Theoretical Proposal: Hyperbolic Gated Delta Networks (HGDN) - Ultra-big memory of hyperbolic space

理论提案：双曲门控 Delta 网络 (HGDN) - 双曲空间的超大记忆

Vuk Rosić
vukrosic

1. Abstract

We propose **Hyperbolic Gated Delta Networks (HGDN)**, a theoretical Linear Transformer architecture designed to integrate non-Euclidean geometry with hardware-efficient parallel training. By reformulating the Gated Delta Rule as a sequence of manifold-constrained updates on a product of Poincaré balls, HGDN is theorized to exploit the exponential volume of hyperbolic space to eliminate memory collisions in ultra-long contexts.

1. 摘要

我们提出了 **双曲门控 Delta 网络 (HGDN)**，这是一种理论上的线性 Transformer 架构，旨在将非欧几里得几何与硬件高效的并行训练相结合。通过将门控 Delta 规则重新表述为庞加莱球乘积上的一系列流形约束更新，理论上 HGDN 利用双曲空间的指数级体积来消除超长上下文中的记忆冲突。

2. The Core Intuition

Language and logic are inherently hierarchical (animal $\rightarrow$ mammal $\rightarrow$ dog), often branching like a tree. Euclidean space (used in Mamba2/DeltaNet) is “flat” and struggles to store many branches without them overlapping and blurring (memory collisions).

Hyperbolic space is naturally curved like a saddle, where the “available room” grows exponentially as you move away from the center. HGDN treats its hidden state as a point in this saddle-shaped space. When storing a new memory (a needle in the haystack), it pushes that memory toward the “edge” of the space where there is infinite room to keep it distinct. When it needs to forget, it pulls the state back toward the center. This “radial memory management" is hypothesized to make HGDN essentially collision-free for sequences exceeding 1 million tokens.

2. 核心直觉

语言和逻辑本质上是分层的（动物 → 哺乳动物 → 狗），通常像树一样分支。欧几里得空间（用于 Mamba2/DeltaNet）是"平坦"的，很难在该空间存储许多分支而不让它们重叠和模糊（记忆冲突）。

双曲空间自然弯曲像马鞍，当你远离中心时，"可用空间"呈指数级增长。HGDN 将其隐藏状态视为这个马鞍形空间中的一个点。当存储新记忆（大海捞针）时，它将该记忆推向空间的"边缘"，那里有无限的空间来保持它的独特性。当需要遗忘时，它将状态拉回中心。这种"径向记忆管理"被假设为使 HGDN 在超过 100 万个 token 的序列中基本上无冲突。

3. Mathematical Framework

3. 数学框架

3.1 Manifold Representation: The Product Poincaré Space

Instead of a flat Euclidean matrix, the HGDN state S_t is theorized to reside on a **product manifold** $\mathcal{M}$:

$$\mathcal{M} = \underbrace{\mathbb{B}_c^{d_k} \times \mathbb{B}_c^{d_k} \times \cdots \times \mathbb{B}_c^{d_k}}_{h \text{ times}}$$

3.1 流形表示：乘积庞加莱空间

HGDN 状态 S_t 不是平坦的欧几里得矩阵，而是理论上驻留在 **乘积流形** $\mathcal{M}$ 上：

$$\mathcal{M} = \underbrace{\mathbb{B}_c^{d_k} \times \mathbb{B}_c^{d_k} \times \cdots \times \mathbb{B}_c^{d_k}}_{h \text{ 次}}$$

Formula Breakdown:

1. **The Poincaré Ball** ($\mathbb{B}_c^d$): Defined as the set of points $\{x \in \mathbb{R}^d : c\|x\|^2 < 1\}$.

- x: A point (vector) representing a specific memory or hidden state value.
- $\mathbb{R}^d$: The standard d-dimensional space we use for calculations (vector x has d dimensions).
- c: The **Curvature**. It determines how "sharp" the saddle shape is. If $c = 0$, it's flat Euclidean space.
- $\|x\|^2 < 1/c$: This defines the boundary. The ball has a radius of $1/\sqrt{c}$. Points cannot "leave" this radius; instead, as they approach the edge, the space itself stretches to infinity.

公式分解：

1. **庞加莱球** ($\mathbb{B}_c^d$)：定义为点集 $\{x \in \mathbb{R}^d : c\|x\|^2 < 1\}$。

- x：代表特定记忆或隐藏状态值的点（向量）。
- $\mathbb{R}^d$：我们用于计算的标准 d 维空间（向量 x 有 d 个维度）。
- c：**曲率**。它决定了马鞍形状的"锐利"程度。如果 $c = 0$，它是平坦的欧几里得空间。
- $\|x\|^2 < 1/c$：这定义了边界。球的半径为 $1/\sqrt{c}$。点不能"离开"这个半径；相反，当它们接近边缘时，空间本身延伸到无限大。

Concept	**What it is**	**Role in HGDN**
Hyperbolic Space	The "True Reality"	The curved, infinite-volume geometry where memories live.
Saddle / Ball	Models (Maps)	Two different ways to map the same d-dimensional reality. The **Ball** is preferred because it fits neatly into $[-1, 1]$ coordinate
State Matrix (S_t)	Group of Vectors	A collection of h vectors. Each vector is a point (a "memory") tucked into its own Poincaré ball.

概念	**它是什么**	**在 HGDN 中的作用**
双曲空间	"真实实像"	记忆存在的弯曲、无限体积的几何结构。
马鞍 / 球	模型（地图）	映射同一 d 维实像的两种不同方式。 **球**更受欢迎，因为它整齐地适应 $[-1, 1]$ 坐标。
状态矩阵 (S_t)	向量组	h 个向量的集合。每个向量都是一个点（一个"记忆"）， 被塞进它自己的庞加莱球中。

Analogy: If the "Hyperbolic Space" is the Earth, the **Ball** and the **Saddle** are just two different map projections (like Mercator vs. Globe). The **Vectors** (your hidden state) are the actual cities pinned on those maps. Everything here—the space, the maps, and the vectors—has d **dimensions**.

类比：如果“双曲空间”是地球，那么 **球** (Ball) 和 **马鞍** (Saddle) 只是两种不同的地图投影（像墨卡托投影与地球仪）。**向量**（你的隐藏状态）是地图上标记的实际城市。这里的一切——空间、地图和向量——都有 d **个维度**。

2. **The Product Manifold** ($\mathcal{M}$):

   - $\times$ **(Cartesian Product)**: This means “concatenation of independent spaces.”
   - d_k: The dimensionality of each individual head (the “key/value dimension”).
   - h **(The number of balls)**: Corresponds to the number of value-heads in the Linear Transformer.
     - **Why separate balls?**: One hyperbolic space is essentially one **tree**. By giving each head its own ball, we allow the model to learn a **“Forest of Trees.”** This prevents a hierarchy in one head (like grammar rules) from colliding or interfering with a hierarchy in another head (like factual relationships). It ensures that each head can specialize in its own independent “semantic branch.”

3. **乘积流形** ($\mathcal{M}$):

   - $\times$ **(笛卡尔积)**：这意味着“独立空间的串联”。
   - d_k：每个单独头的维度（“键/值维度”）。
   - h **(球的数量)**：对应于线性 Transformer 中值头 (value-heads) 的数量。
     - **为什么要分开的球?**：一个双曲空间本质上是 **一棵树**。通过给每个头自己的球，我们允许模型学习 **“树的森林”**。这可以防止一个头中的层次结构（如语法规则）与另一个头中的层次结构（如事实关系）发生冲突或干扰。它确保每个头都可以专注于自己独立的“语义分支”。

- **The Infinite Boundary**: While the ball looks bounded to our Euclidean eyes, the hyperbolic distance to the boundary is actually infinite. This provides a “bottomless” storage area; as we push memories toward the shell, they become mathematically farther apart from each other, even if they appear close in Euclidean coordinates.
- **Product Domain Advantage**: By treating each **head** (which corresponds to a row s_i in the state matrix) as an independent point in its own ball, we allow the model to learn a **high-dimensional product of trees**. This means HGDN doesn't just store

one hierarchy, but hundreds of overlapping, independent hierarchies (e.g., one head for syntax, one for semantic clusters, one for temporal ordering).

- **Geometric Retrieval**: In standard Euclidean attention, deep branches of a tree "crowd" together because the space is too small. This causes **Branch Interference**: a query for a specific detail (a leaf) accidentally has high similarity with an unrelated detail in a neighboring branch, leading to a "blurry" or mixed retrieval.
  - **In HGDN**: The standard dot-product is conceptually replaced by seeking the point on the manifold that minimizes the **geodesic distance**. Because hyperbolic space expands exponentially at the edges, the mathematical "gap" between unrelated branches is massive. This ensures the model pulls exactly the right "needle" without any signal leakage from neighboring branches.

- **无限边界**：虽然球在我们的欧几里得眼睛看来是有界的，但到边界的双曲距离实际上是无限的。这提供了一个"无底"的存储区域；当我们把记忆推向外壳时，它们在数学上彼此相距更远，即使它们在欧几里得坐标中看起来很近。
- **乘积域优势**：通过将每个 **头**（对应于状态矩阵中的一行 s_i）视为其自身球中的独立点，我们允许模型学习 **高维树的乘积**。这意味着 HGDN 不仅仅存储一个层次结构，而是存储数百个重叠、独立的层次结构（例如，一个头用于语法，一个用于语义聚类，一个用于时间排序）。
- **几何检索**：在标准的欧几里得注意力中，树的深层分支"挤"在一起，因为空间太小了。这导致 **分支干扰**：对特定细节（叶子）的查询意外地与相邻分支中不相关的细节具有高相似性，导致"模糊"或混合检索。
  - **在 HGDN 中**：标准点积在概念上被替换为寻找流形上最小化 **测地线距离**的点。因为双曲空间在边缘呈指数级扩展，不相关分支之间的数学"间隙"是巨大的。这确保模型准确地提取正确的"针"，没有任何来自相邻分支的信号泄漏。

3.2 Tangent Space Parallel Training (TSPT)

3.2 切空间并行训练 (TSPT)

The Problem: The "Associativity Gap"

In standard Linear Transformers (like Mamba or DeltaNet), we use an **Associative Scan** to calculate the hidden state for a million tokens in parallel.

What is an Associative Scan?
Normally, a memory S_t is calculated one step at a time: $S_t = S_{t-1} +$new memory. This is slow ($O(L)$).
An **Associative Scan** is a parallel algorithm that calculates everything in a **tree structure**:

1. **Round 1**: Calculate $(1+2), (3+4), (5+6), (7+8)$ all at once.
2. **Round 2**: Combine those results: $((1+2)+(3+4))$ and $((5+6)+(7+8))$.
3. **Round 3**: Combine the final two chunks into the grand total.

Instead of 8 sequential steps, it finishes in just 3 "rounds." This logarithmic speed-up is what allows Linear Transformers to process 1 million tokens while staying faster than standard Attention. This only works if your "addition" is **associative** ($A + B + C = (A + B) + C$).

Hyperbolic space is NOT associative. If you try to add memories directly on a curve, the order matters too much (rotating a globe 90° North then 90° East is different from 90° East then 90° North). This breaks the parallel scan. Without TSPT, the model would be 100× slower to train.

问题："结合律缺口"

在标准线性 Transformer（如 Mamba 或 DeltaNet）中，我们使用 **关联扫描 (Associative Scan)** 并行计算 100 万个 token 的隐藏状态。

什么是关联扫描?
通常，记忆 S_t 是一步一步计算的：$S_t = S_{t-1} +$ 新记忆。这很慢 ($O(L)$)。
关联扫描是一种并行算法，以 **树结构**计算所有内容：

1. **第 1 轮**：一次性计算 $(1+2), (3+4), (5+6), (7+8)$。
2. **第 2 轮**：组合这些结果：$((1+2)+(3+4))$ 和 $((5+6)+(7+8))$。
3. **第 3 轮**：将最后两个块组合成总和。

它不是 8 个顺序步骤，而是仅用 3 "轮"就完成了。这种对数加速使得线性 Transformer 能够处理 100 万个 token，同时保持比标准 Attention 更快。这只有在你的"加法"满足 **结合律** ($A + B + C = (A + B) + C$) 时才有效。

双曲空间不满足结合律。如果你试图直接在曲线上添加记忆，顺序非常重要（先向北旋转地球 90° 再向东旋转 90°，与先向东 90° 再向北 90° 是不同的）。这打破了并行扫描。如果没有 TSPT，模型的训练速度将慢 100×。

The Geometric Logic: Tangent Planes

To fix this, we use a "Locally Euclidean" trick. Imagine a large globe (the hyperbolic ball). If you zoom in on one tiny patch, that patch looks flat (like a piece of paper touching the globe). This flat paper is the **Tangent Space**.

TSPT works by "flattening" a chunk of data onto this paper, doing the fast math there, and then "wrapping" the result back onto the globe.

几何逻辑：切平面

为了解决这个问题，我们使用"局部欧几里得"技巧。想象一个大地球仪（双曲球）。如果你放大其中一小块，那一块看起来是平的（像一张纸接触地球仪）。这张平坦的纸就是 **切空间**。

TSPT 的工作原理是将一块数据"压平"到这张纸上，在那里进行快速数学运算，然后将结果"包裹"回地球仪上。

Step-by-Step Breakdown:

For each sequence chunk (e.g., 2048 tokens), we do the following:

1. **Projection (The "Log" Map)**:

$$K_{tan} = \mathrm{Log}^c_{S_{[0]}}(K), \quad Q_{tan} = \mathrm{Log}^c_{S_{[0]}}(Q)$$

   - $S_{[0]}$: The starting state (the "anchor point") for this chunk.
   - $\mathbf{Log}^c_{S_{[0]}}$: The **Logarithm Map**. It "unrolls" the hyperbolic curve into a flat Euclidean plane centered at $S_{[0]}$.
   - K_{tan}, Q_{tan}: The Keys and Queries now live in a flat space where standard addition works.

2. **Euclidean Scan (The "Parallel Fast-Forward")**:

$$\Delta S_{tan} = \mathrm{AssociativeScan}(Q_{tan}, K_{tan}, V)$$

   - We perform the standard Gated Delta Rule update. Because we are in the flat Tangent Space, we can use optimized GPU kernels (like Flash-Linear-Attention) to process the whole chunk in $O(L)$ time.
   - ΔS_{tan}: The total "movement" or memory update calculated in flat space.

3. **Manifold Mapping (The "Exp" Map)**:

$$S_{[end]} = \text{Exp}^c_{S_{[0]}}(\Delta S_{tan})$$

- $\mathbf{Exp}^c_{S_{[0]}}$: The **Exponential Map**. It takes the flat result ΔS_{tan} and "wraps" it back onto the hyperbolic manifold.
- $S_{[end]}$: The final, valid hyperbolic state that becomes the starting point for the next chunk.

Why this is a breakthrough: It allows us to keep the massive memory capacity of Hyperbolic space while keeping the 10× training speed of Euclidean models.

逐步分解:

对于每个序列块（例如，2048 个 token），我们执行以下操作：

1. **投影（"对数" 映射）**:

$$K_{tan} = \text{Log}^c_{S_{[0]}}(K), \quad Q_{tan} = \text{Log}^c_{S_{[0]}}(Q)$$

- $S_{[0]}$：该块的起始状态（"锚点"）。
- $\mathbf{Log}^c_{S_{[0]}}$：**对数映射**。它将双曲曲线"展开"成以 $S_{[0]}$ 为中心的平坦欧几里得平面。
- K_{tan}, Q_{tan}：键和查询现在位于标准加法有效的平坦空间中。

2. **欧几里得扫描（"并行快进"）**:

$$\Delta S_{tan} = \text{AssociativeScan}(Q_{tan}, K_{tan}, V)$$

- 我们执行标准的门控 Delta 规则更新。因为我们在平坦的切空间中，我们可以使用优化的 GPU 内核（如 Flash-Linear-Attention）在 $O(L)$ 时间内处理整个块。
- ΔS_{tan}：在平坦空间中计算的总"移动"或记忆更新。

3. **流形映射（"指数" 映射）**:

$$S_{[end]} = \text{Exp}^c_{S_{[0]}}(\Delta S_{tan})$$

- $\mathbf{Exp}^c_{S_{[0]}}$：**指数映射**。它获取平坦结果 ΔS_{tan} 并将其"包裹"回双曲流形。
- $S_{[end]}$：最终的、有效的双曲状态，成为下一个块的起点。

为什么这是一个突破: 它允许我们保持双曲空间的巨大记忆容量，同时保持欧几里得模型 10× 的训练速度。

3.3 Stability: Hyperbolic Spectral Normalization (HSN)

To prevent the "Boundary Catastrophe" (numerical overflow as $\|x\| \to 1$), we propose a radial contraction after each state update:

$$S_t \leftarrow \tanh\left(\frac{R_{max}}{2} \cdot \frac{S_t}{\|S_t\|}\right)$$

This is intended to ensure the state matrix remains within a stable disk $\|x\| \leq 0.99$, guaranteeing differentiable gradients and training stability in FP16/BF16.

3.3 稳定性：双曲谱归一化 (HSN)

为了防止"边界灾难"（当 $\|x\| \to 1$ 时的数值溢出），我们建议在每次状态更新后进行径向收缩：

$$S_t \leftarrow \tanh\left(\frac{R_{max}}{2} \cdot \frac{S_t}{\|S_t\|}\right)$$

这旨在确保状态矩阵保持在稳定圆盘 $\|x\| \leq 0.99$ 内，保证 FP16/BF16 中的可微梯度和训练稳定性。

4. Summary

HGDN offers a novel synthesis of non-Euclidean geometry and parallel processing. By leveraging the exponential volume of hyperbolic space, it provides a theoretically collision-free memory for ultra-long contexts, while Tangent Space Parallel Training (TSPT) ensures it remains as fast as standard Linear Transformers on modern hardware.

4. 总结

HGDN 提供了非欧几里得几何与并行处理的新颖综合。通过利用双曲空间的指数级体积，它为超长上下文提供了理论上无冲突的记忆，而切空间并行训练 (TSPT) 确保它在现代硬件上保持与标准线性 Transformer 一样快。