, p. 100559.

Mishne, Gal et al. (2023). "The numerical stability of hyperbolic representation learning". In: *International Conference on Machine Learning*. PMLR, pp. 24925–24949.

Nagano, Yoshihiro et al. (2019). "A wrapped normal distribution on hyperbolic space for gradient-based learning". In: *International Conference on Machine Learning*. PMLR, pp. 4693–4702.

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

— (2017b). "Poincaré embeddings for learning hierarchical representations". In: *Advances in neural information processing systems* 30.

— (2018). "Learning continuous hierarchies in the lorentz model of hyperbolic geometry". In: *International conference on machine learning*. PMLR, pp. 3779–3788.

Qipeng, Guo et al. (2019). "Star-transformer". In: *Proceedings of HLT-NAACL*, pp. 1315–1325.

Qu, Eric and Dongmian Zou (2022). "Lorentzian Fully Hyperbolic Generative Adversarial Network". In: *arXiv preprint arXiv:2201.12825*.

Rives, Alexander et al. (2021). "Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences". In: *Proceedings of the National Academy of Sciences* 118.15, e2016239118.

Sala, Frederic et al. (2018). "Representation tradeoffs for hyperbolic embeddings". In: *International conference on machine learning*. PMLR, pp. 4460–4469.

Sannai, Akiyoshi, Yuuki Takai, and Matthieu Cordonnier (2019). "Universal approximations of permutation invariant/equivariant functions by deep neural networks". In: *arXiv preprint arXiv:1903.01939*.

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

Schwaller, Philippe et al. (2019). "Molecular transformer: a model for uncertainty-calibrated chemical reaction prediction". In: *ACS central science* 5.9, pp. 1572–1583.

Shao, Taihua et al. (2019). "Transformer-based neural network for answer selection in question answering". In: *IEEE Access* 7, pp. 26146–26156.

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

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

Sonata, I et al. (2021). "Autonomous car using CNN deep learning algorithm". In: *Journal of Physics: Conference Series*. Vol. 1869. 1. IOP Publishing, p. 012071.

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

Sun, Jianing et al. (2021). "Hgcf: Hyperbolic graph convolution networks for collaborative filtering". In: *Proceedings of the Web Conference 2021*, pp. 593–601.

Tay, Yi et al. (2020). "Sparse sinkhorn attention". In: *International Conference on Machine Learning*. PMLR, pp. 9438–9447.

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

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

Väisälä, Jussi (2005). "Gromov hyperbolic spaces". In: *Expositiones Mathematicae* 23.3, pp. 187–231.

Vaswani, A (2017). "Attention is all you need". In: *Advances in Neural Information Processing Systems*.

Vermeer, J (2005). "A geometric interpretation of Ungar's addition and of gyration in the hyperbolic plane". In: *Topology and its Applications* 152.3, pp. 226–242.

Wang, Hao et al. (2021). "Hypersorec: Exploiting hyperbolic user and item representations with multiple aspects for social-aware recommendation". In: *ACM Transactions on Information Systems (TOIS)* 40.2, pp. 1–28.

Wang, Qiang et al. (2019). "Learning deep transformer models for machine translation". In: *arXiv preprint arXiv:1906.01787*.

Wang, Yijia and Qiaotong Zhao (2022). "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)*. Atlantis Press, pp. 1547–1555.

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

Yang, Menglin et al. (2022). "HRCF: Enhancing collaborative filtering via hyperbolic geometric regularization". In: *Proceedings of the ACM Web Conference 2022*, pp. 2462–2471.

Yun, Chulhee et al. (2019). "Are transformers universal approximators of sequence-to-sequence functions?" In: *arXiv preprint arXiv:1912.10077*.

Yun, Chulhee et al. (2020). *Are Transformers universal approximators of sequence-to-sequence functions?* arXiv: 1912.10077 [cs.LG]. URL: https://arxiv.org/abs/1912.10077.

Zaheer, Manzil et al. (2020). "Big bird: Transformers for longer sequences". In: *Advances in neural information processing systems* 33, pp. 17283–17297.

Zhang, Hanqing et al. (2023). "A survey of controllable text generation using transformer-based pre-trained language models". In: *ACM Computing Surveys* 56.3, pp. 1–37.

Zhou, Ding-Xuan (2018). "Deep distributed convolutional neural networks: Universality". In: *Analysis and applications* 16.06, pp. 895–919.

— (2020a). "Universality of deep convolutional neural networks". In: *Applied and computational harmonic analysis* 48.2, pp. 787–794.

— (2020b). "Universality of deep convolutional neural networks". In: *Applied and computational harmonic analysis* 48.2, pp. 787–794.