HypTBench: Rethinking Hyperbolic Geometry for Transformers

Anonymous authors
Paper under double-blind review

Abstract

Hyperbolic geometry has been increasingly incorporated into Transformers to model hierarchical and relational structures. However, existing hyperbolic Transformers typically alter the geometric representation, the Transformer operators, and training configurations simultaneously, making it difficult to determine whether observed performance differences arise from the hyperbolic models themselves or from the geometric operators. To disentangle these factors, we introduce **HypTBench**, a unified benchmark that decomposes hyperbolic Transformers into interchangeable components and evaluates them under a consistent training protocol. Through **556** experimental runs covering 10 prevalent hyperbolic Transformers, we find that methods based on the Lorentz model generally outperform those on the Poincaré ball model in both accuracy and computational efficiency. Moreover, our component-level ablations reveal that compatibility among operators substantially affects overall performance. These findings suggest that the effectiveness of hyperbolic Transformers is determined not merely by the choice of hyperbolic representation, but by how hyperbolic geometry and fundamental Transformer operators are integrated into a compatible system.

1 Introduction

Transformers have become a foundational architecture across diverse domains, often implicitly assuming Euclidean geometry for the underlying data structure (Vaswani et al., 2017; Dosovitskiy et al., 2021; Dalla-Torre et al., 2025; He et al., 2025b). Real-world learning tasks can involve hierarchical relationships that motivate non-Euclidean representations (Nickel & Kiela, 2017; He et al., 2025b). Relevant examples include coarse-to-fine entity labels in language understanding (Ding et al., 2021; Xu et al., 2025), visual classes and superclasses in CIFAR-100 (Krizhevsky, 2009; Kasarla et al., 2025), and evolutionary relationships studied in genomic sequence modeling (Khan et al., 2025). Embedding tree-like hierarchies in low-dimensional Euclidean space can distort the distances between their nodes (Nickel & Kiela, 2017; Sala et al., 2018).

Hyperbolic geometry offers an inductive bias for these hierarchies through its exponential volume growth, which accommodates branching structures and enables low-distortion embeddings in few dimensions (Sala et al., 2018; van Spengler & Mettes, 2025). This property has motivated hyperbolic neural operators (Ganea et al., 2018; Chen et al., 2022) and their integration into Transformer architectures. Hyperbolic Transformers have been successfully applied to language understanding (Chen et al., 2024), visual representation learning and image classification (Ermolov et al., 2022; Fein-Ashley et al., 2024), and language modeling and reasoning (He et al., 2025a).

Existing hyperbolic Transformer methods broadly fall into three families: Euclidean-backbone, Poincaré-manifold, and Lorentz-manifold models (Figure 1 and Table 1). Euclidean-backbone methods introduce hyperbolic structure through output embeddings, training objectives, or selected attention computations (Ermolov et al., 2022; Tseng et al., 2023; He et al., 2024). Manifold-based methods maintain hidden representations on the Poincaré ball (Shimizu et al., 2021; Fein-Ashley et al., 2024) or the Lorentz model (Chen et al., 2022; Yang et al., 2024; He et al., 2025a), with method-specific geometric operators. These designs often change the manifold parameterization, attention, normalization, residual connections, and training configurations together. This coupling makes it difficult to attribute performance differences to the representation geometry or to the operators built on it.

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

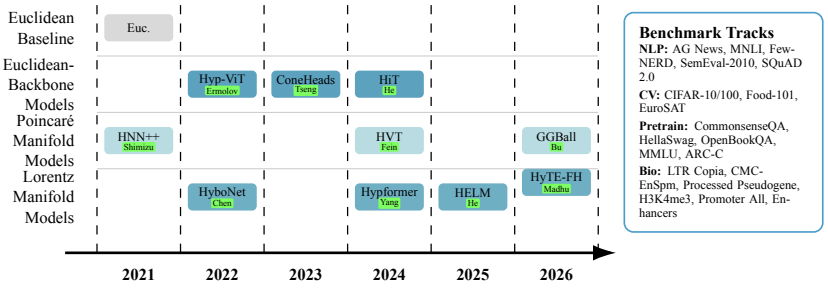

Figure 1: Overview of the collected hyperbolic Transformer methods and benchmark tracks. The timeline groups methods by model family and year; the right panel lists datasets across the four evaluation tracks.

Method	Space
Euclidean	Euclidean
HiT	Euclidean
ConeHeads	Euclidean
Hyp-ViT	Euclidean
HVT	Poincaré
GGBall	Poincaré
HNN++	Poincaré
HyboNet	Lorentz
HyTE-FH block	Lorentz
Hypformer	Lorentz
HELM	Lorentz

Table 1: Method taxonomy of HypTBench

To address these problems, we introduce **HypTBench**, the first unified benchmarking and analysis framework for hyperbolic Transformers. Our evaluation spans NLP, computer vision, pretrained reasoning, and genomic sequence modeling under a consistent training protocol, complemented by controlled coordinate and module-level ablations. Across **556** experimental runs, we find that (i) Lorentz-family methods attain the highest track-level accuracies; (ii) expressing hidden states in Poincaré coordinates while retaining Lorentz operators does not consistently reduce accuracy in our coordinate ablations; and (iii) the effects of operator substitutions depend on the base model and task, highlighting the importance of compatibility among Transformer components.

In summary, our main contributions are as follows:

- We introduce **HypTBench**. To the best of our knowledge, it is the first benchmark for hyperbolic Transformers with broad coverage across method families, tasks, and random seeds. HypTBench comprises **556** experimental runs.
- We reveal that existing Lorentz manifold methods generally exhibit stronger performance across tasks, whereas current Poincaré manifold methods suffer from performance degradation and higher computational overhead on multiple tasks.
- We further show that the design of fundamental Transformer operators and their mutual compatibility can substantially improve model performance through ablation studies.

2 Related Work

2.1 Hyperbolic Representation Learning

The central motivation of hyperbolic representation learning is to model hierarchical structures that are difficult to represent effectively in Euclidean space using negatively curved geometry. Poincaré embeddings demonstrated that hyperbolic space can efficiently organize tree-like and partially ordered structures in relatively low-dimensional representations (Nickel & Kiela, 2017). The Lorentz model was subsequently introduced as another widely used manifold formulation, offering advantages for modeling continuous hierarchies and improving numerical stability (Nickel & Kiela, 2018). Subsequent studies further incorporated such geometric structure directly into neural network components. Hyperbolic Neural Networks generalized fundamental operations, including linear transformations, nonlinear activations, and classifiers, to hyperbolic space (Ganea et al., 2018), while hyperbolic Graph Convolutional Neural Networks embedded node representations and message passing within a hyperbolic geometric framework to model graph data with hierarchical or tree-like topology (Chami et al., 2019). Despite these advances, it remains unclear which class of hyperbolic Transformer performs best under controlled task and experimental settings.

2.2 Hyperbolic Transformer

Transformers have become a general-purpose backbone for sequence modeling, graph representation learning, and large-scale pretraining (Vaswani et al., 2017; Devlin et al., 2019; Ying et al., 2021). Existing hyperbolic Transformer methods can be broadly categorized according to where Transformer

components operate in hyperbolic space. Some studies focus on the representation and network levels, emphasizing the migration of neural network modules as a whole into hyperbolic space. For example, HyboNet (Chen et al., 2022) and hyperbolic PLM (Chen et al., 2024) explore fully hyperbolic neural architectures and hyperbolic representations in pretrained language models, respectively. Another line of work directly modifies the attention mechanism or the modeling of hierarchical relations. ConeHeads (Tseng et al., 2023), for instance, employs hierarchy-aware attention to represent partial-order structures, whereas HiT (He et al., 2024) analyzes Transformer representations from the perspective of encoding hierarchical knowledge in language models. Other approaches target more complete architectures or specific application scenarios. HVT (Fein-Ashley et al., 2024) incorporates non-Euclidean representations into Vision Transformer architectures, Hypformer (Yang et al., 2024) seeks to construct more complete Transformer blocks in hyperbolic space, and Hyp-RAG (Madhu et al., 2026) applies hyperbolic retrieval to retrieval-augmented generation.

These studies demonstrate the diversity of ways in which hyperbolic geometry can be integrated with Transformers. However, different methods often vary simultaneously in architectural design, target tasks, and training protocols. As a result, existing empirical findings make it difficult to disentangle whether observed performance gains arise from the choice of hyperbolic geometry itself or from specific operator implementations and architectural designs. In this work, we place representative hyperbolic Transformer methods within a unified benchmark for systematic comparison and further analyze their design differences at the component level. Table 1 summarizes the taxonomy of hyperbolic Transformer methods considered in this study. **Detailed descriptions of the dataset sources and task settings are provided in Appendix B.**

3 Preliminaries

We use $c > 0$ to denote the magnitude of the shared sectional curvature $-c$ in the Lorentz model and the Poincaré ball.

Lorentz Model. The d-dimensional Lorentz model of hyperbolic geometry is defined as $\mathbb{H}^{d,c}_{\mathcal{L}} = \left\{x \in \mathbb{R}^{d+1} : \langle x, x\rangle_{\mathcal{L}} = -\frac{1}{c},\ x_0 > 0\right\}$, where the Lorentz inner product is given by $\langle x, y\rangle_{\mathcal{L}} = -x_0 y_0 + \sum_{i=1}^{d} x_i y_i$. Under this model, the hyperbolic distance between two points can be expressed as $d^c_{\mathcal{L}}(x, y) = \frac{1}{\sqrt{c}} \operatorname{arcosh}\left(-c\langle x, y\rangle_{\mathcal{L}}\right)$.

A practical advantage of the Lorentz model is its favorable numerical stability. Many hyperbolic Transformer methods construct the spatial coordinates from Euclidean features $u \in \mathbb{R}^d$ and subsequently derive the corresponding time coordinate: $\Phi^c_{\mathcal{L}}(u) = \left(\sqrt{\frac{1}{c} + \|u\|_2^2}, u\right)$, thereby ensuring that the resulting point satisfies the Lorentz constraint.

Poincaré Ball. The Poincaré model represents the same hyperbolic space of curvature $-c$ on an open ball of Euclidean radius $1/\sqrt{c}$: $\mathbb{B}^{d,c} = \left\{x \in \mathbb{R}^d : c\|x\|_2^2 < 1\right\}$. In the Poincaré ball, points closer to the boundary correspond to increasingly fine-grained concepts. Given Möbius addition $\oplus_c$, the hyperbolic distance in the Poincaré ball is defined as $d^c_{\mathbb{B}}(x, y) = \frac{2}{\sqrt{c}} \operatorname{artanh}\left(\sqrt{c}\,\|(-x) \oplus_c y\|_2\right)$. Here, $x \oplus_c y = \frac{\left(1+2c\langle x,y\rangle+c\|y\|_2^2\right)x+\left(1-c\|x\|_2^2\right)y}{1+2c\langle x,y\rangle+c^2\|x\|_2^2\|y\|_2^2}$.

4 Benchmarking

4.1 Compared Methods

We compare a Euclidean Transformer baseline with ten representative hyperbolic Transformer variants: HiT (He et al., 2024), ConeHeads (Tseng et al., 2023), hyperbolic Vision Transformer (Ermolov et al., 2022), HVT (Fein-Ashley et al., 2024), GGBall (Bu et al., 2026), HNN++ (Shimizu et al., 2021), HyboNet (Chen et al., 2022), Hyp-RAG (Madhu et al., 2026), Hypformer (Yang et al., 2024), and HELM (He et al., 2025a).

For clarity and to distinguish it from HVT (Fein-Ashley et al., 2024), we abbreviate hyperbolic Vision Transformer (Ermolov et al., 2022) as Hyp-ViT throughout the remainder of this paper. In addition, because HypTBench is not designed as a benchmark for retrieval-augmented generation, we do not

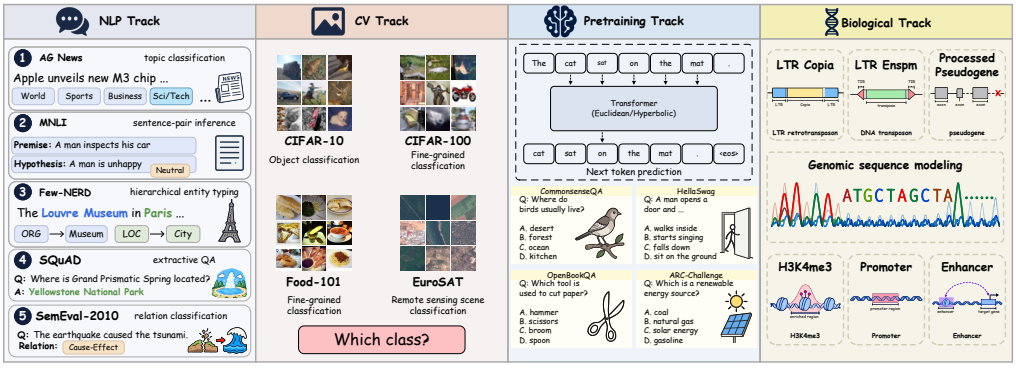

Figure 2: Overview of HypTBench. The benchmark spans four tracks: NLP, CV, pretrained reasoning, and genomic sequence modeling. Each track comprises multiple representative tasks of different types, enabling a cross-domain evaluation of the performance of different geometric Transformer designs.

evaluate the complete Hyp-RAG framework (Madhu et al., 2026). Instead, we incorporate only its HyTE-FH block as a representative hyperbolic Transformer component. Accordingly, we refer to this component as HyTE-FH rather than Hyp-RAG in the remainder of the paper.

4.2 Tasks and Evaluation Metrics

Tasks. The benchmark covers four categories of tasks: natural language processing, computer vision, pretrained reasoning, and genomic sequence modeling.

In the main text, we report compact track-level accuracy summaries, while complete results with method-level F1 and accuracy are provided in Appendix E. All tabulated scores are presented on a 0-100 scale. Macro averages are computed over the four track-level results, with OOT entries counted as zero.

Evaluation protocol. We adapt each method to a shared interface with track-specific input adapters and task heads (Appendix C). Within each track and dataset, training settings are held fixed wherever possible. Main results report mean and standard deviation over three seeds (42, 1337, and 3407). OOT denotes runs exceeding the 24-hour training budget; assigning these entries zero in the macro average makes the aggregate score reflect both predictive performance and completion within this budget. **Further implementation and hardware details are provided in Appendix D.**

5 Experimental Results

5.1 Main Results

Table 2 summarizes the main results as track-level Avg. Acc. The strongest methods are concentrated in the Lorentz family, but the leading method differs across tracks. On NLP, the HyTE-FH block obtains the best average accuracy, 68.14±0.34, only slightly above Hyp-ViT at 68.08±0.32.

The advantage of Lorentz methods becomes clearer on CV, Pretraining, and Bio. Hypformer leads the CV Track with 69.29±0.09 and the Biological Sequence Track with 71.37±0.20, while the HyTE-FH block leads the Pretraining Track with 25.03±0.21. These best scores exceed the strongest Euclidean-backbone alternatives in the same tracks: Hyp-ViT reaches 65.97±0.21 on CV and 23.71±0.11 on Pretraining, and HiT reaches 69.79±0.30 on Bio. In contrast, Poincaré-manifold methods remain consistently weaker in this summary. GGBall is the strongest Poincaré method on NLP, CV, and Bio, but its track averages remain below the leading Euclidean-backbone and Lorentz models; on Pretraining, both GGBall and HNN++ are OOT.

Table 2: Overall results across HypTBench tracks. Higher is better. Parentheses denote the number of datasets. Red marks the best entry in each track. Avg. averages the four track-level accuracy columns and treats OOT as zero; Wins counts best dataset-metric cells, with ties credited to every tied method. **Full experimental results are provided in Appendix E.**

Method family	Method	NLP (5)	CV (4)	Pretrain (5)	Bio (6)	Avg.	Wins
Euclidean baseline	Euclidean (Vaswani et al., 2017)	67.46±0.12	67.80±0.09	23.28±0.14	69.30±0.11	56.96	0
Euclidean-backbone	HiT (He et al., 2024)	67.30±0.37	65.65±0.21	23.30±0.15	69.79±0.30	56.51	1
	ConeHeads (Tseng et al., 2023)	67.79±0.31	65.21±0.05	23.51±0.06	69.62±0.27	56.53	1
	Hyp-ViT (Ermolov et al., 2022)	68.08±0.32	65.97±0.21	23.71±0.11	67.45±0.14	56.30	7
Poincaré manifold	HVT (Fein-Ashley et al., 2024)	61.99±0.10	53.02±0.17	23.36±0.06	67.08±0.09	51.36	0
	GGBall (Bu et al., 2026)	64.48±0.20	60.39±0.05	OOT	68.71±0.11	48.40	0
	HNN++ (Shimizu et al., 2021)	62.95±0.22	50.19±0.15	OOT	68.32±0.14	45.37	0
Lorentz manifold	HyboNet (Chen et al., 2022)	66.26±0.04	68.58±0.14	23.62±0.14	68.03±0.11	56.62	5
	HyTE-FH block (Madhu et al., 2026)	**68.14±0.34**	53.79±0.26	**25.03±0.21**	71.26±0.48	54.56	5
	Hypformer (Yang et al., 2024)	67.89±0.30	**69.29±0.09**	24.25±0.10	**71.37±0.20**	58.20	17
	HELM (He et al., 2025a)	66.60±0.11	59.50±0.12	23.44±0.11	68.08±0.08	54.41	1

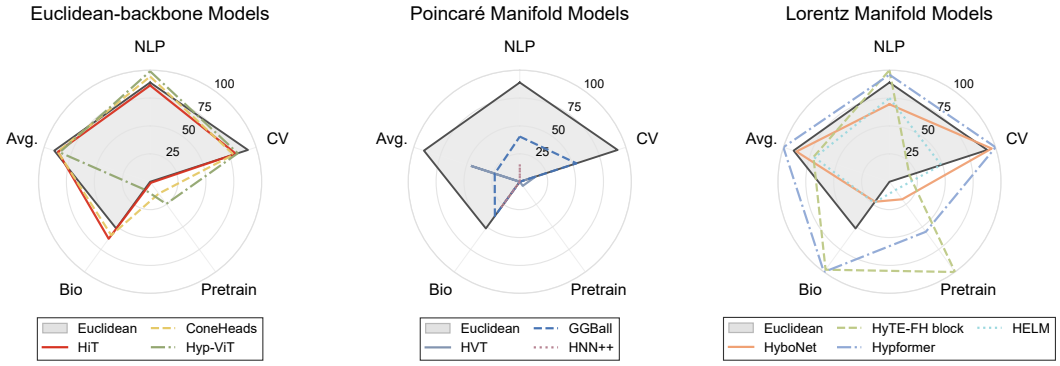

Figure 3: Cross-track **relative performance** of Euclidean-backbone, Poincaré manifold, and Lorentz manifold methods. Scores are min-max normalized over completed numeric entries on each axis; OOT entries are assigned a relative score of 0 on the corresponding axis.

At the macro level, Hypformer achieves the highest Avg. of 58.20 and the largest number of wins, with 17 best dataset-metric cells. The family-level statistics in Table 3 tell the same story: Lorentz-manifold methods account for 28 wins, compared with 9 for Euclidean-backbone methods and 0 for Poincaré-manifold methods. The method-level Top-2 and Gap Std. statistics further suggest that Hypformer is not only frequently competitive, but also has the most stable gap to the best method across available cells. Figure 3 visualizes this cross-dataset pattern using relative scores, where each axis is normalized between the weakest and strongest available method.

Beyond predictive performance, the different geometric implementations also exhibit substantial differences in computational efficiency. As shown in Figure 4, the task-level cost profile is not uniform across datasets: some methods retain competitive scores with moderate duration and memory, while others require substantially longer runs or larger peak memory for comparable or weaker task scores. This suggests that the observed efficiency gap cannot be explained solely by model size or by a single track-level runtime statistic. Additional task-specific efficiency measurements are provided in Appendix E.2.

5.2 Geometric Analysis

5.2.1 Does Coordinate Representation Explain the Gap?

The main experiments reveal a clear performance gap between existing Lorentz-family and Poincaré-family methods. This naturally raises an important question: is this gap primarily caused by the hyperbolic coordinate representations adopted by the two families? To investigate this issue, we keep

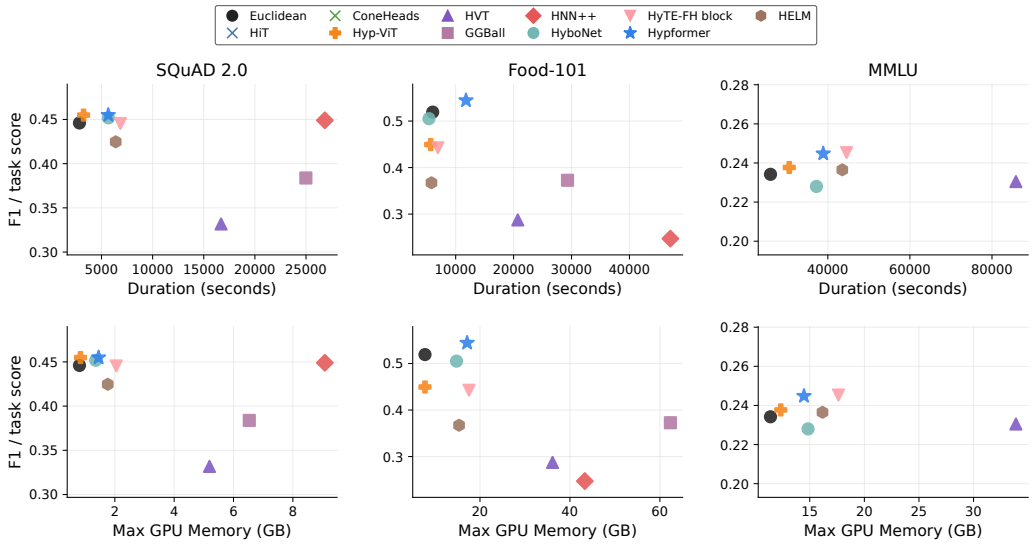

Figure 4: Task-level efficiency-performance trade-offs on SQuAD 2.0, Food-101, and MMLU. The top row plots the reported task score against training duration in seconds, and the bottom row plots the same score against peak GPU memory in GB. Colors and markers denote methods. The corresponding Promoter All results are provided in Appendix E.2.

Table 3: SOTA-count summary by method family and method-level aggregate statistics. Top-2 Count includes both best and second-best results; Gap Std. is the standard deviation of $s_a^* - s_{m,a}$ over each method's available cells, where lower values indicate a more stable gap to the best method.

(a) SOTA counts by method family.

Family	Methods	Method Wins	Family Wins
Euclidean baseline	Euclidean	0	0
Euclidean-backbone	HiT	1	9
	ConeHeads	1	
	Hyp-ViT	7	
Poincaré manifold	HVT	0	0
	GGBall	0	
	HNN++	0	
Lorentz manifold	HyboNet	5	28
	HyTE-FH block	5	
	Hypformer	17	
	HELM	1	

(b) Method-level top-2 and stability statistics.

Method	Top-2 Count↑	Gap Std.↓
Euclidean	5	1.90
HiT	4	2.29
ConeHeads	5	2.38
Hyp-ViT	7	11.50
HVT	0	6.98
GGBall	0	4.14
HNN++	3	9.68
HyboNet	10	2.98
HyTE-FH block	14	9.70
Hypformer	23	1.35
HELM	1	4.38

Table 4: Poincaré Ablation Study Results

Method	CIFAR-100		Food-101	
	F1	Acc	F1	Acc
HyboNet	51.26 ↑+2.04	51.48 ↑+2.46	53.49 ↑+3.06	53.72 ↑+3.26
Hypformer	51.22 ↑+1.31	51.42 ↑+1.35	52.21 ↓-2.30	52.37 ↓-2.46

the original internal designs of HyboNet and Hypformer unchanged and modify only the representation of the intermediate Transformer hidden states to use Poincaré coordinates. Whenever a Lorentz geometric operator is invoked, the hidden states are converted back to Lorentz coordinates.

The results are reported in Table 4. Replacing the coordinate representation in this manner does not lead to a consistent performance degradation. On the contrary, HyboNet improves its accuracy on CIFAR-100 and Food-101 by 2.46 and 3.26 percentage points, respectively, while Hypformer also achieves a 1.35 percentage-point improvement on CIFAR-100. These results suggest that the

Table 5: Module-level agreement with Hypformer among Lorentz Transformer variants. Check marks indicate that a method uses the same module choice as Hypformer, while crosses indicate a different design.

Method	Proj.	Attn. score	Value agg.	Residual	Norm	FFN
HyboNet	✗	✓	✓	✓	✓	✗
HyTE-FH block	✗	✗	✓	✗	✓	✗
Hypformer	✓	✓	✓	✓	✓	✓
HELM	✗	✗	✓	✗	✗	✗

Poincaré representation itself is not responsible for the observed performance degradation. Instead, the gap is more likely associated with the specific geometric operators and Transformer computation schemes coupled with these representations. Motivated by this observation, in the next section we fix the Lorentz-manifold family and conduct module-level ablations to examine how different Transformer operators affect model performance.

5.2.2 How Much Do Transformer Operators Matter?

Because the performance differences among Lorentz-manifold methods are particularly pronounced on CIFAR-100 and Food-101 in the CV Track, we conduct the ablation study primarily on these two datasets to further identify which components account for Hypformer's performance advantage. Specifically, we decompose a Lorentz Transformer into six core components: Projection, Attention Score, Value Aggregation, Residual, Normalization, and FFN. We then transplant the corresponding Hypformer design for each component into HyboNet, the HyTE-FH block, and HELM, while keeping all method-specific structures unchanged. The results are reported in Table 6; Figure 5 further visualizes the average accuracy gain of each module transplant over the two CV datasets, making the direction and magnitude of the module effects easier to compare across base models.

The results first show that Hypformer's advantage cannot be attributed to any single geometric operator. Instead, it arises from the combined effect of multiple components, although their individual contributions to the final performance vary substantially. This pattern is particularly evident for the HyTE-FH block and HELM. For the HyTE-FH block, replacing the Projection module with the corresponding Hypformer design yields a Gain Sum of 27.65 points. A similar effect is observed for HELM. Its original accuracy is only 41.91% on CIFAR-100 and 36.59% on Food-101, whereas replacing the Projection module alone raises these values to 49.46% and 46.57%, corresponding to gains of 7.55 and 9.98 percentage points, respectively.

Beyond Projection, modifications to the other components also yield substantial performance improvements in most cases. The effect of replacing the Residual module depends on the base model and task. For the HyTE-FH block, the replacement improves F1 and accuracy by 0.66 and 0.75 percentage points on CIFAR-100, respectively, and by 4.09 and 3.60 percentage points on Food-101. HELM exhibits a collapse in learning dynamics on Food-101 after the same module replacement.

Moreover, the benefits of Hypformer components do not transfer uniformly across all Lorentz-manifold methods. As shown in Table 5, the main architectural differences in HyboNet are concentrated in the Projection and FFN modules. Replacing only the Projection module with the Hypformer design increases CIFAR-100 accuracy from 49.02% to 51.52%, even surpassing the original Hypformer, but decreases Food-101 accuracy from 50.46% to 45.01%. Consequently, the overall benefit of this isolated replacement remains limited. Similarly, replacing only the FFN leads to accuracy drops of 1.26 and 1.32 percentage points on CIFAR-100 and Food-101, respectively.

In contrast, when both Projection and FFN are replaced simultaneously, resulting in an architecture that is nearly identical to Hypformer, the Food-101 accuracy increases to 53.86%, representing a 3.40 percentage-point improvement over the original HyboNet and approaching the 54.83% achieved by the full Hypformer model.

The interaction between components is also visible quantitatively. For HyboNet on Food-101, replacing only Projection decreases accuracy by 5.45 percentage points, and replacing only FFN decreases it by 1.32 percentage points. If the two effects were additive, the combined replacement would be expected to decrease accuracy by 6.77 percentage points. Instead, replacing both Projec-

Table 6: Ablation design matrix for Lorentz Transformer modules. Check marks indicate Hypformer-style module choices, while crosses indicate that the module keeps the corresponding base method's non-Hypformer design. Colored arrows report changes relative to the corresponding original base method, with red upward arrows for gains and blue downward arrows for drops. Gain Sum sums all positive point changes in the four result cells; drops and collapsed runs are not added.

Base	Variant	Proj.	Attn. score	Value agg.	Residual	Norm	FFN	CIFAR-100		Food-101		Gain Sum↑
								F1	Acc	F1	Acc	
HyboNet	original	✗	✓	✓	✓	✓	✗	49.22	49.02	50.43	50.46	–
HyTE-FH block	original	✗	✗	✓	✗	✓	✗	42.80	43.14	43.91	44.32	–
HELM	original	✗	✗	✓	✗	✗	✗	41.74	41.91	36.60	36.59	–
Hypformer	original	✓	✓	✓	✓	✓	✓	49.91	50.07	54.51	54.83	–
HyboNet	+ Hypformer Proj.	✓	✓	✓	✓	✓	✗	51.40 ↑+2.18	51.52 ↑+2.50	44.73 ↓-5.70	45.01 ↓-5.45	4.68
HyboNet	+ Hypformer FFN	✗	✓	✓	✓	✓	✓	47.46 ↓-1.76	47.76 ↓-1.26	49.06 ↓-1.37	49.14 ↓-1.32	0.00
HyboNet	+ Hypformer Proj.+FFN	✓	✓	✓	✓	✓	✓	49.82 ↑+0.60	49.78 ↑+0.76	53.95 ↑+3.52	53.86 ↑+3.40	8.28
HyTE-FH block	+ Hypformer Proj.	✓	✗	✓	✗	✓	✗	48.42 ↑+5.62	48.50 ↑+5.36	52.42 ↑+8.51	52.48 ↑+8.16	27.65
HyTE-FH block	+ Hypformer Attn. score	✗	✓	✓	✗	✓	✗	47.90 ↑+5.10	48.33 ↑+5.19	51.40 ↑+7.49	51.43 ↑+7.11	24.89
HyTE-FH block	+ Hypformer Residual	✗	✗	✓	✓	✓	✗	43.46 ↑+0.66	43.89 ↑+0.75	48.00 ↑+4.09	47.92 ↑+3.60	9.10
HyTE-FH block	+ Hypformer FFN	✗	✗	✓	✗	✓	✓	49.17 ↑+6.37	48.85 ↑+5.71	47.61 ↑+3.70	47.94 ↑+3.62	19.40
HELM	+ Hypformer Proj.	✓	✗	✓	✗	✗	✗	49.35 ↑+7.61	49.46 ↑+7.55	46.47 ↑+9.87	46.57 ↑+9.98	35.01
HELM	+ Hypformer Attn. score	✗	✓	✓	✗	✗	✗	50.79 ↑+9.05	50.93 ↑+9.02	46.77 ↑+10.17	46.86 ↑+10.27	38.51
HELM	+ Hypformer Residual	✗	✗	✓	✓	✗	✗	42.61 ↑+0.87	42.84 ↑+0.93	Collapse	Collapse	1.80
HELM	+ Hypformer Norm	✗	✗	✓	✗	✓	✗	50.70 ↑+8.96	50.92 ↑+9.01	46.30 ↑+9.70	46.30 ↑+9.71	37.38
HELM	+ Hypformer FFN	✗	✗	✓	✗	✗	✓	50.43 ↑+8.69	50.43 ↑+8.52	46.98 ↑+10.38	47.14 ↑+10.55	38.14

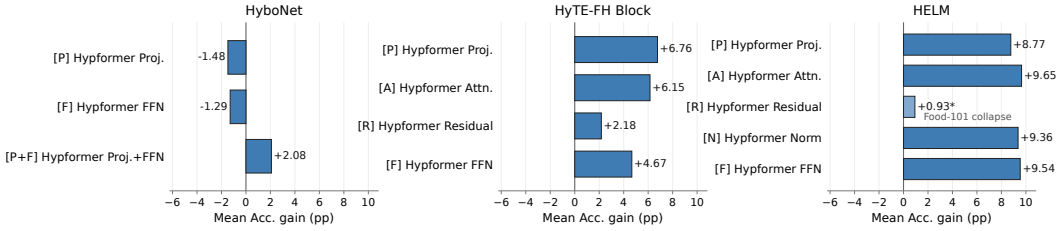

Figure 5: Average accuracy gain from transplanting Hypformer-style modules. Each horizontal bar reports the mean accuracy change on CIFAR-100 and Food-101 relative to the corresponding original base model. The asterisk indicates that the value only uses the completed dataset result because HELM + Hypformer Residual collapses on Food-101.

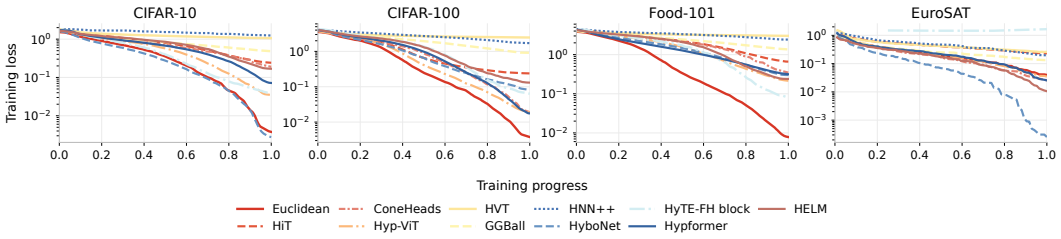

Figure 6: Training loss curves on CV tasks from seed-42 runs.

tion and FFN increases accuracy by 3.40 percentage points, yielding a positive interaction of 10.17 percentage points. On CIFAR-100, the same calculation gives an interaction of -0.48 percentage points, indicating that the compatibility effect is task-dependent rather than a uniform gain. This supports our central observation that hyperbolic Transformer performance depends not only on individual geometric operators, but also on how these operators interact within the full block. Figure 6 further reports the training loss curves on the corresponding CV tasks, providing an additional view of the optimization dynamics across methods.

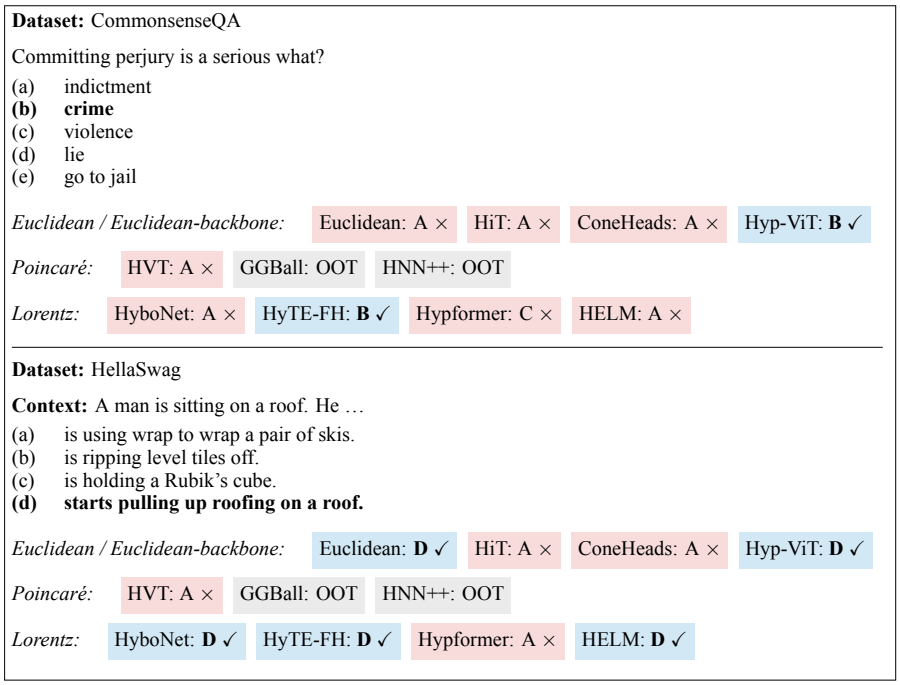

Figure 7: Qualitative comparison of different Transformer designs on representative examples from the Pretraining Track. The correct answer is shown in **bold**. Blue boxes indicate correct model predictions, red boxes indicate incorrect predictions, and gray boxes denote runs that exceeded the training-time budget (OOT).

5.3 Case Study

Figure 7 reports two representative Pretraining Track examples. On the CommonsenseQA question, only Hyp-ViT and the HyTE-FH block recover the correct answer, while the Euclidean baseline, HiT, ConeHeads, HVT, HyboNet, Hypformer, and HELM all select plausible but incorrect alternatives. The HellaSwag example shows a different pattern. Here the Euclidean baseline, Hyp-ViT, HyboNet, HyTE-FH block, and HELM all select the correct continuation, whereas HiT, ConeHeads, HVT, and Hypformer choose an implausible action.

6 Conclusion

This paper introduces HypTBench, the first unified benchmark and analysis framework for hyperbolic Transformers. Under consistent training protocols and evaluation settings, it systematically compares Euclidean-backbone, Poincaré-manifold, and Lorentz-manifold designs across multiple tasks. The unified evaluation shows that different hyperbolic implementations, even when mathematically equivalent in the underlying geometry, can exhibit substantially different performance and computational behavior inside Transformers. The results further indicate that the key to incorporating hyperbolic geometry into Transformers is whether the geometric representation and core Transformer operators form a stable and mutually compatible computational system. Future research on hyperbolic Transformers should therefore look beyond the choice of negative-curvature space and examine numerical implementation and the coordination among geometric operators. HypTBench provides a unified and reproducible basis for studying these questions and aims to support future geometry-aware Transformer design.

AI Use Statement

This paper did not use generative artificial intelligence tools or large language models for research conception, manuscript writing, or any other part of the workflow. All experiments, analyses, figures, textual expressions, and conclusions in the paper were completed and verified by the authors. The authors take full responsibility for the entire content of the paper.

References

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101: Mining discriminative components with random forests. In *Computer Vision – ECCV 2014*, pp. 446–461. Springer, 2014.

Tianci Bu, Chuanrui Wang, Hao Ma, Haoren Zheng, Xin Lu, and Tailin Wu. GGBall: Graph generative model on Poincaré ball. In *International Conference on Learning Representations*, volume 2026, pp. 39923–39959, 2026.

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

Weize Chen, Xu Han, Yankai Lin, Hexu Zhao, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. Fully hyperbolic neural networks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5672–5686, Dublin, Ireland, 2022. Association for Computational Linguistics.

Weize Chen, Xu Han, Yankai Lin, Kaichen He, Ruobing Xie, Jie Zhou, Zhiyuan Liu, and Maosong Sun. Hyperbolic pre-trained language model. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:3101–3112, 2024.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Hugo Dalla-Torre, Liam Gonzalez, Javier Mendoza-Revilla, Nicolas Lopez Carranza, Adam Henryk Grzywaczewski, Francesco Oteri, Christian Dallago, Evan Trop, Bernardo P. de Almeida, Hassan Sirelkhatim, Guillaume Richard, Marcin Skwark, Karim Beguir, Marie Lopez, and Thomas Pierrot. Nucleotide transformer: Building and evaluating robust foundation models for human genomics. *Nature Methods*, 22(2):287–297, 2025.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, 2019. Association for Computational Linguistics.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Haitao Zheng, and Zhiyuan Liu. Few-NERD: A few-shot named entity recognition dataset. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pp. 3198–3213, Online, 2021. Association for Computational Linguistics.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Aleksandr Ermolov, Leyla Mirvakhabova, Valentin Khrulkov, Nicu Sebe, and Ivan Oseledets. Hyperbolic vision transformers: Combining improvements in metric learning. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7409–7419. IEEE, 2022.

Jacob Fein-Ashley, Ethan Feng, and Minh Pham. HVT: A comprehensive vision framework for learning in non-euclidean space, 2024.

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

Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassiulas, Menglin Yang, and Rex Ying. HELM: Hyperbolic large language models via mixture-of-curvature experts. *Advances in Neural Information Processing Systems*, 38:142604–142635, 2025a.

Neil He, Jiahong Liu, Buze Zhang, Ngoc Bui, Ali Maatouk, Menglin Yang, Irwin King, Melanie Weber, and Rex Ying. Position: Beyond Euclidean – foundation models should embrace Non-Euclidean geometries. In *Proceedings of the Fourth Learning on Graphs Conference*, 2025b.

Yuan He, Zhangdie Yuan, Jiaoyan Chen, and Ian Horrocks. Language models as hierarchy encoders. In *Advances in Neural Information Processing Systems*, volume 37, pp. 14690–14711, 2024.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. EuroSAT: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. SemEval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pp. 33–38, Uppsala, Sweden, 2010. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021.

Tejaswi Kasarla, Ruthu Hulikal Rooparaghunath, Stefano D'arrigo, Gowreesh Mago, Abhishek Jha, Melika Ayoughi, Swasti Shreya Mishra, Ana Manzano Rodríguez, Teng Long, Mina Ghadimi Atigh, Max van Spengler, and Pascal Mettes. HierVision: Standardized and reproducible hierarchical sources for vision datasets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pp. 682–695, 2025.

Raiyan R. Khan, Philippe Chlenski, and Itsik Pe'er. Hyperbolic genome embeddings. In *International Conference on Learning Representations*, 2025.

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

Hiren Madhu, Ngoc Bui, Ali Maatouk, Leandros Tassiulas, Smita Krishnaswamy, Menglin Yang, Sukanta Ganguly, Kiran Srinivasan, and Rex Ying. HypRAG: Hyperbolic dense retrieval for retrieval augmented generation, 2026.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391, Brussels, Belgium, 2018. Association for Computational Linguistics.

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

Maximillian Nickel and Douwe Kiela. Learning continuous hierarchies in the Lorentz model of hyperbolic geometry. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3779–3788. PMLR, 2018.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pp. 784–789, Melbourne, Australia, 2018. Association for Computational Linguistics.

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

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

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4149–4158, Minneapolis, Minnesota, 2019. Association for Computational Linguistics.

Albert Tseng, Tao Yu, Toni Liu, and Christopher M. De Sa. Coneheads: Hierarchy aware attention. In *Advances in Neural Information Processing Systems*, volume 36, pp. 51421–51433, 2023.

Max van Spengler and Pascal Mettes. Low-distortion and GPU-compatible tree embeddings in Hyperbolic space. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 60792–60815. PMLR, 2025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, pp. 1112–1122, New Orleans, Louisiana, 2018. Association for Computational Linguistics.

Mingying Xu, Wenxuan Zhang, Jie Liu, Weiping Ding, Lei Shi, and Kaiyang Zhong. Fine-grained entity typing based on Hyperbolic representation and label-context interaction. *Information Sciences*, 719:122434, 2025.

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

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? In *Advances in Neural Information Processing Systems*, volume 34, pp. 28877–28888, 2021.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, Florence, Italy, 2019. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Contents

A Appendix Overview . . . 14

B Datasets . . . 14

B.1 NLP Track . . . 14

B.2 CV Track . . . 14

B.3 Pretraining Track . . . 15

B.4 Biological Sequence Track . . . 15

C Method Adaptation Details . . . 16

D Benchmark Protocol and Implementation Details . . . 18

D.1 Unified Training Protocol . . . 18

D.2 Hardware and Runtime . . . 18

E Additional Main Results . . . 18

E.1 Multi-Seed Results . . . 18

E.2 Additional Efficiency Results on Promoter All . . . 18

F Qualitative Analysis and Visualizations . . . 20

F.1 CV Classification Samples . . . 20

F.2 Pretraining Reasoning Samples . . . 21

F.3 Biological Sequence Case Studies . . . 21

F.4 Training Loss Curves . . . 22

A Appendix Overview

This appendix is organized around reproducibility details, complete results, and qualitative diagnostics. Appendix B first describes the datasets and task backgrounds covered by the benchmark across the NLP, CV, Pretraining, and Biological Sequence tracks. Appendix C describes method adaptation and model backbones, while Appendix D records the unified training protocol, hardware setting, and runtime statistics. Appendix E reports the full per-dataset results that are compressed in the main text, while Appendix E.1 provides the multi-seed mean and standard-deviation tables. Finally, Appendix F collects qualitative case studies for the CV, Pretraining, and Biological Sequence tracks, and Appendix F.4 supplements the cleaned training-loss curves used to diagnose optimization behavior.

B Datasets

B.1 NLP Track

AG News. AG News (Zhang et al., 2015) is a news topic classification dataset. Inputs usually consist of news titles and short descriptions, and the task is to classify each instance into one of several coarse-grained news categories. Because the class boundaries are relatively clear and the texts are short, AG News is suitable for testing basic discriminative ability in standard text classification.

MNLI. MNLI (Williams et al., 2018) is a natural language inference task in which the model must determine whether a given premise and hypothesis pair expresses entailment, neutrality, or contradiction.

Few-NERD Supervised. Few-NERD Supervised (Ding et al., 2021) is a named entity recognition task with a clear hierarchy between coarse-grained and fine-grained entity types. The model must not only identify entity spans, but also assign each entity to a more specific semantic type, making the task naturally suitable for evaluating whether hyperbolic space can capture label hierarchies.

SemEval-2010 Task 8. SemEval-2010 Task 8 (Hendrickx et al., 2010) is a sentence-level relation classification task whose goal is to identify the semantic relation between two marked entities in a sentence. The task requires the model to understand local entity context, entity-pair directionality, and semantic distinctions among relation types, and therefore emphasizes structured relation modeling more strongly than ordinary text classification.

SQuAD 2.0. SQuAD 2.0 (Rajpurkar et al., 2018) is an extractive question answering task in which the model must extract an answer span from a given passage and determine whether some questions are unanswerable.

B.2 CV Track

CIFAR-10. CIFAR-10 (Krizhevsky, 2009) is a ten-class natural image classification dataset. The images have low resolution and the number of classes is small, making it a foundational benchmark for visual classification.

CIFAR-100. CIFAR-100 (Krizhevsky, 2009) has the same image scale and resolution as CIFAR-10, but expands the label set to one hundred classes, whose fine-grained categories can be grouped into higher-level superclasses. Because its label structure is finer and inter-class similarity is stronger, CIFAR-100 is better suited for evaluating how models capture category hierarchies and fine-grained visual differences.

Food-101. Food-101 (Bossard et al., 2014) is a fine-grained food image classification dataset with many food categories that can look visually similar. The task is challenging because of intra-class appearance variation, inter-class visual similarity, and background or plating effects. Models therefore need to capture subtle differences in local texture, shape composition, and category semantics.

EuroSAT. EuroSAT (Helber et al., 2019) is a remote-sensing scene classification dataset built from Sentinel-2 satellite imagery. It covers land-use and land-cover categories such as residential areas, industrial areas, rivers, forests, and farmland. Unlike natural image classification, EuroSAT emphasizes texture distributions, object compositions, and spatial layouts from an overhead perspective, making it useful for testing whether hyperbolic Transformers can capture hierarchical and compositional category patterns in visual scenes.

B.3 Pretraining Track

CommonsenseQA. CommonsenseQA (Talmor et al., 2019) is a multiple-choice commonsense question answering task whose questions typically involve commonsense relations among concepts. The model must distinguish among candidate answers and use implicit concept-graph knowledge for reasoning. We use this dataset to evaluate whether geometric modules improve the organization of concept hierarchies and commonsense relations in pretrained models.

HellaSwag. HellaSwag (Zellers et al., 2019) is a multiple-choice grounded completion task in which the model selects the ending that best follows the event context. Candidate endings are usually grammatically fluent but differ in semantic plausibility, so the task requires careful reasoning about events, actions, and scenes.

OpenBookQA. OpenBookQA (Mihaylov et al., 2018) is a science commonsense question answering task. Its questions usually require combining elementary science facts with commonsense reasoning, emphasizing the alignment between factual knowledge and question semantics rather than simple pattern recognition.

MMLU. MMLU (Hendrycks et al., 2021) is a multiple-choice evaluation suite covering many disciplines and knowledge domains, ranging from basic knowledge to professional reasoning. Because it spans a wide set of domains with distinct knowledge structures, MMLU is well suited as a comprehensive test of pretraining-style generalization.

ARC-Challenge. ARC-Challenge (Clark et al., 2018) is the challenging subset of grade-school science exam questions and contains difficult multiple-choice science problems. Compared with easier science QA tasks, this dataset relies more heavily on compositional reasoning, background knowledge retrieval, and distractor elimination.

B.4 Biological Sequence Track

LTR Copia. LTR Copia (Khan et al., 2025) is a biological sequence classification task whose goal is to identify sequence patterns associated with Copia-like long terminal repeat retrotransposons. The task requires models to capture relations between local DNA motifs and longer-range sequence structure.

CMC-EnSpm. CMC-EnSpm (Khan et al., 2025) focuses on recognizing DNA transposon-related sequences. The task is to determine whether an input sequence has the characteristics of a specific transposon family or subfamily. Its discriminative signals are often distributed across multiple local fragments, requiring models to integrate motifs, upstream and downstream context, and long-range dependencies.

Processed Pseudogenes. Processed Pseudogenes (Khan et al., 2025) is used to identify processed pseudogene-related sequences. Such sequences can resemble functional genes while differing in structural and functional properties, so models must distinguish fine-grained sequence patterns.

H3K4me3. H3K4me3 (Dalla-Torre et al., 2025) is a histone-modification-related sequence prediction task. The goal is to determine from DNA sequence whether an input is associated with a specific chromatin mark. The task reflects relations among sequence motifs, regulatory regions, and epigenetic signals, and its class boundaries are often less clear than those in standard text or image classification.

Promoter All. Promoter All (Dalla-Torre et al., 2025) is a promoter recognition task whose goal is to determine from DNA sequence whether an input has promoter function. Promoter regions often contain multiple regulatory motifs and are directly related to gene-expression regulation, requiring models to capture both local sequence patterns and higher-level functional semantics.

Enhancers. Enhancers (Dalla-Torre et al., 2025) is an enhancer recognition task whose goal is to determine whether an input DNA fragment corresponds to a potential enhancer region. Enhancer function is often influenced by motif combinations, chromatin context, and long-range regulatory relations, making its sequence patterns harder to characterize than those in simple binary tasks. This dataset provides a regulatory-element prediction setting for the Biological Sequence Track and tests the generalization ability of hyperbolic Transformers beyond image and text sequences.

C Method Adaptation Details

Our adaptation logic is to extract the components from each method that can be abstracted as Transformer-based modeling and encapsulate them as blocks with a unified interface. Different tracks provide only input adapters and output task heads: NLP and Genomic use token embeddings, CV uses patch embeddings, and Pretraining employs a causal language-modeling framework; classification tasks are connected to a linear classification head, extractive question answering to a span head, and pretraining to an LM head.

Euclidean baseline. The Euclidean baseline (Vaswani et al., 2017) serves as our control group. It does not introduce hyperbolic space, relying solely on a standard Transformer encoder, conventional residual connections, LayerNorm, an FFN, and a linear task head.

HiT. The principal modification in the original HiT paper (He et al., 2024) lies in projecting the output representations into Poincaré space. In our adaptation, we retain a standard Euclidean Transformer encoder as the backbone, then project the encoder outputs through a Poincaré output adapter and impose auxiliary geometric constraints before attaching the shared linear task head. In other words, across NLP, CV, and Genomic, HiT inputs are still first encoded as conventional token or patch sequences, and the Transformer backbone itself remains Euclidean; the essential distinction is that the final representations are mapped into Poincaré space and subjected to hierarchical geometric constraints. For Pretraining, we likewise employ a language-modeling framework, attaching HiT's output-space geometric constraints only to the intermediate or output representations.

ConeHeads. The central innovation of the original ConeHeads paper (Tseng et al., 2023) is to replace the similarity computation in conventional self-attention with a cone compatibility score, thereby encouraging the attention mechanism to capture hierarchical containment relationships more effectively. In our adaptation, we preserve the overall Transformer architecture, while replacing the Q/K scoring mechanism in the self-attention module of each layer with ConeHeads' cone-based attention score; the value projection, residual connections, FFN, and task head are incorporated into the unified scaffold. For NLP and Genomic, it operates directly on token sequences; for CV, it operates on patch tokens; for Pretraining, it is integrated into the Transformer layers of a causal LM and combined with a causal mask.

Hyp-ViT. Hyp-ViT (Ermolov et al., 2022) was originally designed primarily for visual representation learning, with its central emphasis on mapping representations into Poincaré space and imposing a metric auxiliary loss to regularize the structure of those representations. To extend it beyond visual tasks, we abstract it into a Euclidean Transformer encoder + Poincaré metric output adapter. In CV, it takes patch tokens as input; in NLP and Genomic, it processes text or DNA tokens; in Pretraining, it is incorporated into a language-modeling framework.

HVT. HVT (Fein-Ashley et al., 2024) is primarily designed for visual representation learning. We abstract its HyperbolicLinear layers, Möbius residual connections, Poincaré LayerNorm, and attention/FFN blocks defined on the Poincaré ball into a reusable HVT Transformer encoder. In the CV Track, it receives image patch tokens; in NLP, Genomic, and Pretraining, the corresponding task-specific token embeddings are first used to construct the input sequence, which is then projected

onto the Poincaré ball before being processed through stacked HVT blocks. In other words, what we preserve is HVT's Poincaré Transformer layers rather than merely its visual input component; visual patch embeddings, textual token embeddings, and DNA token embeddings serve only as distinct input interfaces to the same HVT encoder.

GGBall. The original GGBall framework (Bu et al., 2026) is oriented more toward generative settings, whose task objectives cannot be directly applied to the four tracks in our benchmark. Our adaptation extracts its core geometric operators on the Poincaré ball and reorganizes them into a stackable GGBall Transformer layer. In this way, for NLP/CV/Genomic, GGBall no longer functions as a generative model but instead serves as a Poincaré-native encoder; for Pretraining, it is integrated into a causal LM framework as an intermediate Transformer layer.

HNN++. The original HNN++ framework (Shimizu et al., 2021) provides a collection of fundamental Poincaré neural-network operators, including the Poincaré fully connected layer, attention, multi-head attention, split/concat operations, and weighted centroid aggregation. It is not, by itself, a complete task model naturally capable of covering every benchmark Track. Therefore, we extract PoincaréLinear, PoincaréConcatLinear, Poincaré distance attention, and weighted gyromidpoint aggregation from the official implementation and assemble them into an HNN++ Transformer encoder. The input adaptation strategy is identical to that used for the other methods.

HyboNet. The core idea of HyboNet (Chen et al., 2022) is to replace conventional neural-network layers with native operators defined on the Lorentz manifold. In our adaptation, its Lorentz linear/projection, Lorentz attention score, Lorentz midpoint value aggregation, Lorentz residual connection, and Lorentz FFN are uniformly encapsulated within a Transformer encoder. For NLP, CV, Genomic, and Pretraining, the inputs first pass through their respective embedding or patch-embedding modules, are then uniformly projected into Lorentz space, and subsequently processed through stacked HyboNet blocks. For downstream tasks, we no longer use the prototype head adopted in earlier versions, but instead attach a shared linear task head or LM head; this design prevents differences in the classification head from obscuring the contribution of the Transformer architecture itself.

HyTE-FH block. HyTE-FH (Madhu et al., 2026) originates from Hyp-RAG, but the complete Hyp-RAG system depends on retrieval, knowledge graphs, RAG querying, and external document retrieval, making it unsuitable for direct integration into conventional classification or pretraining benchmarks. Our approach therefore extracts only the HyTE-FH Transformer block while removing the retrieval pipeline, knowledge-base interactions, and RAG generation process. The remaining input and output adaptation strategies are identical to those used for the other methods.

Hypformer. Hypformer (Yang et al., 2024) is among the Lorentz-family methods that most closely resemble complete hyperbolic Transformer blocks. In our adaptation, we preserve the modular organization presented in the original work and encapsulate it as a reusable Transformer block. Although NLP, CV, Genomic, and Pretraining employ different input formats, once they enter the encoder, they are uniformly treated as sequences of token/patch hidden states and projected into Lorentz space. At the output stage, the representations are uniformly mapped back to a Euclidean task head or LM head.

HELM. The original HELM framework (He et al., 2025a) is designed for large language models and incorporates LLM-oriented components such as mixture-of-curvature experts. To adapt it to our benchmark, we extract the Lorentz blocks that can replace conventional Transformer layers: Lorentz RMSNorm, HMLA Lorentz attention, latent key/value compression, Lorentz residual connections, and Lorentz FFN. We then assemble these modules into a dense Lorentz Transformer encoder. For Pretraining, it directly replaces the intermediate layers of the causal LM; for NLP, CV, and Genomic, it serves as a unified encoder that processes token or patch sequences and connects to the shared linear task head.

Table 7: Pretraining runtime and parameter-count summary. Time is reported as wall-clock training time in hours on the 4×A800 setting; the 24.00 h entries for GGBall and HNN++ indicate that the allocated time budget was reached without completion.

Method	Family	Time (h)	Total Params	Params (M)
Euclidean	Euclidean baseline	7.23	101,860,076	101.86
HiT	Euclidean-backbone	8.37	106,028,396	106.03
ConeHeads	Euclidean-backbone	13.47	101,860,076	101.86
Hyp-ViT	Euclidean-backbone	8.50	106,028,396	106.03
HVT	Poincaré manifold	23.84	106,028,486	106.03
GGBall	Poincaré manifold	24.00	106,073,336	106.07
HNN++	Poincaré manifold	24.00	110,236,658	110.24
HyboNet	Lorentz manifold	10.34	93,503,456	93.50
HyTE-FH block	Lorentz manifold	12.38	109,882,040	109.88
Hypformer	Lorentz manifold	10.80	97,397,054	97.40
HELM	Lorentz manifold	12.09	91,286,798	91.29

D Benchmark Protocol and Implementation Details

D.1 Unified Training Protocol

Within each track and dataset, all training factors are kept fixed whenever possible except for the geometric implementation of the Transformer block. Single-run experiments use random seed 42. Multi-seed results use seeds 42, 1337, and 3407, and report mean and standard deviation.

The default learning rate for supervised tasks is 3×10^{-4} for NLP/Bio and 5×10^{-4} for CV. Dropout is fixed to 0.1, and weight decay is set between 0.01 and 0.05 depending on the track. Pretraining uses AdamW with betas $(0.9, 0.95)$, weight decay 0.1, 1000 warmup updates, and cosine decay to the minimum learning rate.

D.2 Hardware and Runtime

Our NLP and Bio Track experiments are run on a single L40S GPU, while the Pretraining and CV Track experiments are run on four A800 80G GPUs. All jobs are submitted through Slurm, and training logs, evaluation metrics, model configurations, and checkpoint metadata are saved under the corresponding track-specific checkpoint directories. Pretraining and CV jobs use a 4-GPU configuration on the A800 server; NLP/Bio jobs can be completed on a single GPU because of their smaller model and input scale. Table 7 summarizes parameter counts and training times on the Pretraining Track to show engineering-cost differences among geometric implementations.

E Additional Main Results

This section supplements the compact overall table in the main text with the multi-seed mean±std results used for the cross-track summary. For cross-task readability, we keep the appendix focused on F1 and accuracy, which are the primary metrics reported consistently across tracks.

E.1 Multi-Seed Results

Tables 8-11 report the multi-seed results over seeds 42, 1337, and 3407 on a 0-100 scale. OOT denotes runs that exceeded the 24h training budget. Best results are marked in red.

E.2 Additional Efficiency Results on Promoter All

We complement the three representative tasks in Figure 4 with Promoter All from the Biological Sequence Track. These measurements use the single-L40S setting described in Appendix D.2. Figure 8 presents the training-duration and peak-memory views separately, retaining the same methods

Table 8: Experimental results on NLP tasks

Method	AG News		MNLI		Few-NERD		SemEval-2010		SQuAD 2.0		Avg.	
	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc
Euclidean	90.88±0.08	90.87±0.08	61.17±0.15	61.15±0.12	51.72±0.12	88.80±0.02	49.37±0.49	52.61±0.34	44.60±0.48	43.85±0.22	59.55±0.11	67.46±0.12
HiT	90.95±0.22	90.96±0.21	60.68±0.72	60.65±0.72	51.96±0.01	88.74±0.03	48.88±0.42	52.15±0.29	44.81±0.94	44.02±0.97	59.46±0.43	67.30±0.37
ConeHeads	90.67±0.23	90.68±0.24	60.36±1.04	60.33±1.06	52.58±0.10	88.96±0.04	**51.20±1.25**	54.67±0.51	45.07±0.91	44.33±1.10	59.98±0.42	67.79±0.31
Hyp-ViT	**91.15±0.22**	**91.11±0.18**	60.92±0.62	60.89±0.59	52.01±0.04	88.78±0.03	50.13±0.41	**54.82±0.27**	**45.49±0.70**	**44.79±0.78**	59.94±0.36	68.08±0.32
HVT	88.18±0.07	88.18±0.08	60.12±0.17	60.38±0.13	41.88±0.13	86.97±0.02	38.65±0.47	42.92±0.32	33.18±0.46	31.50±0.16	52.40±0.08	61.99±0.10
GGBall	88.95±0.19	88.91±0.23	60.30±0.80	60.73±0.82	51.93±0.08	89.14±0.05	43.24±0.80	46.87±0.30	38.38±0.70	36.73±0.74	56.56±0.28	64.48±0.20
HNN++	88.80±0.15	88.83±0.21	61.74±0.14	62.23±0.09	30.20±0.02	84.62±0.03	26.72±0.22	34.34±0.17	44.90±0.70	44.74±0.77	50.47±0.22	62.95±0.22
HyboNet	89.63±0.18	89.64±0.18	63.54±0.57	63.75±0.55	53.00±0.39	89.25±0.08	43.22±0.75	46.78±0.99	45.18±0.22	41.90±0.80	58.92±0.16	66.26±0.04
HyTE-FH block	90.86±0.26	90.86±0.26	62.84±0.44	63.27±0.47	54.42±0.12	89.49±0.03	50.49±1.22	54.01±0.77	44.54±0.98	43.10±0.95	**60.63±0.46**	**68.14±0.34**
Hypformer	90.50±0.05	90.52±0.05	**65.33±0.52**	**65.43±0.59**	**54.75±0.47**	**89.67±0.10**	46.92±1.67	50.30±2.03	**45.49±2.26**	43.54±1.44	60.60±0.34	67.89±0.30
HELM	88.28±0.08	88.32±0.07	62.72±0.17	62.99±0.14	51.52±0.12	88.99±0.02	47.60±0.47	51.16±0.33	42.47±0.46	41.53±0.16	58.52±0.08	66.60±0.11

Table 9: Experimental results on vision tasks

Method	CIFAR-10		CIFAR-100		Food-101		EuroSAT		Avg.	
	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc
Euclidean	77.49±0.09	77.61±0.07	49.76±0.15	49.39±0.14	51.92±0.13	51.34±0.11	92.57±0.11	92.88±0.07	67.93±0.11	67.80±0.09
HiT	75.01±0.12	75.12±0.15	48.94±0.21	49.02±0.23	45.96±0.28	46.57±0.32	91.45±0.25	91.89±0.13	65.34±0.20	65.65±0.21
ConeHeads	72.37±0.17	72.49±0.20	49.27±0.08	49.54±0.11	47.03±0.21	47.16±0.19	91.23±0.21	91.65±0.17	64.97±0.06	65.21±0.05
Hyp-ViT	75.23±0.17	75.30±0.12	49.13±0.24	49.42±0.26	44.95±0.22	45.10±0.27	93.59±0.16	94.04±0.19	65.73±0.19	65.97±0.21
HVT	58.90±0.11	59.42±0.12	33.72±0.22	34.92±0.24	28.75±0.17	30.42±0.19	86.70±0.06	87.31±0.19	52.02±0.13	53.02±0.17
GGBall	70.93±0.08	71.00±0.17	43.73±0.08	43.63±0.12	37.27±0.15	37.13±0.13	89.48±0.17	89.78±0.14	60.35±0.03	60.39±0.05
HNN++	52.73±0.07	53.37±0.05	33.53±0.24	34.41±0.24	24.73±0.17	25.55±0.17	86.68±0.18	87.42±0.17	49.42±0.16	50.19±0.15
HyboNet	**78.77±0.09**	**78.85±0.12**	49.44±0.20	49.22±0.19	50.51±0.09	50.53±0.08	**95.53±0.13**	**95.73±0.20**	68.56±0.12	68.58±0.14
HyTE-FH block	71.76±0.29	71.64±0.25	43.13±0.34	43.52±0.38	44.25±0.33	44.63±0.29	51.83±0.19	55.35±0.11	52.74±0.28	53.79±0.26
Hypformer	77.51±0.04	77.60±0.02	**50.03±0.16**	**50.01±0.07**	**54.43±0.13**	**54.75±0.14**	94.52±0.07	94.81±0.13	**69.12±0.07**	**69.29±0.09**
HELM	66.40±0.09	66.50±0.11	41.97±0.27	42.16±0.25	36.73±0.15	36.72±0.15	92.36±0.13	92.63±0.03	59.37±0.16	59.50±0.12

Table 10: Experimental results on pretraining reasoning tasks

Method	CommonsenseQA	HellaSwag	OpenBookQA	MMLU	ARC-Challenge	Avg.
Euclidean	19.69±0.12	25.43±0.15	24.96±0.17	23.42±0.14	22.90±0.12	23.28±0.14
HiT	19.78±0.14	25.39±0.18	25.35±0.19	23.06±0.12	22.92±0.13	23.30±0.15
ConeHeads	19.62±0.06	25.91±0.06	25.29±0.10	23.05±0.09	23.68±0.04	23.51±0.06
Hyp-ViT	20.08±0.11	25.93±0.11	25.71±0.11	23.77±0.13	23.06±0.10	23.71±0.11
HVT	19.62±0.06	25.15±0.06	25.29±0.10	23.05±0.09	23.68±0.04	23.36±0.06
GGBall	OOT	OOT	OOT	OOT	OOT	OOT
HNN++	OOT	OOT	OOT	OOT	OOT	OOT
HyboNet	20.36±0.14	25.82±0.16	25.97±0.18	22.80±0.12	23.17±0.11	23.62±0.14
HyTE-FH block	21.15±0.20	**26.68±0.19**	**26.65±0.25**	**24.53±0.22**	**26.12±0.20**	**25.03±0.21**
Hypformer	**21.25±0.12**	26.28±0.09	23.31±0.11	24.48±0.12	25.93±0.08	24.25±0.10
HELM	19.67±0.11	25.55±0.11	25.51±0.11	23.65±0.13	22.81±0.10	23.44±0.11

Table 11: Experimental results on biological sequence tasks

Method	LTR Copia		CMC-EnSpm		Processed Pseudogenes		H3K4me3		Promoter All		Enhancers		Avg.	
	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc
Euclidean	70.11±0.06	70.87±0.07	65.26±0.11	69.98±0.11	65.37±0.06	64.64±0.03	69.01±0.37	56.23±0.22	80.05±0.46	82.04±0.24	75.13±0.04	72.06±0.05	70.82±0.12	69.30±0.11
HiT	69.89±0.22	72.10±0.13	64.46±0.51	69.10±0.46	60.98±0.12	63.72±0.09	67.98±0.40	55.91±0.21	82.39±1.02	83.29±0.99	73.41±0.18	**74.63±0.11**	69.85±0.40	69.79±0.30
ConeHeads	70.28±0.17	70.10±0.08	69.81±0.68	70.30±0.66	66.56±0.17	64.13±0.08	61.60±0.75	54.68±0.28	83.75±1.11	84.33±1.13	75.37±0.19	74.16±0.08	71.23±0.38	69.62±0.27
Hyp-ViT	**71.77±0.14**	**72.20±0.10**	63.24±0.14	65.10±0.17	59.90±0.09	62.68±0.08	0.00±0.00	46.95±0.19	82.74±0.46	83.72±0.26	73.82±0.10	74.06±0.07	58.58±0.09	67.45±0.14
HVT	64.55±0.06	67.36±0.07	61.49±0.11	65.33±0.11	63.47±0.08	62.72±0.04	61.73±0.40	56.87±0.24	70.14±0.46	76.45±0.18	75.15±0.05	73.78±0.06	66.09±0.06	67.08±0.09
GGBall	70.70±0.10	69.76±0.08	62.66±0.11	68.95±0.07	63.93±0.22	61.95±0.11	68.83±0.23	56.55±0.14	80.65±0.61	82.38±0.29	75.28±0.17	72.64±0.15	70.34±0.18	68.71±0.11
HNN++	71.06±0.12	70.44±0.08	67.68±0.16	70.26±0.12	65.89±0.02	61.72±0.11	62.67±0.39	56.95±0.25	79.01±0.44	80.70±0.22	72.62±0.06	69.84±0.13	69.82±0.13	68.32±0.14
HyboNet	68.00±0.12	69.31±0.14	59.11±0.31	67.32±0.35	**66.93±0.21**	60.04±0.04	62.92±0.54	54.35±0.53	81.61±0.21	82.84±0.84	75.19±0.08	74.35±0.10	68.96±0.11	68.03±0.11
HyTE-FH block	68.42±0.22	69.48±0.18	71.01±0.50	72.78±0.51	61.71±0.17	65.80±0.25	69.00±1.07	56.69±0.62	88.43±1.26	88.63±1.25	**75.49±0.22**	74.17±0.23	72.34±0.55	71.26±0.48
Hypformer	68.61±0.09	71.92±0.02	**71.58±0.40**	**73.13±0.33**	65.96±0.29	**66.52±0.04**	**69.48±1.19**	**57.43±1.18**	**88.67±2.05**	**88.96±1.46**	74.42±0.19	70.23±0.03	**73.12±0.33**	**71.37±0.20**
HELM	70.47±0.06	70.38±0.07	63.36±0.11	65.88±0.11	65.82±0.07	61.27±0.03	**69.48±0.38**	55.46±0.20	84.37±0.43	85.45±0.15	72.83±0.04	70.02±0.05	71.05±0.06	68.08±0.08

and reported task scores. The panels provide a task-specific view of computational cost alongside the predictive results in Table 11.

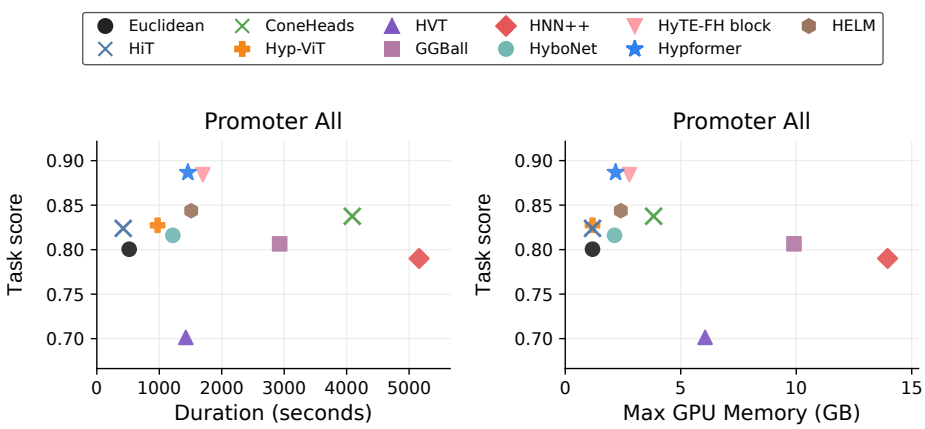

Figure 8: Efficiency-performance trade-offs on Promoter All. Left: reported task score versus training duration in seconds. Right: the same score versus peak GPU memory in GB. Colors and markers denote the same methods as in Figure 4.



Figure 9: CIFAR-10 cross-method prediction examples.

F Qualitative Analysis and Visualizations

F.1 CV Classification Samples

Figures 9-12 provide sample-level predictions for representative CV examples under the same input images. Unlike the aggregate scores in the main tables, these cases are intended to expose how differ-



Figure 10: CIFAR-100 cross-method prediction examples.

ent geometric implementations make concrete decisions. The CIFAR-10 and CIFAR-100 examples show that the Lorentz-family methods are more consistent on these selected samples, whereas several Euclidean-backbone and Poincaré-family variants choose high-confidence but incorrect categories. The Food-101 and EuroSAT cases further show that the disagreement is not tied to a single dataset: on the guacamole and sea/lake samples, the Poincaré-family methods tend to move toward nearby but wrong visual categories, while the Euclidean-backbone and most Lorentz-family methods remain correct.

F.2 Pretraining Reasoning Samples

Figures 7 and 13 compare cross-method predictions on multiple-choice reasoning examples from the Pretraining Track. These cases complement the mean results in Table 10: methods with similar average accuracy can still fail on different distractors or recover different subsets of questions. Some examples show models selecting semantically plausible but constraint-violating answers, while others reveal that a specific module family can recover the correct option even when nearby variants fail. The OOT entries for GGBall and HNN++ also make clear that runtime feasibility is part of the comparison. Thus, these qualitative examples are best interpreted as local decision traces across commonsense reasoning, sentence completion, open-book QA, knowledge questions, and scientific reasoning.

F.3 Biological Sequence Case Studies

Figures 14-17 visualize four representative binary classification cases from the Biological Sequence Track. Each figure shows the input DNA sequence together with the prediction and confidence of each method. In contrast to natural-image or text examples, the evidence for these genomic labels is distributed across local motifs, base composition, and longer-range context; consequently, methods may agree on the final label while assigning quite different confidence values. Across the LTR Copia,



Figure 11: Food-101 cross-method prediction examples.

Promoter All, Enhancers, and Processed Pseudogenes examples, Lorentz-family methods often recover the correct label, but the pattern is not uniform across all Lorentz variants. Poincaré-family and Euclidean-backbone methods also fluctuate across tasks, suggesting that biological-sequence performance depends on how each model aggregates local and long-range sequence signals rather than only on the final classifier.

F.4 Training Loss Curves

Figures 18-20 show the cleaned seed-42 training loss curves for the biological sequence, pretraining, and NLP tracks; the CV training curves are shown in Figure 6. These curves mainly serve as optimization diagnostics rather than final evaluation metrics. They help distinguish two sources of performance differences: one in which a method is difficult to optimize, and another in which optimization is stable but the learned representation does not fully match the downstream evaluation metric.

For the biological sequence track, Figure 18 shows that many methods have similar loss trajectories, especially on tasks where the loss scale changes only mildly during training. This indicates that final differences in accuracy and F1 cannot be fully explained by training-loss decrease alone; threshold effects, class imbalance, and task-specific decision boundaries may also play important roles. For the pretraining track, Figure 19 includes only 100M-scale models with complete training logs and comparable parameter counts, so OOT or incomplete runs are not directly visualized. The NLP curves in Figure 20 show clearer task-level differences: some methods reduce loss quickly on classification tasks, but the distinction becomes less obvious on more complex structured prediction or question answering tasks. Overall, training curves are better interpreted as diagnostic evidence; the main result tables remain the core basis for performance conclusions, while optimization trajectories help judge whether weaker downstream results arise from convergence behavior or from the representation design itself.

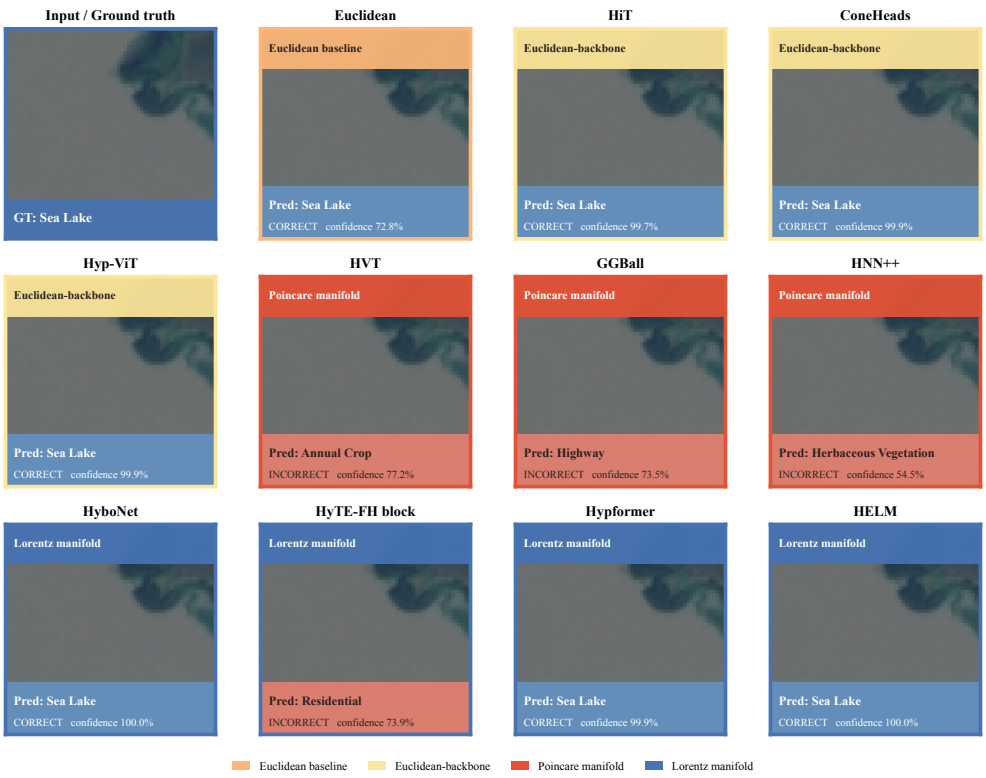

Figure 12: EuroSAT cross-method prediction examples.

Question / Dataset	Choices	Euclidean baseline	Euclidean-backbone			Poincare manifold			Lorentz manifold			
		Euclidean	HiT	ConeHeads	Hyp-ViT	HVT	GGBall	HNN++	HyboNet	HyTE-FH block	Hypformer	HELM
Q1a **CommonsenseQA** Who is a police officer likely to work for?	A. beat B. direct traffic **C. city** D. street E. president	A ✗	A ✗	A ✗	B ✗	A ✗	OOT	OOT	A ✗	B ✗	C ✓	A ✗
Q1b **CommonsenseQA** Where is a doormat likely to be in front of?	A. facade **B. front door** C. doorway D. entrance porch E. hallway	A ✗	A ✗	A ✗	A ✗	A ✗	OOT	OOT	A ✗	B ✓	C ✗	A ✗
Q2a **HellaSwag** Context: A boy is running down a track. the boy What happens next?	A. runs into a car. B. gets in a mat. **C. lifts his body above the height of a pole.** D. stands on his hands and springs.	A ✗	A ✗	A ✗	A ✗	C ✓	OOT	OOT	B ✗	A ✗	C ✓	A ✗
Q2b **HellaSwag** Context: Children bring desert out for their family member. the family What happens next?	A. floats in a river. B. member stands looking into a hut and then handing people photographs. C. member cuts a piece of sunscreen. **D. enjoys eating the desert together.**	A ✗	A ✗	A ✗	A ✗	A ✗	OOT	OOT	D ✓	A ✗	A ✗	A ✗
Q3a **OpenBookQA** What boils at the boiling point?	**A. Kool-Aid** B. Cotton C. Paper towel D. Hair	A ✓	A ✓	A ✓	B ✗	A ✓	OOT	OOT	A ✓	B ✗	C ✗	A ✓
Q3b **OpenBookQA** Pumpkin pie spice comes from	A. animals **B. plants** C. rocks D. air	A ✗	A ✗	A ✗	B ✓	A ✗	OOT	OOT	A ✗	B ✓	C ✗	A ✗
Q4a **MMLU** *Abstract Algebra* Find the characteristic of the ring Z_3 x 3Z.	**A. 0** B. 3 C. 12 D. 30	A ✓	A ✓	A ✓	A ✓	A ✓	OOT	OOT	B ✗	A ✓	C ✗	A ✓
Q4b **MMLU** *Abstract Algebra* Find the order of the factor group (Z_11 x Z_15)/(<1, 1>)	**A. 1** B. 2 C. 5 D. 11	A ✓	A ✓	A ✓	A ✓	A ✓	OOT	OOT	B ✗	A ✓	C ✗	A ✓
Q5a **ARC-Challenge** At which temperature does water freeze?	**A. 0 degrees Celsius** B. 32 degrees Celsius C. 100 degrees Celsius D. 212 degrees Celsius	A ✓	A ✓	A ✓	A ✓	A ✓	OOT	OOT	B ✗	B ✗	C ✗	A ✓
Q5b **ARC-Challenge** The best way to separate salt from water is with the use of	A. oil. **B. heat.** C. a magnet. D. rubbing alcohol.	A ✗	A ✗	A ✗	A ✗	A ✗	OOT	OOT	B ✓	B ✓	C ✗	A ✗

Figure 13: Additional cross-method prediction examples from the Pretraining Track.

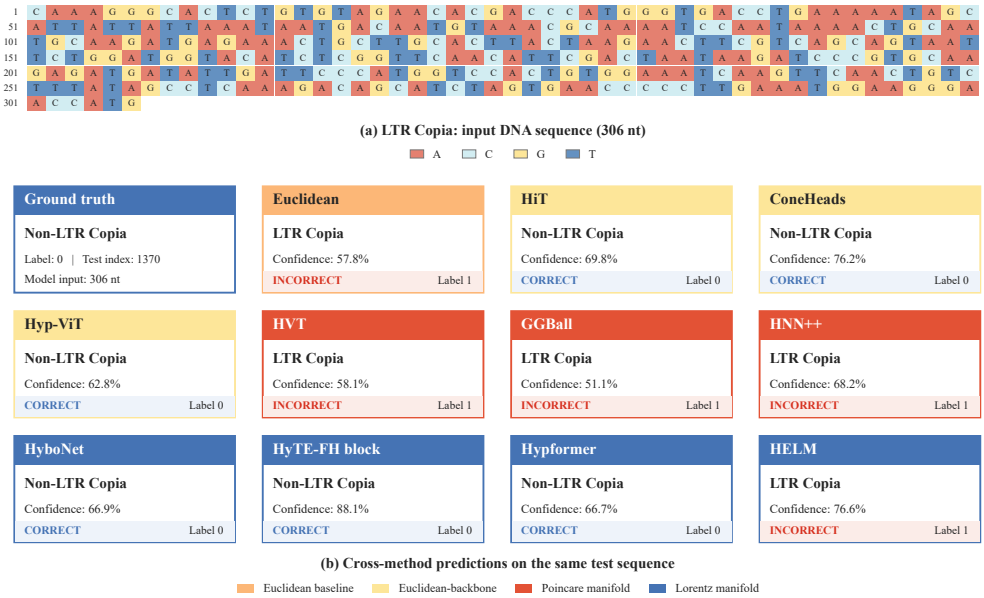

Figure 14: LTR Copia

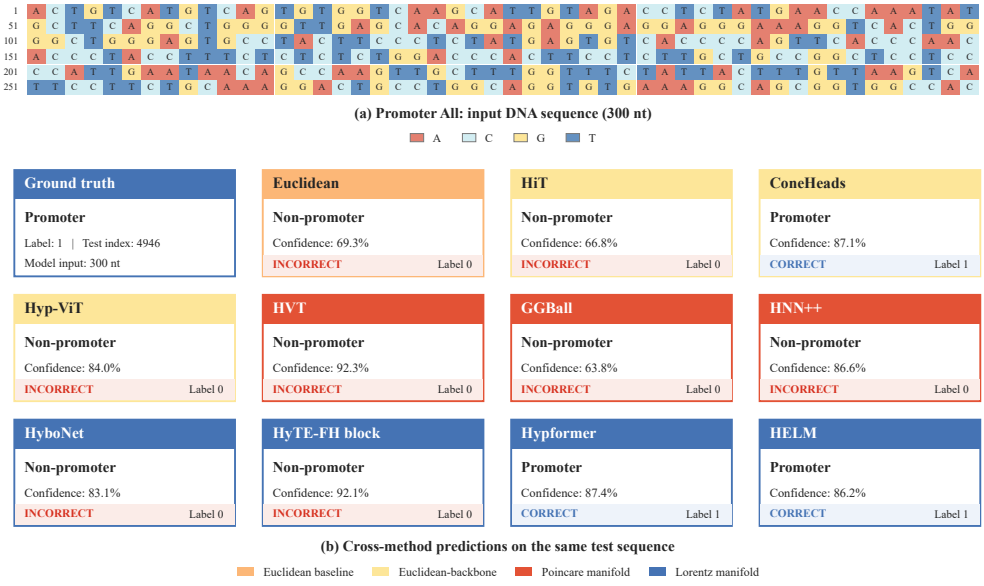

Figure 15: Promoter All

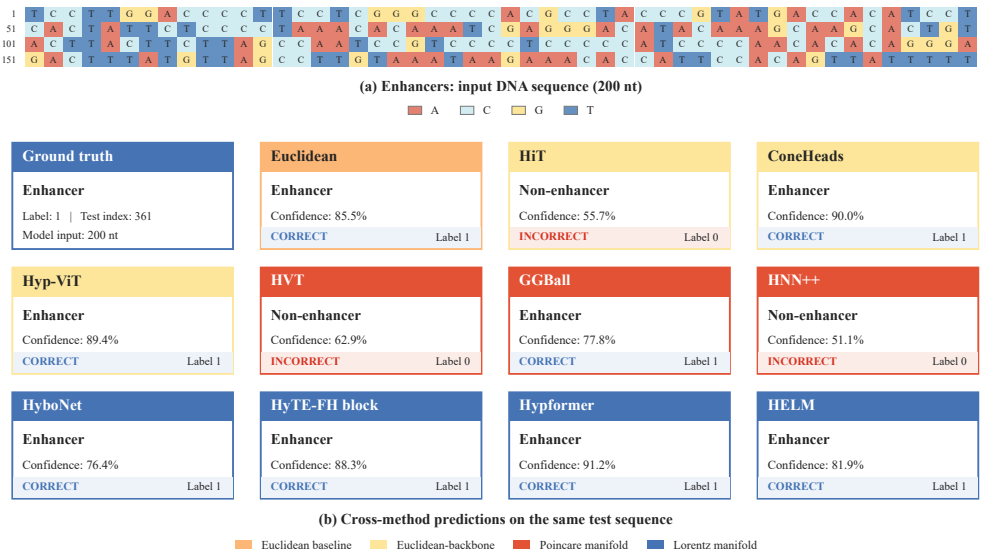

Figure 16: Enhancers

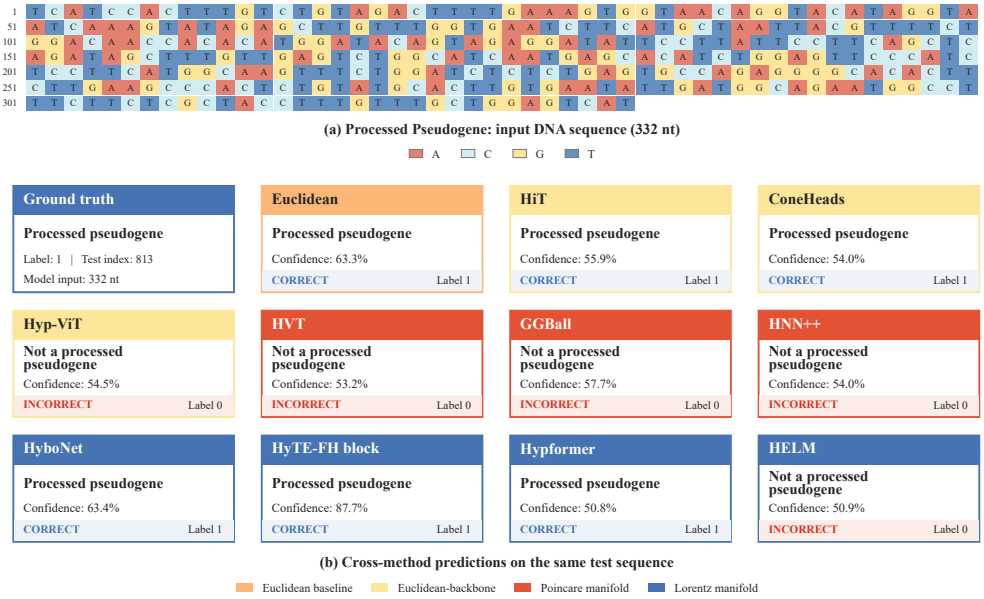

Figure 17: Processed Pseudogenes

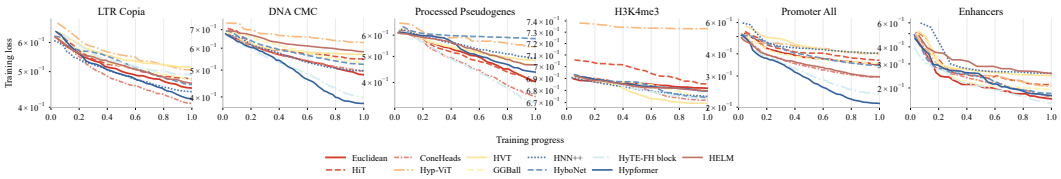

Figure 18: Training loss curves on biological sequence tasks from the cleaned seed-42 runs. Each subplot reports smoothed training loss over normalized training progress.

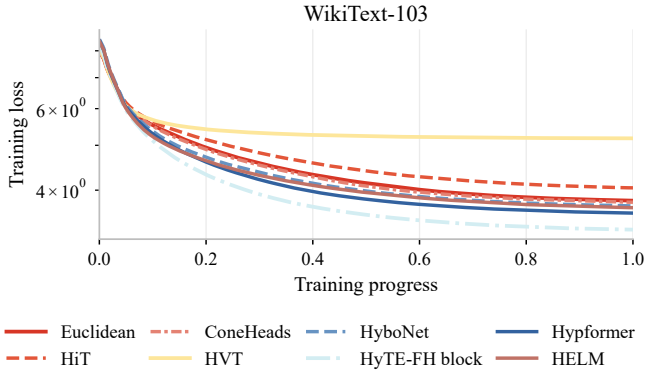

Figure 19: Training loss curve on the pretraining track from comparable 100M-parameter runs with complete logs.

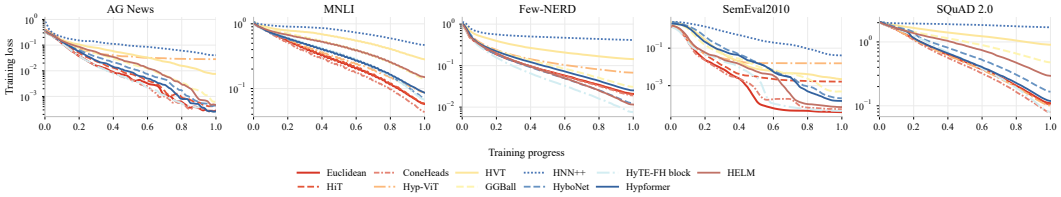

Figure 20: Training loss curves on NLP tasks from the cleaned seed-42 runs. Each curve reports the smoothed training loss over normalized training progress.