

Appendix

A	Proofs and Details of Theoretical Results	20
A.1	Ollivier-Ricci Curvature	20
A.2	HOPE is a Lorentz Rotation	21
A.3	Proposition 4.1: HOPE is a function of embeddings and relative position only	21
A.4	Proposition 4.2: HOPE decays with increasing relative position	22
A.5	Proposition 4.3: HOPE enables tokens to attend across distances	23
A.6	Proposition 4.4: Attention heads with HOPE learn special positional patterns	23
A.7	Proposition 4.5: Invariance guarantees of Hyperbolic RMSNorm	25
B	Additional Details	26
B.1	MICE as a Lorentzian Module	26
B.2	Hyperbolic Multi-Head Latent Attention	26
B.3	Hyperbolic SwiGLU Feedforward Network	27
C	Training and Evaluation Details	27
C.1	Models Setup	28
C.2	Training Details	28
C.3	Evaluation Details	29
D	Ablation Studies	29
D.1	Ablation for HMLA	30
D.2	Ablation for HOPE	30

A Proofs and Details of Theoretical Results

A.1 Ollivier-Ricci Curvature

Ricci curvature is a geometric object that measures average *geodesic dispersion* on a Riemannian manifold, i.e., whether straight paths on a surface remain parallel (zero curvature), converge (positive curvature), or diverge (negative curvature). Ollivier-Ricci curvature is the discrete analog for graphs that do not have a notion of tangent structure.

Suppose we have a graph $\mathcal{G}(V, E)$. For a node $i \in V$, we define probability measures μ_i by:

$$\mu_i(j) = \begin{cases} \frac{1}{\deg(i)} & (i, j) \in E \\ 0 & \text{otherwise.} \end{cases}$$

For $i, j \in V$, the Ollivier-Ricci curvature is given by,

$$\kappa_{\mathcal{G}}(i, j) = 1 - \frac{W_1^{\mathcal{G}}(\mu_i, \mu_j)}{d(i, j)}.$$

Here, $W_1^{\mathcal{G}}$ is the 1-Wasserstein distance between measures, μ_i and μ_j . Intuitively, the curvature is defined as random walks between i and j . If the random walks tend to stay at equal distances, $\kappa(i, j) = 0$; if they diverge, $\kappa(i, j) < 0$; and if they converge, $\kappa(i, j) > 0$. Since curvature is a local property of a Riemannian surface, for graphs, we examine local neighborhoods with a step size of 1. We thus choose $\mu_i(j) = \frac{1}{\deg(j)}$ if $(i, j) \in E$, otherwise 0.

In our preliminary analysis of popular decoder-only LLMs (Figure 1), we draw k -nearest-neighbors graphs with the final-layer token embeddings for a collection of prompts from RedPajama [45] to ascertain their geometric structure. We observe a high variability of negative curvature values across tokens, hinting at their non-Euclidean nature. Robinson et al. [37] further allude to the non-Euclidean token subspace, necessitating language models that can accommodate this unique geometry.

A.2 HOPE is a Lorentz Rotation

Here, we expand on HOPE being a Lorentz rotation. Since $\mathbf{R}_{i,\Theta}$ is a Euclidean rotation, we have $\|\mathbf{z}\| = \|\mathbf{R}_{i,\Theta}\mathbf{z}\|$. Then, since for any Lorentz vector $\mathbf{z} \in \mathbb{L}^{K,n}$ we have $z_t = \sqrt{-1/K + \|\mathbf{z}_s\|} \in \mathbb{R}$. Computing $(\mathbf{R}_{i,\Theta}\mathbf{z})_t = \sqrt{-1/K + \|\mathbf{R}_{i,\Theta}\mathbf{z}_s\|} = \sqrt{-1/K + \|\mathbf{z}_s\|} = z_t$. Thus, HOPE does not affect the time-like dimension of \mathbf{z} , so we have,

$$\text{HoPE}(\mathbf{z}) = \begin{pmatrix} 1 & 0 \\ 0 & \mathbf{R}_{i,\Theta} \end{pmatrix} \mathbf{z},$$

making HOPE a valid Lorentz rotational operation.

A.3 Proposition 4.1: HOPE is a function of embeddings and relative position only

Proposition. Let \mathbf{X} be T tokens with $\mathbf{x}_i \in \mathbb{L}^{K,d}$. Let \mathbf{Q}, \mathbf{K} be queries and keys as in Equation (3). Then $-d_{\mathcal{L}}^2(\text{HoPE}(\mathbf{q}_a), \text{HoPE}(\mathbf{k}_b)) = g(\mathbf{x}_a, \mathbf{x}_b; a - b)$ for some function g .

Proof. We will denote $\text{HoPE}(\mathbf{q}_a), \text{HoPE}(\mathbf{k}_b)$ as $f_q(\mathbf{x}_a), f_k(\mathbf{x}_b)$ where f_q, f_k denotes the function that projects the word embeddings to queries and keys, and then applying HOPE. In practice, the projection is done through a hyperbolic linear layer, which we take to be HLT from Section 2. It suffices to prove this proposition for the case of $d = 2$, since HOPE does not affect the time-like dimension of the inputs and $\mathbf{R}_{i,\Theta}$ acts independently on each 2D block. First, note that we have

$$\begin{aligned} -d_{\mathcal{L}}^2(f_q(\mathbf{x}_a), f_k(\mathbf{x}_b)) &= \frac{2}{K} - 2\langle f_q(\mathbf{x}_a), f_k(\mathbf{x}_b) \rangle_{\mathcal{L}} \\ &= \frac{2}{K} + 2(f_q(\mathbf{x}_a)_t f_k(\mathbf{x}_b)_t) - 2\langle f_q(\mathbf{x}_a)_s, f_k(\mathbf{x}_b)_s \rangle, \end{aligned}$$

where $\langle \cdot, \cdot \rangle_{\mathcal{L}}$ denotes Lorentzian inner product and $\langle \cdot, \cdot \rangle$ denotes the regular Euclidean inner product. Since HOPE is a Lorentz rotation, the term $2(f_q(\mathbf{x}_a)_t f_k(\mathbf{x}_b)_t)$ is simply $2((\mathbf{q}_a)_t, (\mathbf{k}_b)_t) = 2(\sqrt{\mathbf{W}^{\mathbf{Q}}\mathbf{x}_a} - 1/K, \sqrt{\mathbf{W}^{\mathbf{K}}\mathbf{x}_b} - 1/K)$, so we focus on the inner product $\langle f_q(\mathbf{x}_a)_s, f_k(\mathbf{x}_b)_s \rangle$. Then, by assuming that $d = 2$, we have $(f_q(\mathbf{x}_a))_s, (f_k(\mathbf{x}_b))_s \in \mathbb{R}^2$; hence, we can parametrize these vectors by their radial and angular components. For simplicity, denote $(f_q(\mathbf{x}_a))_s, (f_k(\mathbf{x}_b))_s$ as \mathbf{a}, \mathbf{b} respectively. Then, write $\langle \mathbf{a}, \mathbf{b} \rangle$ as a function g' . Afterwards, we parametrize the vectors as

$$\begin{aligned} \mathbf{a} &= \varphi_q(\mathbf{x}_a, a) e^{i\vartheta_q(\mathbf{x}_a, a)} \\ \mathbf{b} &= \varphi_k(\mathbf{x}_b, b) e^{i\vartheta_k(\mathbf{x}_b, b)} \\ g' &= \varphi_g e^{i\vartheta_{g'}}, \end{aligned} \tag{8}$$

where $\varphi_{\{q,k,g'\}}$ denote the radial component and $\vartheta_{\{q,k,g'\}}$ denote the angular component. Note that it suffice to show that under HOPE, we can express $\varphi_{g'}, \vartheta_{g'}$ as a function of the word embeddings and relative position. To see this, note that by definition of g' we have

$$\begin{aligned} \varphi_q(\mathbf{x}_a, a) \varphi_k(\mathbf{x}_b, b) &= \varphi_{g'} \\ \vartheta_q(\mathbf{x}_a, a) - \vartheta_k(\mathbf{x}_b, b) &= \vartheta_{g'}. \end{aligned} \tag{9}$$

Now, given the fact that HOPE acts via Euclidean rotation on the time-like dimension of any vector, we have $\varphi_q(\mathbf{x}_a, a) = \varphi_q(\mathbf{x}_a, a'), \varphi_k(\mathbf{x}_b, b) = \varphi_k(\mathbf{x}_b, b')$ for any a', b' . In particular, when $a' = b' = 0$, HOPE acts via identity since all rotation angles become 0. Hence, we have

$$\begin{aligned} \varphi_q(\mathbf{x}_a, a) &= \|(\mathbf{q}_a)_s\| \\ \varphi_k(\mathbf{x}_b, b) &= \|(\mathbf{k}_b)_s\|. \end{aligned} \tag{10}$$

Furthermore, we have $\varphi_{g'} = \|(\mathbf{q}_a)_s\| \|(\mathbf{k}_b)_s\| = \|\mathbf{W}^{\mathbf{Q}}\mathbf{x}_a\| \|\mathbf{W}^{\mathbf{K}}\mathbf{x}_b\|$ by plugging back into Equation (9), which is a function of just the word embeddings. Next, for the angular component, note

that given the definition of HoPE, the rotation on any 2D block of the space-like dimension of the input at position p is simply a scaling of a fixed rotation angle by p . Letting this fixed angle be σ , the rotation is precisely $p\sigma$. Therefore, we have

$$\begin{aligned}\vartheta_q(\mathbf{x}_a, a) &= \theta_q + a\sigma \\ \vartheta_k(\mathbf{x}_b, b) &= \theta_k + b\sigma,\end{aligned}\tag{11}$$

where θ_q, θ_k denote the angular components of $\mathbf{q}_a, \mathbf{k}_b$. Next, given Equation (9), we have $\vartheta_{g'} = (a - b)(\sigma) + (\theta_q - \theta_k)$. Note that we have $e^{i\theta_q} = \frac{\mathbf{W}^Q \mathbf{x}_a}{\|\mathbf{W}^Q \mathbf{x}_a\|}$ and $e^{i\theta_k} = \frac{\mathbf{W}^K \mathbf{x}_b}{\|\mathbf{W}^K \mathbf{x}_b\|}$. Consequently, $\vartheta_{g'}$ is a function of the word embeddings and the relative position $a - b$. All in all, $-d_{\mathcal{L}}^2(f_q(\mathbf{x}_a), f_k(\mathbf{x}_b))$ can be expressed with a function $g(\mathbf{x}_a, \mathbf{x}_b; a - b)$ as desired. \square

A.4 Proposition 4.2: HoPE decays with increasing relative position

Proposition. *Let \mathbf{Q}, \mathbf{K} be as defined in Equation (3), then the negative square Lorentz distance $-d_{\mathcal{L}}(\text{HoPE}(\mathbf{q}_a), \text{HoPE}(\mathbf{k}_b))$ can be upper bounded by $f(\mathbf{q}_a, \mathbf{k}_b)g(a - b) < 0$, where f has no dependencies on position, and g depends entirely on relative position and scales inversely w.r.t. $a - b$.*

Proof. For simplicity, we denote $\mathbf{q}_a = \mathbf{q}, \mathbf{k}_b = \mathbf{k}$. Recall that

$$-d_{\mathcal{L}}^2(\text{HoPE}(\mathbf{q}), \text{HoPE}(\mathbf{k})) = -\frac{2}{K} - 2(q_t k_t) + 2\mathbf{q}_s^\top \mathbf{k}_s.$$

Next, for simplicity, we denote $\mathbf{q}_s, \mathbf{k}_s$ as \mathbf{a}, \mathbf{b} respectively. We group together entries of the queries and keys, where $\mathbf{a}_{[2k:2k+1]}, \mathbf{b}_{[2k:2k+1]}$ as the $2k$ -th and the $(2k + 1)$ -th entries of \mathbf{a}, \mathbf{b} respectively. With this in mind, note that since we take query and key projects to be with HLT as given in Section 2, we obtain $\mathbf{a} = \mathbf{W}^Q \mathbf{x}_a$ and $\mathbf{b} = \mathbf{W}^K \mathbf{x}_b$. To that end, we can assert that

$$\begin{aligned}\mathbf{a}^\top \mathbf{b} &= (\mathbf{R}_{a,\Theta} \mathbf{W}^Q \mathbf{x}_a)^\top (\mathbf{R}_{b,\Theta} \mathbf{W}^K \mathbf{x}_b) \\ &= \mathbf{x}_a^\top \mathbf{W}^Q \mathbf{R}_{b-a,\Theta} \mathbf{W}^K \mathbf{x}_b \\ &= \text{Re} \left(\sum_{k=0}^{n/2} \mathbf{a}_{[2k,2k+1]} \mathbf{b}_{[2k,2k+1]}^* e^{i(b-a)\theta_k} \right),\end{aligned}\tag{*}$$

where $\text{Re}(\mathbf{x})$ denotes the real component of $\mathbf{x} \in \mathbb{C}$. Now, recall Abel's Transformation, which allows one to rewrite the sum of the product of two sequences as the product of the partial sums. Denote one of the $\mathbf{a}_{[2k,2k+1]} \mathbf{b}_{[2k,2k+1]}^*$ as \mathbf{A}_k , and denote the sequence $\sum_{l=0}^k e^{i(b-a)\theta_l}$ as \mathbf{E}_l . With this in mind,

(*) can be written as $\text{Re} \left(\sum_{k=0}^{n/2} \mathbf{A}_k (\mathbf{E}_{k+1} - \mathbf{E}_k) \right)$. Consequently, we obtain (recall that boundary term $\mathbf{A}_{n/2} = 0$)

$$\begin{aligned}|\mathbf{a}^\top \mathbf{b}| &= \left| \sum_{k=0}^{n/2} \mathbf{A}_k (\mathbf{E}_{k+1} - \mathbf{E}_k) \right| \\ &= \left| \sum_{k=0}^{n/2} (\mathbf{A}_{k+1} - \mathbf{A}_k) \mathbf{E}_k \right| \quad (\text{Abel Transformation}) \\ &\leq \sum_{k=0}^{n/2} |(\mathbf{A}_{k+1} - \mathbf{A}_k)| |\mathbf{E}_k|\end{aligned}$$

Now, note that the term $\mathbf{A}_{k+1} - \mathbf{A}_k$ has no dependency on position. The sum $\sum_{k=0}^{n/2} |\mathbf{E}_k|$ scales inversely with $b - a$, as shown by Su et al. [39]. To that end, we have

$$\begin{aligned} -d_{\mathcal{L}}^2(\text{HoPE}(\mathbf{q}), \text{HoPE}(\mathbf{k})) &= -\frac{2}{K} - 2(q_t k_t) + 2\mathbf{q}_s^\top \mathbf{k}_s \\ &\leq -\frac{2}{K} - 2(q_t k_t) + 2 \max_k |(\mathbf{A}_{k+1} - \mathbf{A}_k)| \sum_{k=0}^{n/2} |\mathbf{E}_k|. \end{aligned}$$

Note that \mathbf{A}_k, q_t, k_t depends on only the word embeddings and \mathbf{E}_k depends only on the position. Thus, we have the desired result. \square

A.5 Proposition 4.3: HOPE enables tokens to attend across distances

Proposition. *Let \mathbf{Q}, \mathbf{K} be as defined in Equation (3), then HOPE can be maximal at an arbitrary distance, i.e., for any relative distance $r \in \mathbb{Z}$, there exists a key \mathbf{k}_j such that the softmax value is maximum at distance r .*

Proof. We first restate Lemma A.1. from Barbero et al. [3]. The remainder of our proof follows a similar layout to that of Proposition 3.1. in Barbero et al. [3].

Lemma A.1. (Barbero et al. [3]) *Consider $g \in \mathbb{Q}$ with $g \neq 0$ and $n \in \mathbb{Z}$. Then, $ng \equiv 0 \pmod{2\pi}$ only when $n = 0$. In particular, this also holds if g is algebraic.*

Consider a distance r , a non-trivial query $\mathbf{Q}_p = \psi \in \mathbb{L}^{K,n}$, as well as a key $\mathbf{K} = \text{HoPE}(\mathbf{Q}_p)$. We can represent ψ as a combination of a time and space dimension on the Lorentzian manifold, $[\psi_t, \psi_s] \in \mathbb{L}^{K,n}$. Using our definition of HOPE in Section 4.1, we have

$$\mathbf{K}_q = \left[\sqrt{\|\mathbf{R}_{p,\Theta} \psi_s\|^2 - \frac{1}{K}}, \mathbf{R}_{p,\Theta} \psi_s \right] \in \mathbb{L}^{K,n}.$$

Recall that the operator $\mathbf{R}_{p,\Theta}$ is a valid Euclidean rotation in \mathbb{R}^n while HOPE remains a valid Lorentzian operation. Therefore, $\mathbf{R}_{p,\Theta}$ does not affect the Euclidean norm in the time dimension as it is an isometry. Instead, we can focus on the space dimension $\psi_s \in \mathbb{R}^n$, on which we can use the Euclidean dot product. Assume the query is at position i and the key is at some $j \leq i$. We then compute the following dot product:

$$\begin{aligned} \psi_{s,p}^\top \text{HoPE}(\psi_{s,p}) &= \psi_s^\top \mathbf{R}_{p,\Theta}^{(j-i)+r} \psi_s \\ &= \sum_{l=1, \dots, n/2} \left(\psi_s^{(l)} \right)^\top \mathbf{R}_{p,\theta_l}^{(j-i)+r} \left(\psi_s^{(l)} \right) \\ &= \sum_{l=1, \dots, n/2} \left\| \psi_s^{(l)} \right\|^2 \cos((j-i+r)\theta_l). \end{aligned} \tag{12}$$

Using Lemma A.1. from Barbero et al. [3], we observe the maximum can be achieved when $j-i = -r$ for $j-i \leq 0$ since we are using causal masking, $j \leq i$. This ensures $\cos((j-i+r)\theta_l) = \cos(0) = 1$, concluding the proof. \square

A.6 Proposition 4.4: Attention heads with HOPE learn special positional patterns

Proposition. *Attention heads with HOPE can learn diagonal or off-diagonal attention patterns.*

Proof. The proof follows a similar layout as that of Proposition 5.3. from Barbero et al. [3]. We start with the diagonal case. Suppose $\mathbf{Q}_i = \mathbf{K}_j = \psi \in \mathbb{L}^{K,d}$, for non-trivial $\psi = [\psi_t, \psi_s]$. We assume embedding dimension $d = 2$, i.e., only a single rotation block $\mathbf{R}_{i,\theta}$ acts on the embeddings.

950 Recall the squared Lorentzian distance for any $a, b \in \mathbb{L}^{K,d}$,

$$\begin{aligned} d_{\mathcal{L}}^2(a, b) &= \|a - b\|_{\mathcal{L}}^2 \\ &= \frac{2}{K} - 2\langle a, b \rangle_{\mathcal{L}} \\ &= \frac{2}{K} - 2(-a_t b_t + \mathbf{a}_s^\top \mathbf{b}_s). \end{aligned}$$

951 Without HOPE,

$$\begin{aligned} -d_{\mathcal{L}}^2(\mathbf{Q}_i, \mathbf{K}_j) &= -\left[\frac{2}{K} - 2(-\psi_t \psi_t + \psi_s^\top \psi_s) \right] \\ &= -\left[\frac{2}{K} + 2\psi_t \psi_t - 2\psi_s^\top \psi_s \right]. \end{aligned}$$

952 Using HOPE,

$$\begin{aligned} -d_{\mathcal{L}}^2(\mathbf{R}_{i,\theta} \mathbf{Q}_i, \mathbf{R}_{j,\theta} \mathbf{K}_j) &= -\left[\frac{2}{K} + 2\psi_t^2 - 2\left((\mathbf{R}_{i,\theta} \psi_s)^\top (\mathbf{R}_{j,\theta} \psi_s) \right) \right] \\ &= -\left[\frac{2}{K} + 2\psi_t^2 - 2\left(\psi_s^\top \mathbf{R}_{j-i,\theta} \psi_s \right) \right] \\ &= -\left[\underbrace{\frac{2}{K} + 2\psi_t^2}_C - 2\|\psi_s\|^2 \cos((j-i)\theta) \right] \\ &= -\left[C - 2\|\psi_s\|^2 \cos((j-i)\theta) \right]. \end{aligned} \tag{13}$$

953 Using Lemma A.1. from [3], when $j = i$, we have $(j-i)\theta \equiv 0 \pmod{2\pi}$. Next, let us define $\mathbf{a}_{i,i}$
 954 as $-[C - 2\|\psi_s\|^2]$. This means $\cos((j-i)\theta) = \cos(0) = 1$, and $\mathbf{a}_{i,j} < \mathbf{a}_{i,i}$. This gives us the
 955 following self-attention score:

$$\begin{aligned} \nu_{i,i} &= \frac{\exp(\mathbf{a}_{i,i})}{\sum_{k < i} \exp(\mathbf{a}_{i,k}) + \exp(\mathbf{a}_{i,i})} \\ &= \frac{\exp(-[C - 2\|\psi_s\|^2])}{\sum_{k < i} \exp(-[C - 2\|\psi_s\|^2 \cos((k-i)\theta)]) + \exp(-[C - 2\|\psi_s\|^2])} \\ &= \frac{1}{1 + \sum_{k < i} \exp(-[C - 2\|\psi_s\|^2 \cos((k-i)\theta)] - (-[C - 2\|\psi_s\|^2])} \\ &= \frac{1}{1 + \sum_{k < i} \exp(-C + 2\|\psi_s\|^2 \cos((k-i)\theta) + C - 2\|\psi_s\|^2)} \\ &= \frac{1}{1 + \sum_{k < i} \exp(2\|\psi_s\|^2(\cos((k-i)\theta) - 1))}, \end{aligned}$$

956 for all $k \neq i$, $\cos((k-i)\theta) < 1$. This means $2\|\psi_s\|^2(\cos((k-i)\theta) - 1) < 0$. To that end, we
 957 obtain

$$\sup_{\|\psi_s\|^2 \rightarrow \infty} \frac{1}{1 + \sum_{k < i} \exp(2\|\psi_s\|^2(\cos((k-i)\theta) - 1))} = 1.$$

958 This guarantees $\nu_{i,i} > 1 - \epsilon$ for $\epsilon > 0$, where ϵ is a function of $2\|\psi_s\|^2$.

959 We now consider the off-diagonal pattern. Set $\mathbf{Q}_i = \psi$ for non-trivial $\psi = [\psi_t, \psi_s] \in \mathbb{L}^{K,d}$. Set keys
 960 $\mathbf{K}_i = \mathbf{R}_{1,\theta} \psi$ and define $\mathbf{a}_{i,i-1}$ as the off-diagonal input to the softmax when computing $\nu_{i,i-1}$. To

961 that end, we have

$$\begin{aligned}
\mathbf{a}_{i,i-1} &= -d_{\mathcal{L}}^2(\mathbf{R}_{i,\theta}\mathbf{Q}_i, \mathbf{R}_{i-1,\theta}\mathbf{K}_i) \\
&= -\left[\frac{2}{K} + 2\psi_t^2 - 2\left((\mathbf{R}_{i,\theta}\mathbf{Q}_i)^\top(\mathbf{R}_{i-1,\theta}\mathbf{K}_i)\right)\right] \\
&= -\left[\frac{2}{K} + 2\psi_t^2 - 2\left((\mathbf{R}_{1,\theta}\psi_s)^\top(\mathbf{R}_{i-1,\theta}\mathbf{R}_{1,\theta}\psi_s)\right)\right] \\
&= -\left[\frac{2}{K} + 2\psi_t^2 - 2\left((\mathbf{R}_{i,\theta}\psi_s)^\top(\mathbf{R}_{i,\theta}\psi_s)\right)\right] \\
&= -\left[\frac{2}{K} + 2\psi_t^2 - 2\left(\|\psi_s\|^2 \cos((i-i)\theta)\right)\right] \\
&= -\left[\frac{2}{K} + 2\psi_t^2 - 2\|\psi_s\|^2\right].
\end{aligned}$$

962 Use the same reasoning from the diagonal case to show that attention head with HOPE can learn
963 off-diagonal patterns. This concludes the proof. \square

964 A.7 Proposition 4.5: Invariance guarantees of Hyperbolic RMSNorm

965 **Proposition.** $\text{RMSNorm}_{\mathcal{L}}$ is invariant to scaling of inputs \mathbf{x} during both the forward and backward
966 passes.

967 *Proof.* Euclidean RMSNorm is invariant to input-scaling, both during the forward and backward
968 pass. We observe similar guarantees from our formulation of hyperbolic RMSNorm. To that end, we
969 first prove the input-scaling invariance of Euclidean RMSNorm.

970 Given an input $\mathbf{x} \in \mathbb{R}^n$ and a feed-forward network with parameters $\mathbf{W} \in \mathbb{R}^{n \times m}$,

$$\mathbf{y} = \sigma\left(\frac{\mathbf{W}^\top \mathbf{x}}{\text{RMS}(\mathbf{W}^\top \mathbf{x})} \odot \mathbf{g} + \mathbf{b}\right), \quad \text{RMS}(\mathbf{a}) = \sqrt{\frac{1}{m} \sum_{k=1}^m \mathbf{a}_k^2}.$$

971 Here, \mathbf{g} is a learnable gain parameter, initially set to 1, that re-scales the standardized inputs and \mathbf{b} is
972 a bias term.

973 Suppose the weights are scaled by a small factor, $\mathbf{W}' = \delta \mathbf{W}$. First, observe that the root mean
974 squared operation, RMS, is input-scaling invariant: $\text{RMS}(\alpha \mathbf{a}) = \alpha \text{RMS}(\mathbf{a})$. It is then evident that
975 the final output of RMSNorm is also scale-invariant:

$$\begin{aligned}
\mathbf{y}' &= \sigma\left(\frac{(\mathbf{W}')^\top \mathbf{x}}{\text{RMS}((\mathbf{W}')^\top \mathbf{x})} \odot \mathbf{g} + \mathbf{b}\right) \\
&= \sigma\left(\frac{\delta \mathbf{W}^\top \mathbf{x}}{\text{RMS}(\delta \mathbf{W}^\top \mathbf{x})} \odot \mathbf{g} + \mathbf{b}\right) \\
&= \sigma\left(\frac{\cancel{\delta} \mathbf{W}^\top \mathbf{x}}{\cancel{\delta} \text{RMS}(\mathbf{W}^\top \mathbf{x})} \odot \mathbf{g} + \mathbf{b}\right) = \mathbf{y}
\end{aligned} \tag{14}$$

976 A similar argument can be made for a scaling of the inputs \mathbf{x} . Since hyperbolic RMSNorm uses
977 Euclidean RMSNorm internally, it offers the same invariance guarantees as it operates solely on the
978 space dimension of the Lorentzian input.

Given an input $\mathbf{x} = [x_t, \mathbf{x}_s] \in \mathbb{L}^{K,n}$, we know $\text{RMSNorm}(\delta\mathbf{x}) = \text{RMSNorm}(\mathbf{x})$ for some scaling factor δ . As such,

$$\begin{aligned} \mathbf{y}' &= \text{RMSNorm}_{\mathcal{L}}(\delta\mathbf{x}) \\ &= \left[\sqrt{\|\text{RMSNorm}(\delta\mathbf{x}_s)\| - 1/K}, \text{RMSNorm}(\delta\mathbf{x}_s) \right]^\top \\ &= \left[\sqrt{\|\text{RMSNorm}(\mathbf{x}_s)\| - 1/K}, \text{RMSNorm}(\mathbf{x}_s) \right]^\top = \mathbf{y}. \end{aligned}$$

Next, we analyze the gradient stability of hyperbolic RMSNorm. In Euclidean RMSNorm, for a given loss L , we are interested in computing three gradients: $\frac{\partial L}{\partial \mathbf{g}}$ for the gain parameter, $\frac{\partial L}{\partial \mathbf{b}}$ for the bias, and $\frac{\partial L}{\partial \mathbf{W}}$ for the weights. We compute $\frac{\partial L}{\partial \mathbf{g}}$ and $\frac{\partial L}{\partial \mathbf{b}}$ as follows:

$$\frac{\partial L}{\partial \mathbf{b}} = \frac{\partial L}{\partial \mathbf{v}} \cdot \frac{\partial \mathbf{v}}{\partial \mathbf{b}} \quad \frac{\partial L}{\partial \mathbf{g}} = \frac{\partial L}{\partial \mathbf{v}} \odot \frac{\mathbf{W}^\top \mathbf{x}}{\text{RMS}(\mathbf{W}^\top \mathbf{x})},$$

where \mathbf{v} denotes the inputs to the activation σ . These gradients are invariant to the scaling of Euclidean inputs x and weights \mathbf{W} , trivially for $\frac{\partial L}{\partial \mathbf{b}}$, and due to the linearity established in Equation (14) for $\frac{\partial L}{\partial \mathbf{g}}$. Computing $\frac{\partial L}{\partial \mathbf{W}}$ is more involved due to the quadratic computation in RMS, but also provides invariance to input scaling as shown by Zhang and Sennrich [51].

Given an input $\mathbf{x} = [x_t, \mathbf{x}_s] \in \mathbb{L}^{K,n}$, we know $\frac{\partial L}{\partial \mathbf{g}}$, $\frac{\partial L}{\partial \mathbf{b}}$, and $\frac{\partial L}{\partial \mathbf{W}}$ are scaling-invariant in the backward pass since hyperbolic RMSNorm uses Euclidean RMSNorm. Thus, for any scaled hyperbolic input $\delta\mathbf{x} \in \mathbb{L}^{K,n}$, we get scaling invariance both in the time and space dimension during the backward pass. \square

B Additional Details

B.1 MiCE as a Lorentzian Module

In this section, we expand on the fact that MiCE is indeed a valid hyperbolic module throughout. Note that since Equation (6) consists of the combination of a Lorentzian residual connection [24] and Lorentzian centroid [28], it suffices to show that the projection from input manifold to expert manifold, and the reverse projection, are valid projections between Lorentz hyperbolic spaces. In fact, it suffices to show that given $\mathbf{x} \in \mathbb{L}^{K_1,n}$, we have $\sqrt{K_1/K_2}\mathbf{x} \in \mathbb{L}^{K_2,n}$. To see this, note that

$$\begin{aligned} \left\langle \sqrt{K_1/K_2}\mathbf{x}, \sqrt{K_1/K_2}\mathbf{x} \right\rangle_{\mathcal{L}} &= \frac{K_1}{K_2} \langle \mathbf{x}, \mathbf{x} \rangle_{\mathcal{L}} \\ &= \frac{K_1}{K_2} \cdot \frac{1}{K_1} \quad (\mathbf{x} \in \mathbb{L}^{K_1,n}) \\ &= \frac{1}{K_2} \end{aligned}$$

Thus, $\sqrt{K_1/K_2}\mathbf{x} \in \mathbb{L}^{K_2,n}$ as desired. Then, each projection via scaling by $\sqrt{K/K_{s,i}}$ and $\sqrt{K/K_{r,i}}$ indeed map the input vector \mathbf{x} to the expert manifold, and the projection via $\sqrt{K_{s,i}/K}$ and $\sqrt{K_{r,i}/K}$ maps the output of the experts back to the input manifold. As a result, every vector in Equation (6) lives on the input manifold, hence the output lives in $\mathbb{L}^{K,n}$ as desired.

Additionally, note that since the squared Lorentzian distance is given by $d_{\mathcal{L}}^2(\mathbf{x}, \mathbf{y}) = 2/K - 2\langle \mathbf{x}, \mathbf{y} \rangle = 2/K + 2x_t y_t - 2\mathbf{x}_s^\top \mathbf{y}_s$, it scales inversely w.r.t. $\mathbf{x}_s^\top \mathbf{y}_s$. As a result, the gating score obtained through Equation (5) is minimizing the the squared hyperbolic distance between the input token vector \mathbf{x}_t and the vector \mathbf{y}_j (by viewing centroid vector \mathbf{y}_j as the space-like dimension of a Lorentz hyperbolic vector). Therefore, the gating module is in fact a hyperbolic module as well.

B.2 Hyperbolic Multi-Head Latent Attention

In this section, we provide the details for *hyperbolic Multi-Head Latent Attention* (HMLA). Let $\mathbf{x}_t \in \mathbb{L}^{K, nh_n}$ be the t -token, where n is the embedding dimension and h_n is the number of heads. Let $\mathbf{W}^{DKV} \in \mathbb{R}^{(nh_n+1) \times n_{kv}}$ be the downward projection matrix of the keys and values, and

1012 $\mathbf{W}^{DQ} \in \mathbb{R}^{(nnh+1) \times n_q}$ be the downward projection matrix of the query ($n_{kv}, n_q \ll nh_n$). We
 1013 first compress the token into the latent spaces via $\mathbf{c}_t^{KV} = \text{HLT}(\mathbf{x}_t; \mathbf{W}^{KV}, \mathbf{b}^{KV}) \in \mathbb{L}^{K, n_{kv}}$, $\mathbf{c}_t^Q =$
 1014 $\text{HLT}(\mathbf{x}_t; \mathbf{W}^Q, \mathbf{b}^Q) \in \mathbb{L}^{K, n_q}$. We then project the latent query, key, and value vectors back to the
 1015 higher dimensional spaces. Specifically, let $\mathbf{W}^{UV}, \mathbf{W}^{UK} \in \mathbb{R}^{(n_{kv}+1) \times nh_n}$ be the upward projection
 1016 matrix of the keys and values, and let $\mathbf{W}^{UQ} \in \mathbb{R}^{(n_q+1) \times nh_n}$ be the upward projection matrix of the
 1017 query. Then, the final projected keys, values, and queries are

$$\begin{aligned} [\mathbf{k}_{t,1}^C; \dots; \mathbf{k}_{t,h_n}^C] &= \text{HLT}(\mathbf{c}_t^{KV}; \mathbf{W}^{UK}, \mathbf{b}^{UK}); [\mathbf{v}_{t,1}^C; \dots; \mathbf{v}_{t,h_n}^C] = \text{HLT}(\mathbf{c}_t^{KV}; \mathbf{W}^{UV}, \mathbf{b}^{UV}) \\ [\mathbf{q}_{t,1}^C; \dots; \mathbf{q}_{t,h_n}^C] &= \text{HLT}(\mathbf{c}_t^Q; \mathbf{W}^{UQ}, \mathbf{b}^{UQ}). \end{aligned} \quad (15)$$

1018 Following previous works [12, 11], as RoPE is incompatible with MLA due to position coupling,
 1019 we employ a decoupled HOPE scheme with HMLA, where we use additional query vectors with
 1020 a shared key. Let $\mathbf{W}^{QR} \in \mathbb{R}^{(n_q+1) \times (h_n n_r)}$ and $\mathbf{W}^{KR} \in \mathbb{R}^{(n_{kv}+1) \times n_r}$ be the upward projection
 1021 matrix of the decoupled queries and the shared key respectively, where n_r is the dimension per head.
 1022 We apply HOPE to these vectors to obtain the position-encoded vectors

$$[\mathbf{q}_{t,1}^R; \dots; \mathbf{q}_{t,h_n}^R] = \text{HoPE}(\text{HLT}(\mathbf{c}_t^Q; \mathbf{W}^{QR}, \mathbf{b}^{QR})); \mathbf{k}_t^R = \text{HoPE}(\text{HLT}(\mathbf{c}_t^K; \mathbf{W}^{KR}, \mathbf{b}^{KR})). \quad (16)$$

1023 Then, we obtain the final query and key vectors as

$$\mathbf{q}_{t,i} = \text{HCat}(\mathbf{q}_{t,i}^C; \mathbf{q}_{t,i}^R); \mathbf{k}_{t,i} = \text{HCat}(\mathbf{k}_{t,i}^C; \mathbf{k}_t^R), \quad (17)$$

1024 where HCat denotes hyperbolic concatenation [47, 36]. The attention score is computed based on
 1025 negative squared Lorentz distance similar to Equation (3) as

$$\mathbf{o}_{t,i} = \frac{\sum_{j=1}^N \alpha_{t,i,j} \mathbf{v}_{t,j}^C}{\sqrt{-K} \|\sum_{k=1}^N \alpha_{t,i,k} \mathbf{v}_{t,j}^C\|_{\mathcal{L}}}; \alpha_{t,i,j} = \frac{\exp(-d_{\mathcal{L}}^2(\mathbf{q}_{t,i}, \mathbf{k}_{t,j})/\sqrt{h_n + n_r})}{\sum_{k=1}^N \exp(-d_{\mathcal{L}}^2(\mathbf{q}_{t,i}, \mathbf{k}_{t,k})/\sqrt{h_n + n_r})}. \quad (18)$$

1026 The final output of HMLA can be expressed as the concatenation of the hyperbolic vector

$$\text{HMLA}(\mathbf{X}_t; h_n, n, n_r, n_q, n_{kv}) = \text{HLT}\left(\left[\sqrt{\|\mathbf{o}_t\| - 1/K}, \mathbf{o}_t\right]^\top; \mathbf{W}^O, \mathbf{b}^O\right), \quad (19)$$

1027 where $\mathbf{o}_t = [\mathbf{o}_{t,1}, \dots, \mathbf{o}_{t,h_n}]$ and $\mathbf{W}^O \in \mathbb{R}^{h_n(n+1) \times h_n n}$ is the out-project matrix. HMLA enables
 1028 HELM models to improve computational efficiency during training and inference compared to the
 1029 regular hyperbolic self-attention in Equation (3).

1030 B.3 Hyperbolic SwiGLU Feedforward Network

1031 In this section, we introduce hyperbolic SwiGLU feedforward networks (FFNs), whose Euclidean
 1032 formulation is widely used in LLMs [19, 11]. This differs from previous FNNs used in hyperbolic
 1033 Transformers in the need for feature multiplication and activation function [6, 7, 47]. Let $\mathbf{x} \in \mathbb{L}^{K,n}$
 1034 be the input tokens, $\mathbf{W}_1, \mathbf{W}_3 \in \mathbb{R}^{(n+1) \times m}$ be the weights of internal projection layers and $\mathbf{W}_2 \in$
 1035 $\mathbb{R}^{(m+1) \times n}$ be the weight matrix of the outward projection layer. Then, the hyperbolic SwiGLU FNN
 1036 HFFN_{SG} : $\mathbb{L}^{K,n} \rightarrow \mathbb{L}^{K,n}$ is given by

$$\begin{aligned} \text{HFFN}_{SG}(\mathbf{x}) &= \text{HLT}\left(\left[\sqrt{\|\mathbf{y}\| - 1/K}, \mathbf{y}\right]^\top; \mathbf{W}_2, \mathbf{b}_2\right) \\ \mathbf{y} &= \text{SiLU}_{\mathcal{L}}(\text{HLT}(\mathbf{x}; \mathbf{W}_1, \mathbf{b}_1)) \otimes_s \text{HLT}(\mathbf{x}; \mathbf{W}_3, \mathbf{b}_3), \end{aligned} \quad (20)$$

1037 where $\text{SiLU}_{\mathcal{L}}$ denotes SiLU activation using the HRC activation operations from Hypformer [47]
 1038 and \otimes_s denotes multiplication on the space dimension of a Lorentz vector, i.e. $\mathbf{x} \otimes_s \mathbf{y} = \mathbf{x}_s \mathbf{y}_s$.

1039 C Training and Evaluation Details

1040 In this section we detail the training and evaluation setup for the experiments.

1041 C.1 Models Setup

1042 Here we detail the model setup for all the models we used in the experiments.

1043 **HELM-MiCE model setup.** Here we follow the notation in Appendix B.2 and Section 4.2. For
 1044 HELM-MiCE, we used HMLA as the attention mechanism in the attention block, MiCE as the
 1045 sparse feedforward network, and HFNN_{SG} as the dense feedforward network. For both sizes, only
 1046 the first decoder block uses the dense layer and the rest of the blocks using the MiCE layer. The
 1047 MiCE layers use HFNN_{SG} as well for its feedforward component. For the dense layer, we set the
 1048 intermediate dimension to be $4h_n n$. For the MiCE layers, we set the intermediate dimension of
 1049 HFNN_{SG} as $2h_n n$.

1050 For the $\sim 100M$ sized model, we used 6 total layers, 6 heads ($n_h = 6$) each with $n = 64$, and we
 1051 set $n_{kv} = 64$, $n_r = 16$. For MiCE layers, we employ 4 experts with 2 active experts per token
 1052 ($N_r = 4$, $K_r = 2$). We use one shared expert ($N_s = 1$). For the curvatures of the routed experts,
 1053 we set them to be uniform from -0.1 to 2.0 . The curvature of the shared expert is set to be -1 .
 1054 The curvature of the entire model is set to -1 as well. For HMLA layers, in practice, the upward
 1055 projection matrices do not need to project back to the full dimension of $h_n n$. Due to compute
 1056 restraints, we instead employ a *reduction in dimensionality* during the upward projection, where
 1057 $\mathbf{W}^{UK}, \mathbf{W}^{UV} \in \mathbb{R}^{(n_{kv}+1) \times h_n n/2}$, and the outward projection matrix \mathbf{W}^O projects back to the full
 1058 dimensionality of the input with $\mathbf{W}^O \in \mathbb{R}^{h_n(n/2+1) \times h_n n}$.

1059 For the $\sim 1B$ sized model, we use 16 total layers, 14 heads ($n_h = 14$) each with $n = 64$, and we
 1060 set $n_{kv} = 256$, $n_r = 64$. For MiCE layers, we employ 4 experts with 2 active experts per token
 1061 ($N_r = 8$, $K_r = 2$). We use one shared expert ($N_s = 1$). For the curvatures of the routed experts, we
 1062 set them to be uniform from -0.1 to 2.0 . The curvature of the shared expert is set to be -1 . The
 1063 curvature of the entire model is set to -1 as well. We do not use the same reduction in dimensionality
 1064 during upward projection as we did in the $\sim 100M$ case to enable for more expressive attention
 1065 modules.

1066 **HELM-D model setup.** For the HELM-D model, we only train the $100M$ sized model. Here, we
 1067 use 6 layers, 6 heads each with dimension 64, and we set the intermediate dimension of the HFNN_{SG}
 1068 feedforward networks to be 4 times the total model dimension. We set the overall curvature of the
 1069 model to -1 . All hyperbolic models are built on top of HyperCore He et al. [23].

1070 **Baseline models setup.** For the baseline models, we set them up to have identical dimensionality
 1071 as the HELM models. In particular, for the LLaMA model we train, we use the same number of
 1072 layers, heads, and dimensionality per head as the feedforward network. For the DeepSeek models
 1073 we train, we use the same number of layers, heads, dimensionality per head, dimensionality for the
 1074 feedforward network, dimensionality for the MoE modules, number of routed and shared experts,
 1075 and the same dimensionality in the MLA layers.

1076 **Hyperbolic work embeddings.** For the smaller HELM models, we map the input tokens directly
 1077 to Lorentz hyperbolic vectors, which are then trained as hyperbolic parameters via Riemannian
 1078 optimizers. The parameters are initialized via wrapped Gaussian normal distribution on the manifold.
 1079 However, when training the $\sim 1B$ HELM-MiCE model, we found this to cause training instability.
 1080 As a result, for the larger model, we first map the tokens to the space-like dimension of Lorentz
 1081 hyperbolic vectors, and then compute the time-like dimension of the vectors afterwards. We found
 1082 this to stabilize model training.

1083 C.2 Training Details

1084 **Dataset.** For the training dataset, we use the English portion of the Wikipedia dataset [15]. This
 1085 dataset consists of $\sim 6.4M$ rows of data. We download the dataset directly from Huggingface. The
 1086 raw text data is then passed through the LLaMA3.1-8B tokenizer [19], which has a vocabulary size
 1087 of $\sim 128K$. We use a sequence length of 2048 for all models. Samples longer than 2048 tokens were
 1088 broken up into multiple samples, with the trailing tailed dropped. The tokenized dataset consist of
 1089 roughly $4.5B \sim 5B$ tokens. For training efficiency, as we measured the average number of tokens
 1090 per sample is ~ 700 across the dataset, we used sample packing with a packing ratio of 3.0. Then
 1091 packed samples shorted than 2048 tokens are then padded on the right.

Table 3: Ablation accuracy, where we compare HELM-MiCE with a variant using hyperbolic Multi-Head self-Attention instead of HMLA, denoted as MiCE-HMHA. Bolding denotes the highest accuracy and underline denotes the second-highest. Euclidean DEEPSEEK V3 results are shown for reference. Overall, HELM-MiCE consistently achieves the highest accuracy, while both hyperbolic models still outperform the Euclidean counterpart.

Model	# Params	CommonsenseQA 0-Shot	HellaSwag 0-Shot	OpenbookQA 0-Shot	MMLU 5-Shot	ARC-Challenging 5-Shot	Avg -
DEEPSEEK V3	120M	19.2	25.2	23.4	24.2	21.8	22.8
MiCE-HMHA	120M	19.3	<u>25.7</u>	<u>26.0</u>	23.8	25.3	<u>23.7</u>
HELM-MiCE	120M	19.3	26.2	27.4	24.7	<u>24.1</u>	24.2

Table 4: Ablation accuracy, where we compare HELM with a variants using learned relative positional encoding instead of HOPE, denoted as HELM-D-L and HELM-MiCE-L. Bolding denotes the highest accuracy and underline denotes the second-highest. Euclidean results are shown for reference. Overall, HELM-MiCE and HELM-D consistently achieves the higher accuracy, while both hyperbolic models still outperform the Euclidean counterpart.

Model	# Params	CommonsenseQA 0-Shot	HellaSwag 0-Shot	OpenbookQA 0-Shot	MMLU 5-Shot	ARC-Challenging 5-Shot	Avg -
LLAMA	115M	21.1	25.3	25.3	23.8	21.0	23.3
HELM-D-L	115M	19.7	25.5	28.6	23.0	21.8	23.7
HELM-D	115M	<u>20.1</u>	<u>25.9</u>	27.0	25.8	21.2	24.0
DEEPSEEK V3	120M	19.2	25.2	23.4	24.2	21.8	22.8
HELM-MiCE-L	120M	19.0	25.5	27.0	23.0	25.7	24.0
HELM-MiCE	120M	19.3	26.0	<u>27.4</u>	<u>24.7</u>	<u>23.5</u>	24.2

Pipeline setup. For training, we set up data-parallelism with Huggingface Accelerate. We use an effective batch size of $\sim 2M$ tokens (including padding). To ensure a fair comparison between the hyperbolic and Euclidean models, we use a learning rate of $2e-4$ for all dense models and a learning rate of $4e-4$ for the MoE and MiCE models. A weight decay rate of 0.01 was used for all models. For the HELM-MiCE models and the DeepSeek models, in order to balance the load between each expert, we utilize the auxiliary-loss-free load balancing strategy and the complementary sequence-wise auxiliary loss during training. The former punishes extreme load imbalance among the experts by dynamically updating a bias term during the gating module, while not needing an explicit auxiliary loss computation for better training efficiency. The latter punishes extreme load imbalance for any particular sequence. All training used a cosine annealing learning rate scheduler with a final target learning rate of $0.1 \times$ the initial learning rate, with 3% of the gradient update steps used as warmup steps.

Runtime. We empirically observe that HELM models take roughly 1.5 to 1.8 times the training of their Euclidean counterparts. For example, the larger $\sim 1B$ HELM-MiCE model takes roughly 72 hours to train on 4 NVIDIA A800s while the similarly sized DeepSeekV3 model takes roughly 40 hours on the same machine.

C.3 Evaluation Details

We use the Language Model Evaluation Harness library (github.com/EleutherAI/lm-evaluation-harness) for all evaluations, where the framework prompts the models with the answers choices to each question and picks the one with the highest likelihood value. For OpenbookQA, we convert the answer choices from full sentences to letter choices for all models, to make up for the relatively smaller model and training dataset sizes.

D Ablation Studies

In this section, we perform additional ablation studies to access the effectiveness of HOPE and HMLA.

1117 **D.1 Ablation for HMLA**

1118 Past works have found that in the Euclidean case, Multi-Head Latent Attention can achieve comparable
1119 and in some cases even superior performance compared to regular Multi-Head Attention [11]. Here
1120 we assess the effectiveness of HMLA against hyperbolic Multi-Head self-Attention. We train
1121 HELM-MICE with the same setup, where we replace the HMLA layers with a hyperbolic Multi-
1122 Head self-Attention layer as given in Equation (3). We denote this model as MICE-HMHA. The
1123 results are shown in Table 3. HELM-MICE outperforms MICE-HMHA in 3 out of the 5 tasks,
1124 achieving the same accuracy for 1 task, with the MICE-HMHA achieving better accuracy in the last
1125 task. The results demonstrate the effectiveness of HELM-MICE while significantly reducing the
1126 memory footprint of the KV-cache. Both hyperbolic models still outperform the Euclidean model,
1127 demonstrating the effectiveness of HELM in general.

1128 **D.2 Ablation for HOPE**

1129 In this section we assess the effectiveness of HOPE against other hyperbolic positional encoding
1130 methods, namely the learned relative positional encoding from Hypformer [47]. We devise a variant
1131 of HELM-D, denoted as HELM-D-L and a variant of HELM-MICE, denoted as HELM-MICE-L,
1132 where each model uses the learned positional encoding instead of HOPE. The results are shown
1133 in Table 4. Overall both HELM-MICE and HELM-D outperform their counterparts that use
1134 learned positional encoding instead of HOPE. Interestingly, however, HELM-MICE-L and HELM-D-
1135 L outperformed HELM-MICE and HELM-D respectively on the ARC-Challenging benchmark,
1136 possibly due to better alignment with reasoning prompts with non-uniform encodings. Nevertheless,
1137 the results demonstrate the effectiveness of HOPE over learned positional encodings in 4 out of the 5
1138 tasks.