Geo-Sign: Hyperbolic Contrastive Regularisation for Geometrically Aware Sign Language Translation

Edward Fish
CVSSP, University of Surrey
edward.fish@surrey.ac.uk

Richard Bowden
CVSSP, University of Surrey
r.bowden@surrey.ac.uk

Abstract

Recent progress in Sign Language Translation (SLT) has focussed primarily on improving the representational capacity of large language models to incorporate Sign Language features. This work explores an alternative direction: enhancing the geometric properties of skeletal representations themselves. We propose Geo-Sign, a method that leverages the properties of hyperbolic geometry to model the hierarchical structure inherent in sign language kinematics. By projecting skeletal features derived from Spatio-Temporal Graph Convolutional Networks (ST-GCNs) into the Poincaré ball model, we aim to create more discriminative embeddings, particularly for fine-grained motions like finger articulations. We introduce a hyperbolic projection layer, a weighted Fréchet mean aggregation scheme, and a geometric contrastive loss operating directly in hyperbolic space. These components are integrated into an end-to-end translation framework as a regularisation function, to enhance the representations within the language model. This work demonstrates the potential of hyperbolic geometry to improve skeletal representations for Sign Language Translation, improving on SOTA RGB methods while preserving privacy and improving computational efficiency. Code available here: https://github.com/ed-fish/geo-sign.

1 Introduction

Sign Languages are rich, multi-channel linguistic systems where meaning is conveyed through a composition of movements involving the upper body, hands, face, and mouth. Automatic Sign Language Translation (SLT) is an established research area focused on developing methods to convert these visual expressions directly into text. While Sign Languages are expressed via fluid multi-articulator kinematics, a persistent challenge for SLT methods lies in creating feature representations that concurrently preserve fine-grained, local details (e.g., subtle finger configurations) while embedding the global structure inherent in larger, overarching body motions. Effectively modelling these multi-scale and relational dynamics within a suitable geometric embedding space remains a central hurdle.

Spatio-Temporal Graph Convolutional Networks (ST-GCNs) offer a natural way to encode these hierarchical relationships by treating the body's joints and bones as nodes and edges in a graph [79]. However, when their learned representations are projected into standard Euclidean geometry for processing via a Large Language Model (LLM), essential fine-grained relational distances and movements can become blurred. For instance, the sign for "*water*" in American Sign Language (ASL) is communicated by forming a W shape with the fingers and tapping the chin twice (a fine-grained, "leaf-level" articulation), immediately followed by a sweeping hand movement away from the body (a "branch-level" gesture). When these features are aggregated in Euclidean geometry, the large translation and rotation of the wrist could dominate the vector's norm, effectively "pulling" the embedding toward the global motion and compressing the subtle finger tap into a vanishing tail.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

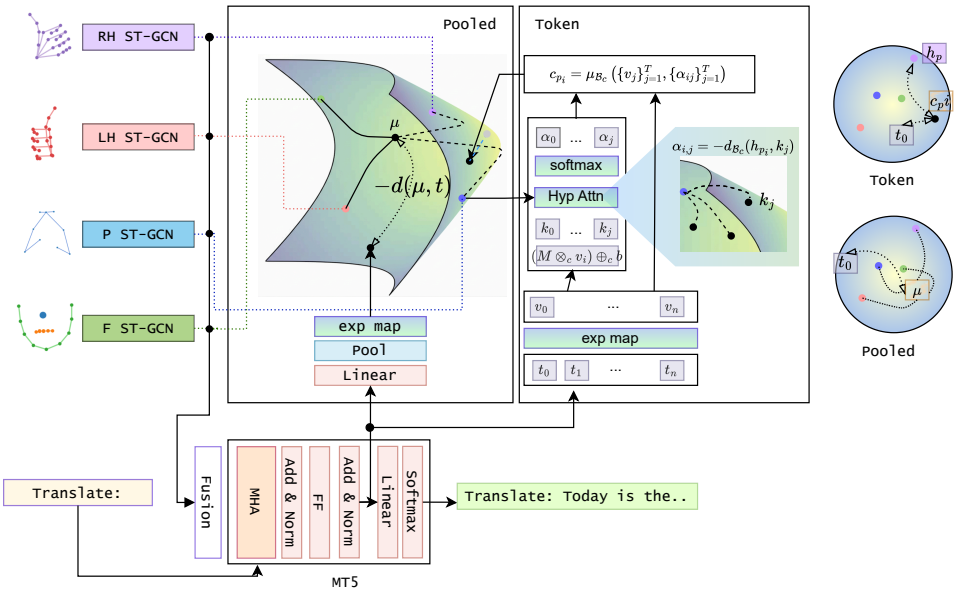


Figure 1: Geo-Sign's hyperbolic framework: (**Left**) Skeletal features from ST-GCN's for different body parts are projected into a Poincaré ball whose curvature is learned, while the original branch fuses the features for processing via the MT5 language model. (**Pooled**) The pose features are aggregated via Frechet Mean in Eq.1, while the text embeddings from the final layer of the MT5 model are pooled and projected to the hyperbolic manifold. Geodesic distance between the text embedding and the mean pose features are minimised for positive samples using the contrastive loss in Eq.5. (**Token**) Alternatively, hyperbolic pose features are used as attention queries against all text embeddings to generate a pose-contextual text embedding. Note the movement of the text features c_{pi} in grey towards the pose feature in blue. (**Right**) A representation of the Poincaré disk demonstrating the difference between Token, and Pooled methods in the tangent space.

Consequently, two signs that differ only in the timing or precision of that tap, which may be critical to lexical meaning, can become nearly indistinguishable once projected into flat Euclidean space.

Large vision-based models [22, 23, 26] appear to be able to implicitly learn these hierarchical structures through extensive video pre-training and visual inductive biases. However, they do so at significant computational cost and with privacy concerns, as they retain identifiable facial and background details that skeletal representations inherently discard.

This work introduces hyperbolic geometry as a means to fundamentally enhance skeletal representations for SLT. Unlike Euclidean space, where volume grows polynomially with radius and can flatten hierarchical structures, hyperbolic manifolds exhibit exponential volume growth. This property is naturally suited to encoding the compositional, tree-like structures found in sign language kinematics. As illustrated in Figure 1, in the Poincaré ball model $\mathbb{B}_c^{d_{\text{hyp}}}$ (with curvature $\kappa = -c < 0$), distances between points near the boundary expand exponentially relative to their Euclidean separation. This provides ample "space" to distinguish nuanced motions (e.g., an open versus a closed fist), while regions near the origin behave more like Euclidean space, suitable for representing broader phrase-level semantics. A key aspect of our approach is that we learn the curvature parameter c end-to-end via Riemannian optimization. This allows the manifold to dynamically adapt its "zoom level": a more negative curvature κ (larger c) amplifies the separation of fine-grained motions, whereas smoother curvature helps preserve sentence-level coherence.

Geo-Sign leverages this geometric inductive bias through a novel regularisation framework for a pre-trained mT5 model [78]. By projecting skeletal features into hyperbolic space and aligning them with text embeddings via a geometric contrastive loss, we guide the mT5 model to internalize the hierarchical nature of sign language kinematics. Our primary contributions include:

- **Hyperbolic Skeletal Representation**: We map multi-part skeletal features, derived from ST-GCNs, into the Poincaré ball using curvature-aware hyperbolic projection layers.
- **Geometric Contrastive Regularisation**: We introduce a contrastive learning objective that operates directly in hyperbolic space, minimizing the geodesic distance between semantically corresponding hyperbolic pose and text embeddings.
- **Hierarchical Aggregation and Alignment Strategies**: We explore two main strategies for this contrastive alignment:
    1. A *global semantic alignment* method, which uses a weighted Fréchet mean to aggregate part-specific hyperbolic embeddings into a single global pose representation, then aligns this with a global text embedding.
    2. A *fine-grained part-text alignment* method, which employs a novel hyperbolic attention mechanism. This allows individual pose part embeddings to attend to specific text tokens within the hyperbolic space, generating contextual text embeddings for more detailed contrastive learning.

This geometric regularisation offers several advantages. It aims to inform the mT5 model's understanding by providing representations that inherently respect kinematic hierarchy. The learnable curvature allows the model to adapt the representational space to dataset-specific characteristics. Furthermore, by relying solely on anonymized skeletal data, our approach inherently preserves signer privacy and offers greater computational efficiency compared to methods requiring extensive video processing.

Experiments on the CSL-Daily benchmark [88] demonstrate Geo-Sign's efficacy. Our skeletal-based approach not only achieves a +1.81 BLEU4 and +3.03 ROUGE score over state-of-the-art pose-based methods but also matches the performance of comparable vision-based networks. We demonstrate that our method extends to American Sign Language and Isolated Sign Language Recognition. Finally, we also present the first method to surpass SOTA gloss based methods (with respect to the ROUGE score) with a gloss-free approach, highlighting the potential of geometrically-aware representations.

2 Related Work

Our work intersects with several research areas: Sign Language Translation (SLT), the use of skeletal data for sign and action recognition, and the application of hyperbolic geometry in machine learning.

2.1 Sign Language Translation (SLT)

Sign Language Translation aims to bridge the communication gap between Deaf and hearing communities by automatically converting sign language videos into spoken or written language text [4, 19, 22, 64, 74]. Distinct from Sign Language Recognition (SLR), which often focuses on isolated signs or gloss transcription [6, 28, 59, 65, 84], SLT tackles the more complex task of translating continuous signing across modalities with potentially disparate grammatical structures.

Early SLT methods often involved a two-stage process: recognizing sign glosses (individual lexical units of sign language grammar) and then translating the gloss sequence into the target language [6, 25, 27, 49, 49, 50, 69–72, 89]. However, this intermediate representation can lead to information loss, while gloss transcriptions are limited in availability. Consequently, end-to-end sequence-to-sequence models have become the dominant paradigm [5]. Initial approaches utilized Recurrent Neural Networks (RNNs) like LSTMs or GRUs, often with attention mechanisms [21, 63]. More recently, Transformer architectures [6] have demonstrated superior performance in capturing long-range dependencies and context [32, 62, 91], enabling direct video-to-text translation [9, 15, 22, 68, 74]. Many recent state-of-the-art architectures leverage large pre-trained language models, such as T5 variants, fine-tuned for the task of SLT [11, 89]. These often rely on large pre-trained visual encoders, with incremental improvements seen by upgrading visual backbones from ResNet [87], to I3D [67], and more recently to ViT variants like DINO [74, 75]. However, as these backbones increase in size, they can limit the number of frames processed concurrently due to quadratically scaling resource demands.

Key challenges in SLT remain, including the scarcity of large-scale annotated datasets [1, 7, 37], handling signer variability, modelling linguistic divergence between sign and spoken languages [12, 73], capturing co-articulation effects [92], and distinguishing visually similar signs [17].

2.2 Skeletal Representations for Sign Language and Action Recognition

Using skeletal keypoints, extracted via pose estimation algorithms like OpenPose [8], MediaPipe [46], or MMPose [13] (in this work we use RTMPose for skeletal features [33]), offers several advantages over raw RGB video for sign language analysis. Skeletal data is computationally efficient, robust to background and lighting variations, directly encodes articulation kinematics, enhances privacy by design, and can potentially improve generalization across different signers and environments [31, 57, 92].

Graph Convolutional Networks (GCNs) and particularly Spatio-Temporal GCNs (ST-GCNs) have shown great promise by explicitly modelling the spatial structure of the skeleton and its temporal dynamics [14, 58, 76, 77, 79]. However, the quality of skeletal data is heavily dependent on the accuracy of the underlying pose estimation algorithms [30]. Furthermore, skeletal data might discard subtle visual cues present in RGB video that could be important for disambiguation. While multi-modal fusion (RGB + pose) has been explored to combine the strengths of both modalities [54, 64, 75, 90], it typically increases computational cost. Our work focuses on enhancing the representational power of skeletal data itself by embedding it in hyperbolic space, aiming to improve its discriminability for SLT without resorting to RGB fusion.

2.3 Hyperbolic Geometry in Machine Learning

Hyperbolic geometry, characterized by its constant negative curvature, offers unique properties for representation learning [18, 61]. Its most notable feature is the exponential growth of volume with radius, which allows hyperbolic spaces to embed tree-like or hierarchical structures with significantly lower distortion than Euclidean spaces. This makes them particularly suitable for data where such latent hierarchies are believed to exist. Common models of hyperbolic geometry used in machine learning include the Poincaré ball model [51] and the Lorentz (or hyperboloid) model [52].

2.4 Hyperbolic Representation Learning Applications

The advantageous properties of hyperbolic spaces for modelling hierarchies have led to their successful application in various domains. Hyperbolic Graph Neural Networks (HGNNs) have extended GNN principles to hyperbolic space, demonstrating strong performance on graph-related tasks, especially those involving scale-free or hierarchical graphs [44, 81]. In Natural Language Processing (NLP), Poincaré embeddings [52] effectively captured word hierarchies (e.g., WordNet taxonomies), leading to the development of hyperbolic RNNs and Transformers for improved modeling of sequential and relational data [82]. Applications in computer vision include hyperbolic Convolutional Neural Networks (CNNs) [2] and vision-language models that leverage hyperbolic spaces to better align visual and textual concept hierarchies [29].

Our work contributes to this growing body of research by applying hyperbolic representation learning specifically to the domain of skeletal Sign Language Translation. While hyperbolic geometry has been explored for general action recognition from skeletons [16, 38, 40] and in broader NLP contexts [48], its systematic application to enhance the discriminability of multi-part skeletal features for end-to-end SLT, particularly through a geometric contrastive loss operating in hyperbolic space to regularize a large language model, represents a novel direction. We aim to leverage the geometric properties of the Poincaré ball to refine skeletal representations as they are processed by the language model, thereby improving the translation quality, especially for signs involving fine-grained hierarchical motion.

3 Methodology

Geo-Sign regularises a pre-trained mT5 model [78] by integrating hyperbolic geometry to capture the hierarchical nature of sign kinematics. We employ the d_{hyp}-dimensional Poincaré ball model, $\mathbb{B}_c^{d_{\text{hyp}}} = \{\mathbf{x} \in \mathbb{R}^{d_{\text{hyp}}} : \|\mathbf{x}\|_2 < 1/\sqrt{c}\}$, with a learnable curvature magnitude $c > 0$. This section first briefly introduces essential hyperbolic operations, then details our pose encoding, hyperbolic projection, and two distinct contrastive alignment strategies.

3.1 Hyperbolic Geometry Essentials

Hyperbolic spaces exhibit exponential volume growth ($V_H(r) \propto e^{(d-1)r}$ for large radius r), making them adept at embedding hierarchies with low distortion compared to Euclidean spaces ($V_E(r) \propto r^d$) [18, 51]. In the Poincaré ball, geometry near the origin ($\|\mathbf{x}\|_2 \approx 0$) is approximately Euclidean, while near the boundary ($\|\mathbf{x}\|_2 \to 1/\sqrt{c}$), distances are magnified, providing capacity to distinguish fine details.

The geodesic distance $d_{\mathbb{B}_c}(\mathbf{u}, \mathbf{v})$ between points $\mathbf{u}, \mathbf{v} \in \mathbb{B}_c^{d_{\text{hyp}}}$ is:

$$d_{\mathbb{B}_c}(\mathbf{u}, \mathbf{v}) = \frac{2}{\sqrt{c}} \operatorname{artanh}\left(\sqrt{c}\, \|(-\mathbf{u}) \oplus_c \mathbf{v}\|_2\right). \tag{1}$$

This utilizes Möbius addition $\oplus_c$, the hyperbolic analogue of vector addition:

$$\mathbf{u} \oplus_c \mathbf{v} = \frac{(1 + 2c\langle \mathbf{u}, \mathbf{v}\rangle_2 + c\|\mathbf{v}\|_2^2)\mathbf{u} + (1 - c\|\mathbf{u}\|_2^2)\mathbf{v}}{1 + 2c\langle \mathbf{u}, \mathbf{v}\rangle_2 + c^2\|\mathbf{u}\|_2^2\|\mathbf{v}\|_2^2}. \tag{2}$$

To map Euclidean vectors $\mathbf{v}$ from the tangent space at the origin $\mathcal{T}_{\mathbf{0}}\mathbb{B}_c^{d_{\text{hyp}}} \cong \mathbb{R}^{d_{\text{hyp}}}$ into $\mathbb{B}_c^{d_{\text{hyp}}}$, we use the exponential map at the origin $\exp_{\mathbf{0}}^c(\cdot)$:

$$\exp_{\mathbf{0}}^c(\mathbf{v}) = \tanh\left(\frac{\sqrt{c}\|\mathbf{v}\|_2}{2}\right) \frac{\mathbf{v}}{\frac{\sqrt{c}}{2}\|\mathbf{v}\|_2}, \quad (\mathbf{v} \neq \mathbf{0}). \tag{3}$$

Its inverse is the logarithmic map at the origin, $\log_{\mathbf{0}}^c(\cdot)$. General maps $\exp_{\mathbf{x}}^c(\cdot)$ and $\log_{\mathbf{x}}^c(\cdot)$ facilitate operations at arbitrary points $\mathbf{x} \in \mathbb{B}_c^{d_{\text{hyp}}}$.

3.2 Skeletal Feature Extraction and Hyperbolic Projection

3.2.1 ST-GCN Backbone

We process 2D skeletal keypoints extracted using RTM-Pose [33], partitioned into four anatomical groups (body, left/right hands, face). Each group is processed by a part-specific ST-GCN [79] which combines spatial graph convolutions with temporal convolutions to model both joint interdependencies and motion dynamics. Residual connections allow information flow from body joints to hand/face representations. The ST-GCNs output part-specific feature maps $\mathbf{Z}_p \in \mathbb{R}^{T \times d'_{\text{gcn\_out}}}$ (T is sequence length). For direct input to the mT5 encoder, these are concatenated and linearly projected to d_{mT5}, yielding dynamic Euclidean pose embeddings $\mathbf{E}_{\text{pose}} \in \mathbb{R}^{T \times d_{\text{mT5}}}$. For the hyperbolic regularisation branch, each $\mathbf{Z}_p$ is temporally mean-pooled to a static summary vector $\bar{\mathbf{f}}_p \in \mathbb{R}^{d'_{\text{gcn\_out}}}$, capturing the overall kinematics of part p.

3.2.2 Part-Specific Projection to Poincaré Ball

Each Euclidean summary vector $\bar{\mathbf{f}}_p$ is projected to a hyperbolic embedding $\mathbf{h}_p \in \mathbb{B}_c^{d_{\text{hyp}}}$. This projection involves a linear transformation of $\bar{\mathbf{f}}_p$ to dimension d_{hyp} using a learnable matrix $\mathbf{W}^p$, followed by multiplication with a learnable positive scalar s_p. This scalar s_p adaptively scales the features in the tangent space, allowing the model to place features from parts with varying motion scales at appropriate "depths" in the hyperbolic space. The resulting tangent vector is then mapped onto the Poincaré ball using the exponential map at the origin (Eq. 3):

$$\mathbf{h}_p = \exp_{\mathbf{0}}^c(s_p \mathbf{W}^p \bar{\mathbf{f}}_p). \tag{4}$$

The set of hyperbolic part embeddings $\{\mathbf{h}_p\}$ forms the input for the subsequent alignment strategies.

3.3 Hyperbolic Contrastive Loss

We regularize the mT5 model by minimizing a Geometric Contrastive Loss, adapted from InfoNCE [53], between hyperbolic pose and text embeddings. This loss encourages semantic consistency by pulling corresponding pose-text pairs closer in hyperbolic space while pushing non-corresponding pairs apart. For a batch of B pose embeddings $\{\mathbf{p}_j\}$ and text embeddings $\{\mathbf{t}_j\}$ in $\mathbb{B}_c^{d_{\text{hyp}}}$, the loss for a positive pair $(\mathbf{p}_i, \mathbf{t}_i)$ is:

$$\mathcal{L}_{\text{hyp\_pair}}(\mathbf{p}_i, \mathbf{t}_i) = -\log \frac{\exp(-d_{\mathbb{B}_c}(\mathbf{p}_i, \mathbf{t}_i)/\tau)}{\sum_{j=1}^{B} \exp(-d_{\mathbb{B}_c}(\mathbf{p}_i, \mathbf{t}_j)/\tau + m \cdot \mathbb{I}(i \neq j))}. \tag{5}$$

Here, $\tau > 0$ is a learnable temperature scaling the similarities (negative distances), and $m \geq 0$ is a learnable additive margin for negative pairs. The total regularisation term $\mathcal{L}_{\text{hyp\_reg}}$ is the batch average of $\mathcal{L}_{\text{hyp\_pair}}$.

3.4 Alignment Methods

We present two methods for selecting features for alignment which offer benefits and trade-offs. The first takes the geometric mean of the pose and the text embeddings which comes at greater computational efficiency but decreased accuracy. The second method uses poses as individual queries over the text embeddings in hyperbolic space. We then compute the distance between each pose and modified token pair. This improves translation accuracy but incurs additional memory and inference costs (Full details provided in the appendix.)

3.4.1 Strategy 1: Global Semantic Alignment (Pooled Method)

This strategy aligns holistic pose and text semantics, promoting high-level understanding.

- **Pose Embedding (p)**: A global hyperbolic pose $\boldsymbol{\mu}_{\text{pose}} \in \mathbb{B}_c^{d_{\text{hyp}}}$ is computed as the weighted Fréchet mean of the part embeddings $\{\mathbf{h}_p\}$. The Fréchet mean is a geometrically sound average in hyperbolic space. Weights $w_p \propto \exp(d_{\mathbb{B}_c}(\mathbf{0}, \mathbf{h}_p))$, normalized via softmax, emphasize parts with more distinct hyperbolic embeddings. The mean is found iteratively (Algorithm 1) using general logarithmic maps $\log_{\mathbf{x}}^c(\cdot)$ and exponential maps $\exp_{\mathbf{x}}^c(\cdot)$ for tangent space computations.

- **Text Embedding (t)**: A global hyperbolic text embedding $\mathbf{h}_{\text{text}} \in \mathbb{B}_c^{d_{\text{hyp}}}$ is obtained by mean-pooling Euclidean token embeddings (e.g., from mT5 decoder's final layer) and then projecting this single sentence vector to $\mathbb{B}_c^{d_{\text{hyp}}}$ using a hyperbolic projection layer (structurally similar to Eq. 4).

The contrastive loss $\mathcal{L}_{\text{hyp\_reg}}$ (Eq. 5) is then computed between the sets of these global pose embeddings $\{\boldsymbol{\mu}_{\text{pose},i}\}$ and global text embeddings $\{\mathbf{h}_{\text{text},i}\}$.

Algorithm 1 Iterative Weighted Fréchet Mean in $\mathbb{B}_c^{d_{\text{hyp}}}$

Require: Hyperbolic embeddings $\{\mathbf{h}_p\}_{p=1}^N$, normalized positive weights $\{w_p\}_{p=1}^N$, c, $I_{\max}$, ϵ_{tol}.
Initialize $\boldsymbol{\mu}^{(0)} \leftarrow \mathbf{h}_1$ (or other suitable initialization).
for $k = 0$ to $I_{\max} - 1$ **do**
  $\mathbf{v}_{\text{agg}} \leftarrow \mathbf{0} \in \mathcal{T}_{\boldsymbol{\mu}^{(k)}} \mathbb{B}_c^{d_{\text{hyp}}}$. ▷ Aggregated tangent vector at current mean
  for $p = 1$ to N **do**
    $\mathbf{v}_{\text{agg}} \leftarrow \mathbf{v}_{\text{agg}} + w_p \log_{\boldsymbol{\mu}^{(k)}}^c(\mathbf{h}_p)$. ▷ Sum weighted log-mapped vectors
  end for
  $\boldsymbol{\mu}^{(k+1)} \leftarrow \exp_{\boldsymbol{\mu}^{(k)}}^c(\mathbf{v}_{\text{agg}})$. ▷ Update mean via exponential map
  Project $\boldsymbol{\mu}^{(k+1)}$ into $\mathbb{B}_c^{d_{\text{hyp}}}$ if numerically necessary.
  if $d_{\mathbb{B}_c}(\boldsymbol{\mu}^{(k+1)}, \boldsymbol{\mu}^{(k)}) < \epsilon_{\text{tol}}$ **then** ▷ Check convergence
    break ▷ Exit loop on convergence
  end if
end for
$\boldsymbol{\mu}_{\text{pose}} \leftarrow \boldsymbol{\mu}^{(k+1)}$ ▷ Assign final mean
Ensure: Estimated Fréchet mean $\boldsymbol{\mu}_{\text{pose}}$.

3.4.2 Strategy 2: Fine-Grained Part-Text Alignment (Token Method)

This strategy aligns individual pose parts (Queries $\boldsymbol{h}_p$) with relevant text segments via hyperbolic attention. For each $\boldsymbol{h}_p \in \mathbb{B}_c^{d_{\text{hyp}}}$ (from Eq. 4), a specific context vector $\boldsymbol{c}_p \in \mathbb{B}_c^{d_{\text{hyp}}}$ is computed.

- **Tokenization:** First, the sequence of T Euclidean text token embeddings, $\{\boldsymbol{e}_t\}_{t=1}^T$, is projected into the Poincaré ball. This yields a sequence of hyperbolic embeddings $V = \{\boldsymbol{v}_t\}_{t=1}^T$, which serve as the values and hold the original semantic meaning of each token.

- **Key Transformation:** The value sequence V is transformed by a learnable Möbius affine transformation (using $\boldsymbol{M}$ and $\boldsymbol{b}$) to create a sequence of keys $K = \{\boldsymbol{k}_t\}_{t=1}^{T}$:

$$\boldsymbol{k}_t = (\boldsymbol{M} \otimes \boldsymbol{v}_t) \oplus \boldsymbol{b} \tag{6}$$

- **Attention Scores:** Scores s_{pt} are computed as the negative geodesic distance between each pose query $\boldsymbol{h}_p$ and each text key $\boldsymbol{k}_t$. A smaller distance signifies greater relevance.

$$s_{pt} = -d_{\mathbb{B}_c}(\boldsymbol{h}_p, \boldsymbol{k}_t) \tag{7}$$

- **Context Vector:** These scores are normalized via softmax (with masking for padding) to produce attention weights α_{pt}. The final context vector $\boldsymbol{c}_p$ is the hyperbolic weighted midpoint (μ) of the original values V weighted by the pose-shifted attention embeddings:

$$\boldsymbol{c}_p = \mu_{\mathcal{B}_c}\left(\{\boldsymbol{v}_t\}_{t=1}^{T}, \{\alpha_{pt}\}_{t=1}^{T}\right) \tag{8}$$

  The final $\mathcal{L}_{\text{hyp\_reg}}$ is the average of K individual contrastive losses (Eq. 5), one for each $(\boldsymbol{h}_p, \boldsymbol{c}_p)$ alignment pair.

3.5 Training Objective and Optimization

The model is trained end-to-end by minimizing the total loss $\mathcal{L}_{\text{total}} = \alpha \cdot \mathcal{L}_{\text{CE}} + (1-\alpha) \cdot \mathcal{L}_{\text{hyp\_reg}}$. This combines the standard cross-entropy translation loss $\mathcal{L}_{\text{CE}}$ (with label smoothing) with the hyperbolic regularisation term $\mathcal{L}_{\text{hyp\_reg}}$ from one of the alignment strategies. The blending factor $\alpha \in [0.1, 1.0]$ is dynamically adjusted during training via a learnable parameter and training progress, allowing an initial focus on $\mathcal{L}_{\text{hyp\_reg}}$ before increasing the influence of $\mathcal{L}_{\text{CE}}$.

Optimization employs AdamW [35, 45] for Euclidean parameters (ST-GCNs, mT5, linear layers), with learning rate 3×10^{-5}. Hyperbolic parameters, including the learnable curvature c (optimized in log-space, e.g., $\log c$) and manifold-constrained parameters, use Riemannian Adam (RAdam) [3] with a comparable learning rate. RAdam adapts updates to the manifold's geometry by operating in tangent spaces. All hyperbolic computations utilize high-precision floating-point numbers (e.g., 'float32') for numerical stability. A key stabilization step before applying any exponential map $\exp_{\mathbf{x}}^{c}(\mathbf{v})$ involves projecting the input tangent vector $\mathbf{v}$ via $\mathbf{v} \leftarrow \mathbf{v} / \max(1, \sqrt{c}\|\mathbf{v}\|_2 + \epsilon)$ for a small $\epsilon > 0$ (e.g., 10^{-5}), ensuring the argument is well-behaved and the output point remains strictly within the Poincaré ball.

4 Experiments

We evaluate Geo-Sign on Chinese Sign Language (CSL) and American Sign Language (ASL). For CSL we use the CSL-Daily dataset [88, 89], a large-scale corpus for Chinese Sign Language to Chinese text translation, comprising over 20,000 videos. For American Sign Language we perform translation experiments on How2Sign [15] and isolated sign language recognition experiments on WLASL2000 [39].

Translation quality is assessed using BLEU [55] (B-1, B-4) and ROUGE-L [42] (R-L) scores where a higher percentage represents a more accurate translation.

4.1 Experimental Setup

Our framework builds upon the Uni-Sign architecture [41], using its pre-trained ST-GCN weights (trained on skeletal features from the CSL-News dataset [41] for CSL and the YTASL [68] dataset for ASL) and an mT5 model [78] as the language decoder. Following Uni-Sign's fine-tuning protocol, which involves 40 epochs of supervised finetuning on CSL-Daily or YTASL, with fused skeletal and RGB features, we remove the RGB encoder and instead apply our hyperbolic regularisation. This allows for a fair comparison of the impact of our geometric regularisation. We investigate both the "Pooled Method" (Strategy 1) and the "Token Method" (Strategy 2) for hyperbolic alignment. To assess the specific contribution of hyperbolic geometry, we also compare against a "Euclidean regularisation" baseline, where the contrastive loss operates on Euclidean projections to the Poincaré ball where curvature is minimal (0.001) and approximately Euclidean. Key hyperparameters for the hyperbolic components (initial curvature $c = 1.5$, dimension $d_{\text{hyp}} = 256$, and $\alpha = 0.70$) are minimally tuned on the development set (further details in the appendix).

Method	Modality		Dev Set			Test Set		
	Pose	RGB	B-1	B-4	R-L	B-1	B-4	R-L
Gloss-Based Methods (Prior Art)								
SLRT [6]	–	✓	37.47	11.88	37.96	37.38	11.79	36.74
TS-SLT [11]	✓	✓	55.21	25.76	55.10	55.44	25.79	55.72
CV-SLT [85]	–	✓	–	<u>28.24</u>	56.36	<u>58.29</u>	<u>28.94</u>	57.06
Gloss-Free Methods (Prior Art)								
MSLU [90]	✓	–	33.28	10.27	33.13	33.97	11.42	33.80
SLRT [6] (Gloss-Free variant)	–	✓	21.03	4.04	20.51	20.00	3.03	19.67
GASLT [83]	–	✓	–	–	–	19.90	4.07	20.35
GFSLT-VLP [88]	–	✓	39.20	11.07	36.70	39.37	11.00	36.44
FLa-LLM [10]	–	✓	–	–	–	37.13	14.20	37.25
Sign2GPT [74]	–	✓	–	–	–	41.75	15.40	42.36
SignLLM [19]	–	✓	42.45	12.23	**39**.18	39.55	15.75	39.91
C^2RL [9]	–	✓	–	–	–	49.32	21.61	48.21
Our Models and Baselines								
Uni-Sign [41] (Pose)	✓	–	53.24	25.27	54.34	53.86	25.61	54.92
Uni-Sign [41] (Pose+RGB)	✓	✓	55.30	26.25	56.03	55.08	26.36	56.51
Geo-Sign (Euclidean Pooled)	✓	–	53.53	25.78	55.38	53.06	25.72	55.57
Geo-Sign (Euclidean Token)	✓	–	53.93	25.91	55.20	54.02	25.98	53.93
Geo-Sign (Hyperbolic Pooled)	✓	–	55.19	26.90	56.93	55.80	27.17	57.75
Geo-Sign (Hyperbolic Token)	✓	–	**55.57**	**27.05**	**57.27**	**55.89**	**27.42**	**57.95**

Table 1: Sign Language Translation performance on the CSL-Daily dataset. BLEU scores (B-1, B-4) and ROUGE-L (R-L) are reported as percentages (%). Higher is better. 'Pose' and 'RGB' indicate input modalities. Uni-Sign is the base architecture sharing pre-training/fine-tuning setups but without our regularisation. Euclidean regularisation applies contrastive loss in Euclidean space. Our Hyperbolic Token method surpasses all other pose-only methods and is competitive with top RGB/multimodal methods. Gloss-based methods that outperform our method are underlined.

4.2 Results on Chinese Sign Language (CSL)

Section 4.2.1 presents our main results on the CSL-Daily test set, comparing Geo-Sign with prior art and baselines. Our Geo-Sign (Hyperbolic Token) model, using only pose data, achieves a test BLEU-4 of 27.42% and ROUGE-L of 57.95%. This represents a significant improvement of +1.81 BLEU-4 and +3.03 ROUGE-L over the strong Uni-Sign (Pose) baseline (25.61% BLEU-4, 54.92% ROUGE-L). Notably, this performance surpasses all other reported gloss-free pose-only methods and is competitive with, or exceeds, several RGB-only and even some gloss-based methods, underscoring the efficacy of our geometric regularisation. The Geo-Sign (Hyperbolic Pooled) variant also outperforms the Euclidean regularisation methods and the Uni-Sign pose baseline, demonstrating the general benefit of hyperbolic geometry. The "Euclidean Token" regularisation already shows improvement over the Uni-Sign baseline, suggesting the contrastive alignment itself is beneficial, but the further gains from hyperbolic geometry are substantial.

4.2.1 Results on American Sign Language

In Table 2 we show results on Sign Language Translation for American Sign Language on the How2Sign [15] dataset. Our method shows increased performance over all pose based methods but performs marginally worse than the best RGB method [60] which benefits from a longer pre-training duration and scale. In Table 3 we also compare our approach on Isolated Sign Language Recognition (ISLR) with the WLASL2000 [39] dataset. For isolated recognition our method shows a small improvement in Top-1 Accuracy for both instance (+0.12) and class-level (+0.57).

4.2.2 Ablation Studies

Ablation studies on the CSL-Daily test set for our best performing Geo-Sign (Hyperbolic Token) model are presented in Table 4. We investigate the impact of the initial hyperbolic curvature c and

Table 2: Sign Language Translation (SLT) results on the How2Sign dataset. Metrics are BLEU (B-1, B-4) and ROUGE-L (R-L). Higher is better.

Method	Modality		Test Set		
	Pose	RGB	B-1	B-4	R-L
Gloss-Free Methods (Prior Art)					
GloFE-VN [43]	✓	–	14.9	2.2	12.6
YouTube-ASL [68]	✓	–	37.8	12.4	-
MSLU [90]	✓	–	20.1	2.4	17.2
SLT-IV [67]	–	✓	34.0	8.0	-
C^2RL [9]	–	✓	29.1	9.4	27.0
FLa-LLM [10]	–	✓	29.8	9.7	27.8
SignMusketeers [20]	–	✓	<u>41.5</u>	14.3	-
SSVP-SLT [60]	–	✓	**43.2**	**15.5**	**38.4**
Our Models and Baselines					
Uni-Sign (Pose)	✓	–	40.4	14.5	34.3
Uni-Sign (Pose+RGB)	✓	✓	40.2	14.9	<u>36.0</u>
Geo-Sign (Token)	✓	–	40.8	<u>15.1</u>	35.4

Table 3: Isolated Sign Language Recognition (ISLR) results on WLASL2000. We report Top-1 Accuracy for Per-Instance (P-I) and Per-Class (P-C). † from [23].

Method	Mod.		Test Acc. (%)	
	P	RGB	P-I	P-C
Prior Art				
ST-GCN† [80]	✓	–	34.40	32.53
SignBERT [23]	✓	–	39.40	36.74
HMA [24]	–	✓	37.91	35.90
BEST [86]	✓	–	46.25	43.52
SignBERT+ [26]	✓	–	48.85	46.37
MSLU [90]	✓	–	56.29	53.29
NLA-SLR [93]	✓	✓	61.05	58.05
Sign-Rep [75]	✓	✓	61.05	58.89
Our Models				
Uni-Sign (Pose)	✓	–	63.13	60.90
Uni-Sign (Pose+RGB)	✓	✓	<u>63.52</u>	<u>61.32</u>
Geo-Sign (Token)	✓	—	**63.64**	**61.89**

Table 4: Ablation studies for Geo-Sign on the CSL-Daily test set, examining (a) initial curvature c, (b) loss blending factor α, and (c) robustness of poses to Gaussian noise. We demonstrate that hyperbolic regularisation improves robustness to pertubation and poor pose estimation.

(a) Impact of Curvature c.

Curvature (c)	B-4	R-L
0.00 (Euclidean)	25.98	53.93
0.10	26.56	57.56
0.50	26.34	56.30
1.00	27.04	57.67
1.50	**27.42**	57.95
2.00	27.25	**58.08**

(b) Impact of α.

α	B-4	R-L
0.10	25.74	56.20
0.50	26.79	57.38
0.70	**27.42**	**57.95**
0.90	26.92	57.67

(c) Noise Robustness (B-4).

Noise	Geo-Sign	Uni-Sign
0.00	27.42	26.25
0.01	26.30 (-4%)	24.14 (-8%)
0.02	24.60 (-10%)	21.50 (-18%)
0.03	19.07 (-30%)	14.40 (-45%)
0.04	11.63 (-58%)	7.20 (-73%)
0.05	5.98 (-78%)	3.01 (-89%)

the loss blending factor α. For curvature c (with $\alpha = 0.7$), setting $c = 0.001$ effectively makes the projection Euclidean (as $\tanh(x) \approx x$ for small x, which means almost zero hyperbolic warping). We observe that increasing curvature from this Euclidean-like baseline ($c = 0.001$, BLEU-4 25.91%) generally improves performance, with optimal BLEU-4 (27.42%) achieved at $c = 1.5$. ROUGE-L peaks at $c = 2.0$ (58.08%), though BLEU-4 slightly dips to 27.25%, suggesting a trade-off. This indicates that a significant degree of negative curvature is beneficial for capturing sign language structure. For the loss blending factor α (with $c = 1.5$), a value of $\alpha = 0.7$ (i.e., 30% weight to the hyperbolic loss) yields the best BLEU-4 (27.42%) and ROUGE-L (57.95%). Lower or higher α values result in decreased performance, indicating that the hyperbolic regularisation provides a substantial complementary signal to the primary translation loss, but should not entirely dominate it during the 40 epochs of fine-tuning. Finally, we assess the robustness of our approach to pose perturbation and poor pose estimation. To do so, we add Gaussian noise to the pose embeddings before embedding via the ST-GCN. Our method shows how hyperbolic regularisation improves robustness to pose noise compared to the Uni-Sign Euclidean baseline. We attribute this to the larger geodesic margins between pose embeddings in the hyperbolic space, making the model less susceptible to noisy perturbation.

4.3 Qualitative Analysis: Visualizing Embedding Spaces

To intuitively understand the effect of hyperbolic regularisation, we visualise the learned pose embeddings. Figure 2 shows UMAP [47] projections of these embeddings into the 2D Poincaré disk (by log-mapping hyperbolic embeddings to the tangent space at the origin, then applying UMAP). We compare embeddings from our Geo-Sign (Hyperbolic Token) model against those from the

Geo-Sign (Euclidean Token) model, which uses the same contrastive token-level alignment but without hyperbolic projection (curvature $c = 0$).

The Euclidean embeddings (Figure 2, Left) appear relatively clustered and undifferentiated. In contrast, the hyperbolic embeddings (Figure 2, Right) exhibit a more structured distribution. Notably, embeddings corresponding to hand articulations (often carrying fine-grained lexical information) tend to occupy regions further from the origin, towards the periphery of the Poincaré disk. This is consistent with hyperbolic geometry's property of expanding space near the boundary, providing more capacity to distinguish subtle variations. Conversely, features representing larger body movements or overall posture (often conveying prosodic or grammatical information) tend to be located more centrally. This visualised structure suggests that the hyperbolic model indeed learns to place features in a manner that reflects the hierarchical nature of sign kinematics, with fine details pushed to high-curvature regions and global features remaining near the low-curvature origin.

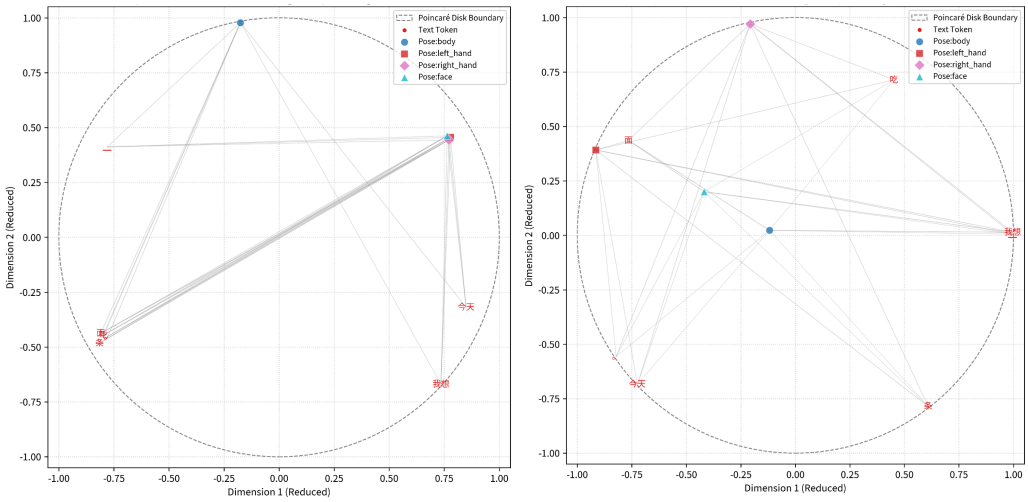


Figure 2: UMAP projection of pose part summary embeddings ($\bar{\mathbf{f}}_p$ onto the 2D Poincaré disk). (**Left**) Embeddings from the Euclidean Token regularisation model ($c = 0.001$). (**Right**) Embeddings from the Geo-Sign (Hyperbolic Token) model. The hyperbolic embeddings show a more structured distribution, with hand features (representing finer details) often pushed towards the periphery indicative of a learned kinematic hierarchy.

5 Conclusion

This paper introduced Geo-Sign, a novel framework that enhances Sign Language Translation by leveraging hyperbolic geometry to model the inherent hierarchical structure of sign language kinematics. By projecting skeletal features from ST-GCNs into the Poincaré ball and employing a geometric contrastive loss, Geo-Sign regularises a pre-trained mT5 model, guiding it to learn more discriminative and geometrically aware representations. We explored two alignment strategies: a global pooled method and a fine-grained token-based attention method operating directly in hyperbolic space. Our experimental results on both Chinese Sign Language and American Sign Language demonstrate the significant benefits of this approach.

Acknowledgements

This work was supported by the SNSF project 'SMILE II' (CRSII5 193686), the Innosuisse IICT Flagship (PFFS-21-47), EPSRC grant APP24554 (SignGPT-EP/Z535370/1) and through funding from Google.org via the AI for Global Goals scheme. This work reflects only the author's views and the funders are not responsible for any use that may be made of the information it contains.

Thank you to Low Jian He for reviewing the Chinese text translations.

References

[1] Samuel Albanie, Gül Varol, Liliane Momeni, Hannah Bull, Triantafyllos Afouras, Himel Chowdhury, Neil Fox, Bencie Woll, Rob Cooper, Andrew McParland, et al. Bbc-oxford british sign language dataset. In *arXiv preprint arXiv:2111.03635*, 2021.

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

[3] Gary Bécigneul and Octavian-Eugen Ganea. Riemannian adaptive optimization methods. In *International Conference on Learning Representations (ICLR 2019)*, 2023.

[4] Jan Bungeroth and Hermann Ney. Statistical sign language translation. In *sign-lang@ LREC 2004*, 2004.

[5] Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, Hermann Ney, and Richard Bowden. Neural sign language translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018.

[6] Necati Cihan Camgoz, Oscar Koller, Simon Hadfield, and Richard Bowden. Sign language transformers: Joint end-to-end sign language recognition and translation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020.

[7] Necati Cihan Camgöz, Ben Saunders, Guillaume Rochette, Marco Giovanelli, Giacomo Inches, Robin Nachtrab-Ribback, and Richard Bowden. Content4all open research sign language translation datasets. In *2021 16th IEEE International Conference on Automatic Face and Gesture Recognition*. IEEE, 2021.

[8] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: realtime multi-person 2d pose estimation using part affinity fields. In *TPAMI*, volume 43. IEEE, 2019.

[9] Zhigang Chen, Benjia Zhou, Yiqing Huang, Jun Wan, Yibo Hu, Hailin Shi, Yanyan Liang, Zhen Lei, and Du Zhang. C^2rl: Content and context representation learning for gloss-free sign language translation and retrieval. In *arxiv*, 2024.

[10] Zhigang Chen, Benjia Zhou, Jun Li, Jun Wan, Zhen Lei, Ning Jiang, Quan Lu, and Guoqing Zhao. Factorized learning assisted with large language model for gloss-free sign language translation. In *LREC-COLING*, 2024.

[11] Zhigang Chen, Benjia Zhou, Yiqing Huang, Jun Wan, Yibo Hu, Hailin Shi, Yanyan Liang, Zhen Lei, and Du Zhang. C 2 rl: Content and context representation learning for gloss-free sign language translation and retrieval. In *IEEE Transactions on Circuits and Systems for Video Technology*, 2025.

[12] Yiting Cheng, Fangyun Wei, Jianmin Bao, Dong Chen, and Wenqiang Zhang. CiCo: Domain-aware sign language retrieval via cross-lingual contrastive learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2023.

[13] MMPose Contributors. Openmmlab pose estimation toolbox and benchmark, 2020.

[14] Haodong Duan, Yue Zhao, Kai Chen, Dahua Lin, and Bo Dai. Revisiting skeleton-based action recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2022.

[15] Amanda Duarte, Shruti Palaskar, Lucas Ventura, Deepti Ghadiyaram, Kenneth DeHaan, Florian Metze, Jordi Torres, and Xavier Giro-i Nieto. How2sign: a large-scale multimodal dataset for continuous american sign language. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2021.

[16] Luca Franco, Paolo Mandica, Bharti Munjal, and Fabio Galasso. Hyperbolic self-paced learning for self-supervised skeleton-based action representations. In *arXiv preprint arXiv:2303.06242*, 2023.

[17] Shiwei Gan, Yafeng Yin, Zhiwei Jiang, Kang Xia, Lei Xie, and Sanglu Lu. Contrastive learning for sign language recognition and translation. In *IJCAI*, 2023.

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

[19] Jia Gong, Lin Geng Foo, Yixuan He, Hossein Rahmani, and Jun Liu. Llms are good sign language translators. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2024.

[20] Shester Gueuwou, Xiaodan Du, Greg Shakhnarovich, and Karen Livescu. SignMusketeers: An efficient multi-stream approach for sign language translation at scale. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Findings of the Association for Computational Linguistics: ACL 2025*, Vienna, Austria, 2025. Association for Computational Linguistics.

[21] Dan Guo, Wen gang Zhou, Houqiang Li, and M. Wang. Hierarchical lstm for sign language translation. In *AAAI Conference on Artificial Intelligence*, 2018.

[22] Zhengsheng Guo, Zhiwei He, Wenxiang Jiao, Xing Wang, Rui Wang, Kehai Chen, Zhaopeng Tu, Yong Xu, and Min Zhang. Unsupervised sign language translation and generation. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, Bangkok, Thailand, 2024. Association for Computational Linguistics.

[23] Hezhen Hu, Weichao Zhao, Wengang Zhou, Yuechen Wang, and Houqiang Li. SignBERT: pre-training of hand-model-aware representation for sign language recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2021.

[24] Hezhen Hu, Wengang Zhou, and Houqiang Li. Hand-model-aware sign language recognition. In *AAAI*, 2021.

[25] Hezhen Hu, Wengang Zhou, Junfu Pu, and Houqiang Li. Global-local enhancement network for nmf-aware sign language recognition. In *ACM transactions on multimedia computing, communications, and applications (TOMM)*, 2021.

[26] Hezhen Hu, Weichao Zhao, Wengang Zhou, and Houqiang Li. Signbert+: Hand-model-aware self-supervised pre-training for sign language understanding. In *IEEE TPAMI*, 2023.

[27] Lianyu Hu, Liqing Gao, Zekang Liu, and Wei Feng. Temporal lift pooling for continuous sign language recognition. In *ECCV*, 2022.

[28] Lianyu Hu, Liqing Gao, Zekang Liu, and Wei Feng. Continuous sign language recognition with correlation network. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2023.

[29] Sarah Ibrahimi, Mina Ghadimi Atigh, Nanne Van Noord, Pascal Mettes, and Marcel Worring. Intriguing properties of hyperbolic embeddings in vision-language models. In *Transactions on Machine Learning Research*, 2024.

[30] Maksym Ivashechkin, Oscar Mendez, and Richard Bowden. Improving 3d pose estimation for sign language. In *2023 IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW)*, 2023. doi: 10.1109/ICASSPW59220.2023.10193629.

[31] Maksym Ivashechkin, Oscar Mendez, and Richard Bowden. Improving 3d pose estimation for sign language. In *IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops*. IEEE, 2023.

[32] Youngjoon Jang, Haran Raajesh, Liliane Momeni, Gül Varol, and Andrew Zisserman. Lost in translation, found in context: Sign language translation with contextual cues. In *arXiv preprint arXiv:2501.09754*, 2025.

[33] Tao Jiang, Peng Lu, Li Zhang, Ning Ma, Rui Han, Chengqi Lyu, Yining Li, and Kai Chen. RTMPose: Real-time multi-person pose estimation based on mmpose. In *arxiv*, 2023.

[34] Sheng Jin, Lumin Xu, Jin Xu, Can Wang, Wentao Liu, Chen Qian, Wanli Ouyang, and Ping Luo. Whole-body human pose estimation in the wild. In *ECCV*, 2020.

[35] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[36] Max Kochurov, Rasul Karimov, and Serge Kozlukov. Geoopt: Riemannian optimization in pytorch, 2020.

[37] Oscar Koller, Jens Forster, and Hermann Ney. Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers. In *CVIU*, volume 141, 2015.

[38] Zhiying Leng, Shun-Cheng Wu, Mahdi Saleh, Antonio Montanaro, Hao Yu, Yin Wang, Nassir Navab, Xiaohui Liang, and Federico Tombari. Dynamic hyperbolic attention network for fine hand-object reconstruction. In *Proceedings of the IEEE/CVF international conference on computer Vision*, 2023.

[39] Dongxu Li, Cristian Rodriguez, Xin Yu, and Hongdong Li. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. In *WACV*, 2020.

[40] Yue Li, Haoxuan Qu, Mengyuan Liu, Jun Liu, and Yujun Cai. Hyliformer: Hyperbolic linear attention for skeleton-based human action recognition. In *arXiv preprint arXiv:2502.05869*, 2025.

[41] Zecheng Li, Wengang Zhou, Weichao Zhao, Kepeng Wu, Hezhen Hu, and Houqiang Li. Uni-sign: Toward unified sign language understanding at scale. In *arXiv preprint arXiv:2501.15187*, 2025.

[42] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, 2004.

[43] Kezhou Lin, Xiaohan Wang, Linchao Zhu, Ke Sun, Bang Zhang, and Yi Yang. Glofe: Gloss-free end-to-end sign language translation. In *ACL*, 2023.

[44] Qi Liu, Maximilian Nickel, and Douwe Kiela. Hyperbolic graph neural networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[45] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *arXiv preprint arXiv:1711.05101*, 2017.

[46] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, et al. Mediapipe: A framework for building perception pipelines. In *arXiv preprint arXiv:1906.08172*, 2019.

[47] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. Umap: Uniform manifold approximation and projection. In *booktitle of Open Source Software*, volume 3. The Open booktitle, 2018.

[48] Marko Valentin Micic and Hugo Chu. Hyperbolic deep learning for chinese natural language understanding. In *arXiv preprint arXiv:1812.10408*, 2018.

[49] Liliane Momeni, Gul Varol, Samuel Albanie, Triantafyllos Afouras, and Andrew Zisserman. Watch, read and lookup: learning to spot signs from multiple supervisors. In *ACCV*, 2020.

[50] Liliane Momeni, Hannah Bull, KR Prajwal, Samuel Albanie, Gül Varol, and Andrew Zisserman. Automatic dense annotation of large-vocabulary sign language videos. In *ECCV*, 2022.

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

[52] Maximillian Nickel and Douwe Kiela. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *International conference on machine learning*. PMLR, 2018.

[53] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. In *arXiv preprint arXiv:1807.03748*, 2018.

[54] Katerina Papadimitriou and Gerasimos Potamianos. Multimodal Sign Language Recognition via Temporal Deformable Convolutional Sequence Learning. In *Interspeech*, 2020.

[55] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 2002.

[56] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, 2019.

[57] Junfu Pu, Wengang Zhou, Hezhen Hu, and Houqiang Li. Boosting continuous sign language recognition via cross modality augmentation. In *ACMMM*, 2020.

[58] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3D residual networks. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2017.

[59] Razieh Rastgoo, Kourosh Kiani, and Sergio Escalera. Sign language recognition: A deep survey. In *ESA*, 2021.

[60] Phillip Rust, Bowen Shi, Skyler Wang, Necati Cihan Camgoz, and Jean Maillard. Towards privacy-aware sign language translation at scale. In *ACL*, 2024.

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

[62] Ozge Mercanoglu Sincan, Necati Cihan Camgoz, and Richard Bowden. Is context all you need? scaling neural sign language translation to large domains of discourse. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2023.

[63] Stephanie Stoll, Necati Cihan Camgöz, Simon Hadfield, and R. Bowden. Sign language production using neural machine translation and generative adversarial networks. In *British Machine Vision Conference*, 2018.

[64] Shengeng Tang, Dan Guo, Richang Hong, and Meng Wang. Graph-based multimodal sequential embedding for sign language translation. In *IEEE TMM*, 2021.

[65] Shengeng Tang, Richang Hong, Dan Guo, and Meng Wang. Gloss semantic-enhanced network with online back-translation for sign language production. In *ACM International Conference on Multimedia*, 2022.

[66] Garrett Tanzer and Biao Zhang. Youtube-sl-25: A large-scale, open-domain multilingual sign language parallel corpus. In *arxiv*, 2024.

[67] Laia Tarrés, Gerard I. Gállego, Amanda Duarte, Jordi Torres, and Xavier Giró i Nieto. Sign language translation from instructional videos. In *CVPRW*, 2023.

[68] Dave Uthus, Garrett Tanzer, and Manfred Georg. Youtube-asl: A large-scale, open-domain american sign language-english parallel corpus. In *Advances in Neural Information Processing Systems*, 2024.

[69] Gul Varol, Liliane Momeni, Samuel Albanie, Triantafyllos Afouras, and Andrew Zisserman. Read and attend: Temporal localisation in sign language videos. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2021.

[70] Gül Varol, Liliane Momeni, Samuel Albanie, Triantafyllos Afouras, and Andrew Zisserman. Scaling up sign spotting through sign language dictionaries. In *IJCV*, volume 130, 2022.

[71] Harry Walsh, Ozge Mercanoglu Sincan, Ben Saunders, and Richard Bowden. Gloss alignment using word embeddings. In *2023 IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW)*. IEEE, 2023.

[72] Harry Walsh, Abolfazl Ravanshad, Mariam Rahmani, and Richard Bowden. A data-driven representation for sign language production. In *Proceedings of the 18th International Conference on Automatic Face and Gesture Recognition*, 2024.

[73] Fangyun Wei and Yutong Chen. Improving continuous sign language recognition with cross-lingual signs. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2023.

[74] Ryan Wong, Necati Cihan Camgoz, and Richard Bowden. Sign2GPT: Leveraging large language models for gloss-free sign language translation. In *ICLR*, 2024.

[75] Ryan Wong, Necati Cihan Camgoz, and Richard Bowden. Signrep: Enhancing self-supervised sign representations. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2025.

[76] Qinkun Xiao, Minying Qin, and Yuting Yin. Skeleton-based chinese sign language recognition and generation for bidirectional communication between deaf and hearing people. In *Neural networks*, 2020.

[77] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *ECCV*, 2018.

[78] Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mT5: A massively multilingual pre-trained text-to-text transformer. In *NAACL*, 2021.

[79] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *AAAI*, 2018.

[80] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *AAAI*, 2018.

[81] Menglin Yang, Min Zhou, Zhihao Li, Jiahong Liu, Lujia Pan, Hui Xiong, and Irwin King. Hyperbolic graph neural networks: A review of methods and applications. In *arXiv preprint arXiv:2202.13852*, 2022.

[82] 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*, 2024.

[83] Aoxiong Yin, Tianyun Zhong, Li Tang, Weike Jin, Tao Jin, and Zhou Zhao. Gloss attention for gloss-free sign language translation. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2023.

[84] Jan Zelinka and Jakub Kanis. Neural sign language synthesis: Words are our glosses. In *WACV*, 2020.

[85] Rui Zhao, Liang Zhang, Biao Fu, Cong Hu, Jinsong Su, and Yidong Chen. Conditional variational autoencoder for sign language translation with cross-modal alignment. In *AAAI*, 2024.

[86] Weichao Zhao, Hezhen Hu, Wengang Zhou, Jiaxin Shi, and Houqiang Li. BEST: Bert pre-training for sign language recognition with coupling tokenization. In *AAAI*, 2023.

[87] Benjia Zhou, Zhigang Chen, Albert Clapés, Jun Wan, Yanyan Liang, Sergio Escalera, Zhen Lei, and Du Zhang. Gloss-free sign language translation: Improving from visual-language pretraining. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2023.

[88] Benjia Zhou, Zhigang Chen, Albert Clapés, Jun Wan, Yanyan Liang, Sergio Escalera, Zhen Lei, and Du Zhang. Gloss-free sign language translation: Improving from visual-language pretraining. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2023.

[89] Hao Zhou, Wengang Zhou, Weizhen Qi, Junfu Pu, and Houqiang Li. Improving sign language translation with monolingual data by sign back-translation. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2021.

[90] Wengang Zhou, Weichao Zhao, Hezhen Hu, Zecheng Li, and Houqiang Li. Scaling up multi-modal pre-training for sign language understanding. In *arxiv*, 2024.

[91] Ronglai Zuo and Brian Mak. Local context-aware self-attention for continuous sign language recognition. In *Proc. Interspeech*, 2022.

[92] Ronglai Zuo and Brian Mak. Improving continuous sign language recognition with consistency constraints and signer removal. In *ACM TOMM*, volume 20, 2024.

[93] Ronglai Zuo, Fangyun Wei, and Brian Mak. Natural language-assisted sign language recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2023.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Demonstrated in results and ablation experiments.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations section in the main paper and discussion in the supplementary material.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Full experiment details provided in the supplementary material and code samples provided. The dataset used is open source.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Dataset is open source, code samples are provided and full code and data is available online.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: All details provided in the supplementary material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We use the same training setup and hyperparameter setup as the code in Uni-Sign for fair comparison including random seed. Due to compute constraints we were not able to perform multiple experiments over additional random seeds.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Full details provided in the supplementary material.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: No conflicts.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Brief discussion on importance of Sign Language Translation on improving communication between hearing and Deaf communities.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

Appendix

A Introduction

In this appendix, we provide comprehensive supplementary details to accompany our main paper. The goal is to offer an in-depth understanding of our methodology, experimental setup, and the underlying geometric principles, thereby ensuring clarity and facilitating the reproducibility of our work.

This document elaborates on:

- The specifics of pose feature extraction and the Spatio-Temporal Graph Convolutional Network (ST-GCN) architecture employed (Appendix C.1).
- Detailed explanations and implementations of our proposed hyperbolic alignment strategies, including the Pooled Method and the Token Method (Appendix C.2).
- Further mathematical derivations and discussions pertinent to hyperbolic operations, such as Fréchet mean computation and contrastive loss gradients (Appendix D).
- Elaboration on the learnable parameters within our model, particularly the manifold curvature c and the loss blending factor α (Appendix E).
- A discussion of computational considerations, experimental setup, and qualitative results (Appendix F).
- Key code snippets for essential components of Geo-Sign are provided in Appendix G to aid in understanding and replication.

B Hyperbolic Geometry Preliminaries: A Brief Refresher

To ensure this supplementary material is self-contained and accessible, this section briefly recaps key concepts from hyperbolic geometry, as introduced in Section 3.1 ("Hyperbolic Geometry Essentials") of the main paper.

We operate within the d_{hyp}-dimensional Poincaré ball model, denoted $\mathbb{B}_c^{d_{\text{hyp}}} = \{\mathbf{x} \in \mathbb{R}^{d_{\text{hyp}}} : \|\mathbf{x}\|_2 < 1/\sqrt{c}\}$. This space is characterised by a constant negative curvature $\kappa = -c$, where $c > 0$ is a learnable parameter representing the magnitude of the curvature.

The Poincaré ball model is chosen for its conformal nature, where angles are preserved locally, and its intuitive representation of hyperbolic space within a Euclidean unit ball (scaled by $1/\sqrt{c}$). Key operations include:

- **Geodesic Distance** $d_{\mathbb{B}_c}(\mathbf{u}, \mathbf{v})$: This is the shortest path between two points $\mathbf{u}, \mathbf{v}$ within the curved space of the Poincaré ball. It is formally defined in Eq. (1) of the main paper. Unlike Euclidean distance, it expands significantly as points approach the boundary of the ball.
- **Möbius Addition** $\mathbf{u} \oplus_c \mathbf{v}$: This operation is the hyperbolic analogue of vector addition in Euclidean space, defined in Eq. (2) of the main paper (consistent with formulations in, e.g., [18]). It is essential for defining translations and other transformations in hyperbolic space while respecting its geometry.
- **Exponential Map** $\exp_{\mathbf{x}}^c(\mathbf{v})$: This map takes a tangent vector $\mathbf{v}$ residing in the tangent space $\mathcal{T}_{\mathbf{x}}\mathbb{B}_c^{d_{\text{hyp}}}$ at a point $\mathbf{x}$ on the manifold and maps it to another point on the manifold along a geodesic. The map from the origin, $\exp_{\mathbf{0}}^c(\cdot)$ (Eq. (3), main paper), is particularly important as it projects Euclidean feature vectors (which can be considered as residing in $\mathcal{T}_{\mathbf{0}}\mathbb{B}_c^{d_{\text{hyp}}}$) into the Poincaré ball.
- **Logarithmic Map** $\log_{\mathbf{x}}^c(\mathbf{y})$: This is the inverse of the exponential map. It takes two points $\mathbf{x}, \mathbf{y}$ on the manifold and returns the tangent vector at $\mathbf{x}$ that points along the geodesic towards $\mathbf{y}$.
- **Möbius Transformations**: These are isometries (distance-preserving transformations) of hyperbolic space. In our work, we use learnable Möbius transformations, such as Möbius matrix-vector products ($\mathbf{M} \otimes_c \mathbf{v} = \exp_{\mathbf{0}}^c(\mathbf{M} \log_{\mathbf{0}}^c(\mathbf{v}))$) and Möbius bias additions, to implement affine-like transformations within our hyperbolic attention mechanism.

These tools allow us to define neural network operations directly within hyperbolic space. As with all hyperbolic operations in the paper, we utilise the geoopt library [36] in Pytorch.

C Methodology Details

C.1 Pose Extraction and ST-GCN Architecture Details

Our Geo-Sign framework utilizes skeletal pose data as input. This section details the extraction process and the architecture of the Spatio-Temporal Graph Convolutional Networks (ST-GCNs) used to encode this data.

C.1.1 Pose Data Source and Preprocessing

We use the 2D skeletal keypoints provided by the UniSign [41] framework, which were originally extracted using RTMPose-X [33] based on the COCO-WholeBody keypoint definition [34]. The keypoints are organised into four distinct anatomical groups for targeted processing:

- **Body**: Includes 9 joints (COCO indices 1, 4–11).
- **Left Hand**: Includes 21 joints (COCO indices 92–112).
- **Right Hand**: Includes 21 joints (COCO indices 113–133).
- **Face**: Includes 16 keypoints from the facial region (COCO indices 24, 26, 28, 30, 32, 34, 36, 38, 40, 54, 84–91).

For normalization, specific anchor joints are used for hand and face parts: joint 92 (left wrist) for the left hand, joint 113 (right wrist) for the right hand, and joint 54 (a central face point) for the face. The body part features are not anchor-normalised in this scheme to preserve global torso positioning.

C.1.2 ST-GCN Architecture

Each anatomical group is processed by a dedicated ST-GCN stream, following the methodology of Yan et al. [79]. The ST-GCN is adept at learning representations from skeletal data by explicitly modeling spatial joint relationships and temporal motion dynamics.

The core of the ST-GCN involves:

1. **Graph Definition**: The skeletal structure for each part is defined as a graph, where joints are nodes and natural bone connections are edges. The `Graph` class, detailed in Listing 1 (Appendix G), handles the construction of these graphs and their corresponding adjacency matrices.
2. **Initial Projection**: Input keypoint coordinates are first linearly projected to a higher-dimensional feature space using a linear layer (referred to as `proj_linear` in our codebase).
3. **ST-GCN Blocks**: A sequence of ST-GCN blocks processes these features. Each block (see `STGCN_block` in Listing 2, Appendix G) consists of:
   - A **Spatial Graph Convolution (SGC)** layer, which aggregates information from neighboring joints. The operation for a node (joint) v_i at layer (l) can be expressed generally as:

$$\mathbf{f}_{\text{out}}(v_i)^{(l)} = \sum_{k=1}^{K} \left(\sigma \left(\mathbf{A}_k \mathbf{X}^{(l)} \mathbf{W}_k^{(l)} \right) \right)_i , \tag{9}$$

   where $\mathbf{X}^{(l)} \in \mathbb{R}^{N \times C_{in}}$ is the matrix of input features for N nodes with C_{in} channels, $\mathbf{W}_k^{(l)} \in \mathbb{R}^{C_{in} \times C_{out}}$ are learnable weight matrices for the k-th kernel transforming node features to C_{out} channels. $\mathbf{A}_k \in \mathbb{R}^{N \times N}$ is the adjacency matrix for the k-th spatial kernel, defining the neighborhood aggregation based on chosen strategies (we use the spatial configuration partitioning as in the original ST-GCN paper [79]). σ is an activation function (ReLU in our case), and $(\cdot)_i$ denotes selection of the i-th row (features for node v_i). The precise implementation involving tensor reshaping and `einsum` for efficient aggregation over multiple adjacency kernels is detailed in the `GCN_unit` code in Listing 2.

- A **Temporal Convolutional Network (TCN)** layer, which applies 1D convolutions across the time dimension to model motion patterns.

4. **Residual Connections**: To allow richer feature interaction, residual connections are introduced from the body stream's ST-GCN output to the hand and face streams before their final temporal fusion layers. This allows global body posture context to inform the interpretation of fine-grained hand and face movements. Details are in Listing 3 (Appendix G). This design choice treats body features as fixed contextual input for the parts during each forward pass, isolating the body feature extractor from direct updates via part-specific losses.

The output of each part-specific ST-GCN stream is a feature map $\mathbf{Z}_p \in \mathbb{R}^{T \times d'_{\text{gcn\_out}}}$, where T is the sequence length and $d'_{\text{gcn\_out}}$ is the GCN output feature dimension. For the hyperbolic regularization branch, these $\mathbf{Z}_p$ are temporally mean-pooled to produce static summary vectors $\bar{\mathbf{f}}_p \in \mathbb{R}^{d'_{\text{gcn\_out}}}$, which encapsulate the overall kinematics of part p for subsequent hyperbolic projection.

C.2 Hyperbolic Alignment Strategies: Detailed Implementation

This section provides a more detailed explanation of the two hyperbolic alignment strategies introduced in Section 3.3 of the main paper. These strategies are designed to regularize the mT5 model by aligning pose and text representations within the Poincaré ball.

C.2.1 Pooled Method (Global Semantic Alignment)

This strategy aims to align the holistic semantic content of the sign language video (represented by pose features) with the corresponding text translation.

1. Part-Specific Hyperbolic Embeddings: The temporally mean-pooled Euclidean feature vectors $\bar{\mathbf{f}}_p$ for each anatomical part p (body, hands, face) are projected into the Poincaré ball $\mathbb{B}_c^{d_{\text{hyp}}}$. This projection, yielding hyperbolic embeddings $\mathbf{h}_p$, is achieved using the `HyperbolicProjection` layer (Listing 4 in Appendix G), as defined in Eq. (4) of the main paper:

$$\mathbf{h}_p = \exp_{\mathbf{0}}^c(s_p \mathbf{W}^p \bar{\mathbf{f}}_p). \tag{10}$$

Here, $\mathbf{W}^p$ represents a linear layer for part p, and s_p is a learnable scalar that adaptively scales the tangent space representation before the exponential map $\exp_{\mathbf{0}}^c(\cdot)$ projects it onto the manifold.

2. Weighted Fréchet Mean for Global Pose Representation: The set of part-specific hyperbolic embeddings $\{\mathbf{h}_p\}$ is aggregated into a single global pose representation $\boldsymbol{\mu}_{\text{pose}} \in \mathbb{B}_c^{d_{\text{hyp}}}$. This is achieved by computing their weighted Fréchet mean, which is the hyperbolic analogue of a weighted average. The Fréchet mean is defined as the point that minimizes the sum of squared weighted geodesic distances to all input points:

$$\boldsymbol{\mu}_{\text{pose}} = \underset{\boldsymbol{\mu} \in \mathbb{B}_c^{d_{\text{hyp}}}}{\operatorname{argmin}} \sum_{p=1}^{P} w_p d^2_{\mathbb{B}_c}(\boldsymbol{\mu}, \mathbf{h}_p). \tag{11}$$

The weights w_p are designed to give more importance to parts whose embeddings are further from the origin of the Poincaré ball (i.e., parts with more "hyperbolic energy" or distinctness), normalised via softmax:

$$w_p = \frac{\exp(d_{\mathbb{B}_c}(\mathbf{0}, \mathbf{h}_p)/\lambda_w)}{\sum_{j=1}^{P} \exp(d_{\mathbb{B}_c}(\mathbf{0}, \mathbf{h}_j)/\lambda_w)}. \tag{12}$$

Here, λ_w is a temperature parameter for the softmax (e.g., fixed to 1.0 in our experiments) controlling the sharpness of the weight distribution. The computation is performed iteratively as detailed in Algorithm 1 of the main paper and Listing 5 (Appendix G).

3. Global Text Representation: Similarly, a global hyperbolic text embedding $\mathbf{h}_{\text{text}} \in \mathbb{B}_c^{d_{\text{hyp}}}$ is derived from the mT5 model's output. Euclidean token embeddings from the final layer of the mT5 decoder are first mean-pooled (respecting padding masks) to obtain a single sentence-level vector $\bar{\mathbf{e}}_{\text{text}}$. This vector is then projected into $\mathbb{B}_c^{d_{\text{hyp}}}$ using a dedicated hyperbolic projection layer (structurally identical to Eq. (10)):

$$\mathbf{h}_{\text{text}} = \exp_{\mathbf{0}}^c(s_{\text{text}} \mathbf{W}^{\text{text}} \bar{\mathbf{e}}_{\text{text}}). \tag{13}$$

The implementation details are shown in Listing 6 (Appendix G).

4. Contrastive Alignment: Finally, the geometric contrastive loss (Eq. (5) in the main paper) is applied between batches of these global pose embeddings $\{\boldsymbol{\mu}_{\text{pose},i}\}$ and global text embeddings $\{\mathbf{h}_{\text{text},i}\}$. This encourages semantically similar pose-text pairs to be closer in hyperbolic space.

C.2.2 Token Method (Fine-Grained Part-Text Alignment)

This strategy facilitates a more detailed alignment by relating individual pose part embeddings $\{\mathbf{h}_p\}$ with contextually relevant text segment embeddings $\{\mathbf{c}_p\}$.

1. Hyperbolic Pose Part Embeddings $\{\mathbf{h}_p\}$: These are obtained exactly as in the Pooled Method, using Eq. (10). Each $\mathbf{h}_p$ represents a specific anatomical part's overall kinematic signature.

2. Hyperbolic Text Token Embeddings: Instead of a global text embedding, each Euclidean text token embedding $\mathbf{e}_{\text{token},j}$ (from the mT5 decoder's final layer) is individually projected into the Poincaré ball $\mathbb{B}_c^{d_{\text{hyp}}}$:

$$\mathbf{h}_{\text{token},j} = \exp_{\mathbf{0}}^{c}(s_{\text{text}}\mathbf{W}^{\text{text}}\mathbf{e}_{\text{token},j}). \tag{14}$$

This results in a sequence of hyperbolic token embeddings $\{\mathbf{h}_{\text{token},j}\}_{j=1}^{L_t}$, where L_t is the text sequence length.

3. Hyperbolic Attention Mechanism: For each hyperbolic pose part embedding $\mathbf{h}_p$ (acting as a query), a contextual text embedding $\mathbf{c}_p$ is generated. This is achieved using a hyperbolic attention mechanism (see Listing 7 in Appendix G) that operates as follows:

- **Key Transformation**: The hyperbolic text token embeddings $\{\mathbf{h}_{\text{token},j}\}$ serve as keys. The embeddings are first transformed using learnable Möbius transformations to enhance their representational capacity:

  $$\mathbf{k}_j = (\mathbf{M}_{\text{key}} \otimes_c \mathbf{h}_{\text{token},j}) \oplus_c \mathbf{b}_{\text{key}},$$

  where $\mathbf{M}_{\text{key}}$ is a learnable Möbius matrix and $\mathbf{b}_{\text{key}}$ is a learnable Möbius bias vector.
- **Attention Scores**: Attention scores are computed based on the negative geodesic distance between each pose query $\mathbf{h}_p$ and each transformed text key $\mathbf{k}_j$:

  $$\text{score}_{pj} = -d_{\mathbb{B}_c}(\mathbf{h}_p, \mathbf{k}_j).$$

- **Attention Weights**: These scores are normalised using a softmax function (after applying padding masks) to obtain attention weights α_{pj}:

  $$\alpha_{pj} = \text{softmax}\left(\frac{\text{score}_{pj}}{\tau_{\text{attn}}}\right),$$

  where τ_{attn} is a learnable temperature parameter for the attention mechanism, distinct from the temperature in the contrastive loss.
- **Contextual Text Embedding** $\mathbf{c}_p$: The contextual text embedding $\mathbf{c}_p$ corresponding to pose part $\mathbf{h}_p$ is then computed as the hyperbolic weighted midpoint of the original hyperbolic text token embeddings $\{\mathbf{h}_{\text{token},j}\}$, using the attention weights $\{\alpha_{pj}\}$.

4. Contrastive Alignment: The geometric contrastive loss (Eq. (5), main paper) is then applied for each pair $(\mathbf{h}_{p,i}, \mathbf{c}_{p,i})$ across the batch. The total regularization loss for this strategy is the average of these individual contrastive losses over all parts P.

C.2.3 Intuition Behind the Token Method

While the Pooled Method aligns the overall semantics of a sign sequence with its translation, it may not capture how specific signing elements (e.g., a handshape, movement, or facial expression) correspond to particular words or phrases. The Token Method aims to establish this more fine-grained understanding.

The core intuition is as follows:

1. **Compositional Language Understanding**: Sign languages, like spoken/written languages, are compositional. Different articulators (hands, body, face) convey distinct lexical or

grammatical information. The Token Method attempts to map these compositional units from pose to corresponding textual tokens (words/sub-words).

2. **Targeted Part-to-Segment Alignment**: Instead of a single global comparison, this method learns to connect individual pose part representations (e.g., features for the dominant hand, to the most relevant segments of the textual translation.
3. **Pose Parts as Queries, Text Tokens as Sources**: Each hyperbolic pose part embedding $\mathbf{h}_p$ acts as a "query", effectively asking: "Which text tokens are most semantically relevant to this pose feature?" The sequence of hyperbolic text token embeddings $\{\mathbf{h}_{\text{token},j}\}$ serves as the "information source" for these queries.
4. **Hyperbolic Attention for Geometric Relevance**:
   - Relevance between a pose part query $\mathbf{h}_p$ and a (transformed) text token key $\mathbf{k}_j$ is measured by their geodesic distance $d_{\mathbb{B}_c}(\mathbf{h}_p, \mathbf{k}_j)$ in the learned hyperbolic space. A smaller distance implies higher relevance. Using hyperbolic geometry allows these comparisons to potentially leverage latent hierarchical relationships between concepts.
   - Learnable Möbius transformations on text tokens (to get keys $\mathbf{k}_j$) enable the model to learn distinct tokens relevant to different pose parts (e.g., a verb token might be transformed to be closer to a body movement embedding).
   - The attention weights α_{pj} then quantify the contribution of each text token j to the meaning conveyed by pose part p.
5. **Learning Textual Context for Each Pose Part**: The contextual text embedding $\mathbf{c}_p$ is a hyperbolic weighted midpoint of all text token embeddings, using the attention weights α_{pj}. Thus, $\mathbf{c}_p$ is a summary of the sentence, but specifically customised by the interaction of pose part p.
6. **Refined Contrastive Learning**: The model is regularised to make each pose part embedding $\mathbf{h}_p$ close to its corresponding contextual text view $\mathbf{c}_p$ in hyperbolic space, while pushing it away from non-corresponding pairs.
7. **Overall Benefit**: This detailed, part-specific alignment encourages the mT5 model to learn more precise mappings between kinematic features of different articulators and semantic units within the text. For example, it can help distinguish visually similar signs based on subtle hand details (encoded in $\mathbf{h}_{\text{hand}}$) that correlate with specific words, leading to more accurate and nuanced translations.

D Mathematical Foundations

This section recalls two geometric components that Geo-Sign relies on:

- the *Weighted Fréchet Mean* inside the Poincaré ball (used in Algorithm 1 of the paper);
- the Euclidean gradient of the hyperbolic distance that appears in the contrastive loss.

D.1 Fréchet Mean in the Poincaré Ball

Given points $x_1, \dots, x_N$ in a metric space $(\mathcal{M}, d)$ with normalised weights $w_i > 0$, $\sum_i w_i = 1$, the **Fréchet mean** minimises

$$\mathcal{F}(\mu) = \sum_{i=1}^{N} w_i \, d^2(\mu, x_i), \qquad \mu^\star = \arg\min_{\mu \in \mathcal{M}} \mathcal{F}(\mu).$$

Why not simply average the embeddings in Euclidean space? Two issues appear inside the curved Poincaré ball:

(a) **Manifold constraint.** A Euclidean average of interior points can fall *outside* the ball, i.e. outside valid hyperbolic space, forcing an ad-hoc projection that distorts geometry.

(b) **Metric distortion.** Euclidean distance underestimates separation near the boundary because the hyperbolic metric stretches space there. A straight average therefore over-emphasises central points and washes out fine structure carried by peripheral ones.

The intrinsic Fréchet mean lives on the manifold and uses the true hyperbolic distance, so it respects curvature.

Why distance-based weights? Each pose part (body, face, left hand, right hand) yields a hyperbolic embedding h_p. We set $w_p \propto \exp\big(d_{\mathbb{B}_c}(0, h_p)/\lambda_w\big)$ so parts farther from the origin, in regions of higher curvature and greater discriminative power, receive more influence. Without this weighting the mean would drift toward the centre, diluting information contributed by the hands and face.

Iterative update. On any Riemannian manifold the mean is found by Riemannian gradient descent; the update at iteration k is

$$\mu^{(k+1)} = \exp_{\mu^{(k)}}\Big(\eta_k \sum_{i=1}^{N} w_i \, \log_{\mu^{(k)}}(x_i)\Big), \tag{15}$$

with step size $\eta_k > 0$.

Proposition D.1 (Convergence in $\mathbb{B}_c^d$)**.** *The Poincaré ball $\mathbb{B}_c^d$ is a Hadamard manifold, hence $\mathcal{F}$ is strictly convex and has a* unique *minimiser $\mu^\star$. Let L be the Lipschitz constant of $\nabla\mathcal{F}$ on the geodesic convex hull of $\{x_i\}$. If $0 < \eta_k \le 2/L$ for all k, the iterates (15) converge to $\mu^\star$. In practice we observe $L \le 2$, so the simple choice $\eta_k = 1$ is usually sufficient and used in our approach.*

D.2 Gradient of the Hyperbolic Distance

For $u, v \in \mathbb{B}_c^d$ let $w = (-u) \oplus_c v$ (the Möbius difference, i.e. the "vector" from u to v transported to the origin). The Poincaré distance is

$$d_{\mathbb{B}_c}(u, v) = \frac{2}{\sqrt{c}} \operatorname{artanh}\big(\sqrt{c}\,\|w\|_2\big).$$

Differentiating [18, 51] gives the Euclidean gradient required for autograd:

$$\nabla_u \, d_{\mathbb{B}_c}(u, v) = -\frac{2}{\lambda_u^c \, \lambda_v^c} \, \frac{w}{\|w\|_2} \, \frac{1}{1 - c\|w\|_2^2} \tag{16}$$

with conformal factor $\lambda_x^c = \dfrac{2}{1 - c\|x\|_2^2}$. The same formula (with sign reversed) holds for ∇_v.

The update rule (15) and the gradient (16) provide all the geometric tools needed by Geo-Sign's hyperbolic contrastive regulariser.

E Learnable Model Parameters: c and α

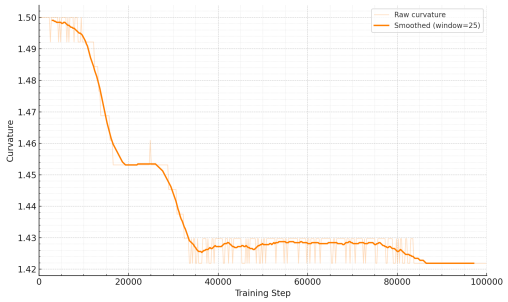
Our Geo-Sign model incorporates several learnable parameters beyond standard network weights. This section details two key ones: the manifold curvature c and the loss blending factor α.

E.1 Discussion on Learnable Curvature

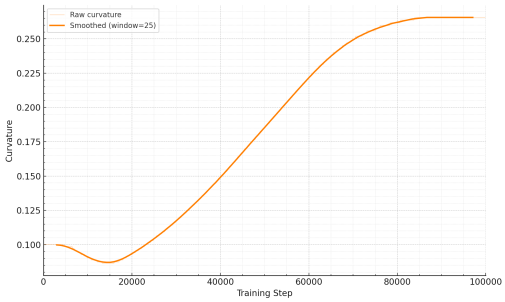
The curvature of the Poincaré ball, $\kappa = -c$ (where $c > 0$), is a crucial hyperparameter that dictates the "shape" of the hyperbolic space. Instead of fixing c heuristically, we make it a learnable parameter of our model (see Listing 9 in Appendix G).

Optimization Strategy: The curvature magnitude c is initialised (e.g., via `args.init_c` as mentioned in the main paper's experiments) and then updated via standard gradient descent as part of the end-to-end training process. The `geoopt` library facilitates this by defining c as an `nn.Parameter` within its `PoincareBall` manifold object when `learnable=True`.

The main paper's ablation studies (Table 2a) show that initializing c in the range of $1.0 - 2.0$ (e.g., optimal BLEU-4 at $c = 1.5$) yields strong performance. Figure 3 illustrates how c adapts during training from different initializations.



(a) c initialised at 1.50, decreases and stabilizes around 1.42.



(b) c initialised at 0.10, increases and stabilizes around 0.20.

Figure 3: Evolution of the learnable manifold curvature c during training for different initializations. (a) When initialised at $c = 1.50$, the curvature magnitude slightly decreases, suggesting an optimal value around 1.42 for this setup. (b) When initialised at a low $c = 0.10$, the curvature increases, indicating the model benefits from more "hyperbolic space" initially. It stabilizes around $c = 0.20$, potentially influenced by the dynamic α schedule that reduces regularization emphasis over time.

E.2 Discussion on Loss Blending Factor α

The total training loss $\mathcal{L}_{\text{total}}$ is a weighted combination of the primary cross-entropy translation loss $\mathcal{L}_{\text{CE}}$ and our hyperbolic contrastive regularization term $\mathcal{L}_{\text{hyp\_reg}}$:

$$\mathcal{L}_{\text{total}} = \alpha \cdot \mathcal{L}_{\text{CE}} + (1 - \alpha) \cdot \mathcal{L}_{\text{hyp\_reg}}.$$

The blending factor α is not fixed but is dynamically adjusted during training. This dynamic scheduling allows the model to potentially benefit from different loss emphases at different training stages. The calculation of α at each training step (see Listing 10 in Appendix G) is:

$$\alpha_{\text{final}} = \text{clamp}\left((\alpha_{\text{init}} + 0.1 \cdot \text{progress}) + \sigma(\text{logit}_{\alpha}) \cdot 0.2,\ 0.1,\ 1.0\right), \tag{17}$$

where:

- α_{init} is the initial value for the blending factor, specified as a hyperparameter (e.g., `args.alpha = 0.7` from the main paper's ablations, Table 2b, which was found to be optimal).
- progress is the current training progress, calculated as $\frac{\text{current\_training\_step}}{\text{total\_training\_steps}}$, ranging from 0 to 1. This component introduces a linear ramp, potentially increasing α's baseline by up to 0.1 over the course of training.
- logit_{α} is an `nn.Parameter` (a learnable scalar, referred to as `self.loss_alpha_logit` in the code). $\sigma(\cdot)$ is the sigmoid function, so $\sigma(\text{logit}_{\alpha})$ maps this learnable scalar to the range $(0, 1)$. This term provides a learnable adjustment to α in the range of $[0, 0.2]$.
- $\text{clamp}(\cdot, 0.1, 1.0)$ ensures that the final α_{final} remains within the bounds $[0.1, 1.0]$.

This dynamic α allows for an initial phase where the hyperbolic regularization might have more relative influence (if α_{init} is smaller), gradually shifting emphasis or allowing the model to fine-tune the balance via the learnable component. The ablation study in the main paper (Table 2b) indicates that an initial $\alpha_{\text{init}} = 0.7$ (i.e., 30% weight to $\mathcal{L}_{\text{hyp\_reg}}$ initially) provides the best results, highlighting the complementary role of the hyperbolic regularization.

F Experimental Setup, Analysis, and Qualitative Results

F.1 Computational Profile

This section discusses the computational profile of Geo-Sign, comparing it to a baseline Uni-Sign (Pose) model without hyperbolic regularization. The analysis is based on DeepSpeed profiler outputs for models run with a batch size of 8 on the CSL-Daily dataset for the Sign Language Translation (SLT) task.

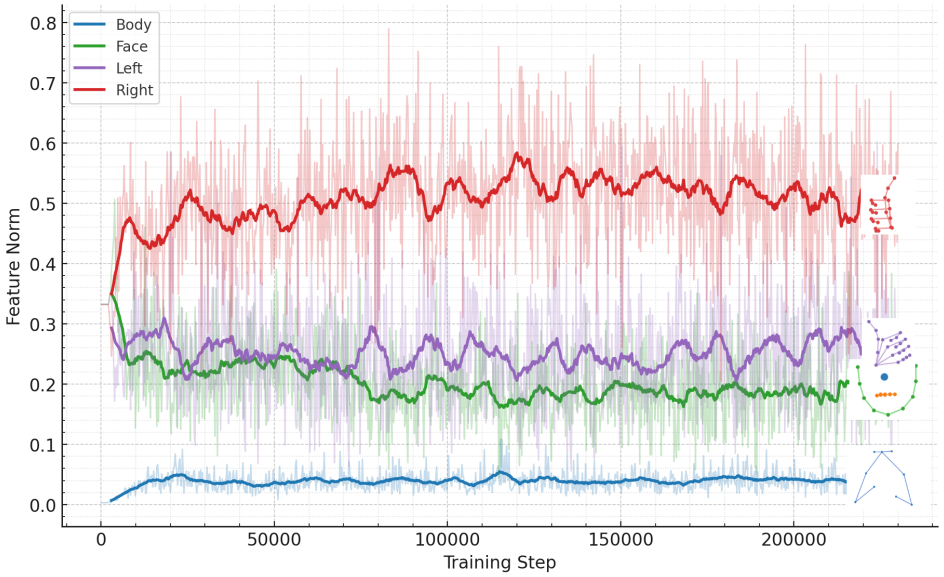


Figure 4: Plot of the geodesic distances from the origin ($\mathbf{0}$) of the Poincaré disk to the hyperbolic pose embeddings ($\mathbf{h}_p$) during training, averaged per part type. This shows how features for different parts utilize the hyperbolic space. For instance, right hand features (often conveying detailed lexical information) tend to move further from the origin, leveraging more of the hyperbolic curvature for discriminability. Body and face features, which might represent broader semantics or prosody, may remain closer to the Euclidean-like central region.

Experimental Context: Key experimental conditions for fine-tuning include:

- **Hardware**: 4 NVIDIA RTX 3090 GPUs.
- **Training Time**: Approximately 10 hours for 40 epochs of fine-tuning on CSL-Daily.
- **Precision**: Mixed-precision training (`bfloat16`) is used for standard PyTorch layers, while `float32` is maintained for Geoopt hyperbolic operations to ensure numerical stability.
- **Batching Strategy**: With an effective batch size of 8 per GPU, the model occupies ≈20GB of memory. During training, we increase the total batch size to 32 and accumulate gradients over 8 steps, achieving a hypothetical batch size of 256. For the following profiler analysis, we report results for a single GPU with a batch size of 8 to provide a clear per-device profile.

F.1.1 Profiler Summary and Comparative Analysis

Table 5 summarizes key metrics from the profiler. Parameter counts are consistent with the main paper's Table 1, while MACs (Multiply-Accumulate operations) and Latency are derived from DeepSpeed profiler outputs for a batch size of 8. Table 6 provides a comparison of model parameters against other gloss-free methods.

Table 5: Computational profile comparison at Batch Size 8: Baseline Uni-Sign (Pose) vs. Geo-Sign variants. Parameter counts from main paper's Table 1. MACs and Latency from DeepSpeed profiler outputs. "Hyperbolic Proj. Layer MACs" reflects profiled contributions from the learnable linear transformations within these layers.

Model Variant (Batch Size 8)	Total Params (M)	Added Params (M)	Total Fwd MACs (GMACs)	Hyperbolic Proj. Layer MACs (MMACs)	Fwd Latency (ms)	Latency Increase (%)
Baseline Uni-Sign (Pose)	587.75	-	116.59	-	415.73	-
Geo-Sign (Hyperbolic Pooled)	588.21	0.46	116.60	3.67	1630.00	292.10
Geo-Sign (Hyperbolic Token)	589.10	1.35	116.60	≈9.96	2550.00	513.40

Parameter Overhead: The increase in parameters due to the hyperbolic components is marginal compared to the overall model size, which is dominated by the mT5 language model (≈ 582.4M parameters).

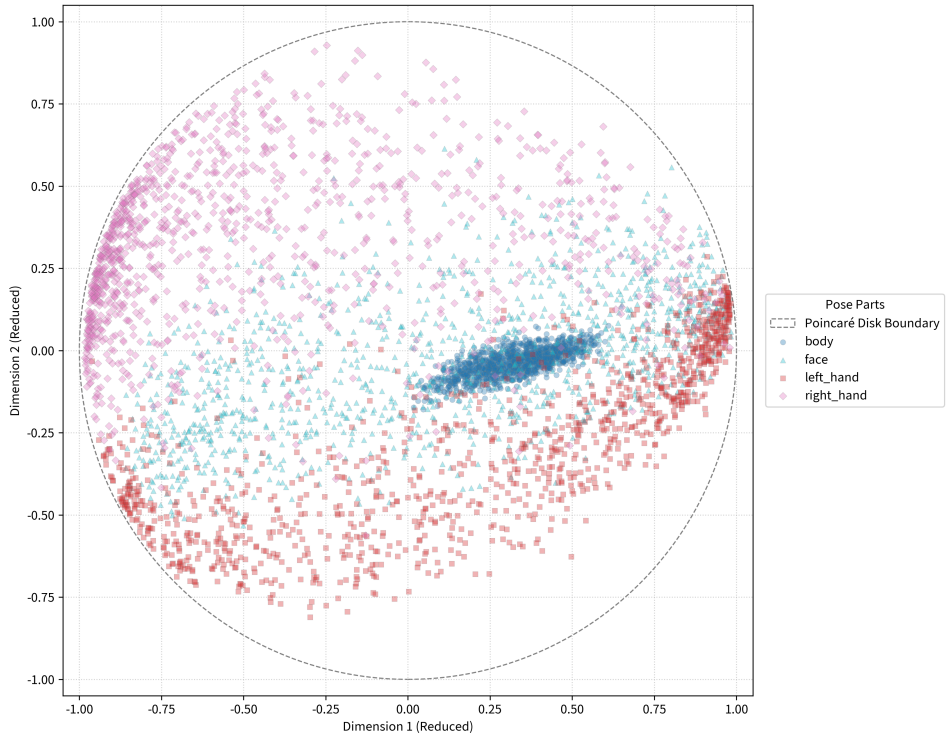


Figure 5: PCA projection of 1000 hyperbolic pose part embeddings (log-mapped to the tangent space at origin, then PCA-reduced to 2D) visualised within the Poincaré disk. Body features (blue) are tightly clustered near the origin, suggesting their discriminability is well-handled in a more Euclidean-like region. Hand features (left: red square, right: pink diamond) and face features (light blue triangle) are more dispersed, with hand features often pushed towards the periphery. This indicates these parts benefit from the increased representational capacity near the boundary of the Poincaré disk, where hyperbolic geometry provides more "space" to distinguish subtle variations crucial for sign language semantics.

- Baseline Uni-Sign (Pose): ≈ 587.75M parameters.
- **Geo-Sign (Pooled)**: Adds ≈ 0.46M parameters, primarily from the five hyperbolic projection layers (one for each of the four pose parts and one for the pooled text embedding).
- **Geo-Sign (Token)**: Adds ≈ 1.35M parameters. This includes the ≈ 0.46M for projection layers plus an additional ≈ 0.89M for the learnable parameters within the hyperbolic attention mechanism (Möbius matrices and biases).

In both Geo-Sign variants, the parameter overhead from hyperbolic components is less than 0.25% of the total model size. As shown in Table 6, our Geo-Sign models achieve competitive or superior performance to recent RGB-based methods while maintaining a significantly smaller total parameter count. This highlights the efficiency of enhancing skeletal representations with geometric priors, challenging the trend that relies solely on scaling up visual encoders and language model decoders for performance gains in SLT.

MACs Analysis: The DeepSpeed profiler indicates that the total forward MACs are very similar across all configurations at this batch size:

- Baseline Uni-Sign (Pose): ≈ 116.59 GMACs.
- Geo-Sign (Hyperbolic Pooled): ≈ 116.60 GMACs. The profiler attributes ≈ 3.67 MMACs to the linear transformations within its `HyperbolicProjection` layers.
- Geo-Sign (Hyperbolic Token): ≈ 116.60 GMACs. Its `HyperbolicProjection` layers account for ≈ 9.96 MMACs from their linear components.

Table 6: Sign Language Translation performance (Test Set: BLEU-4, ROUGE-L) and model parameters on CSL-Daily. Scores are percentages (%). Higher is better. 'Pose' and 'RGB' indicate input modalities. VE/LM/Total Params are in Millions (M). Approx. values indicated by ≈. Data from CSL-Daily (Train: 18,401 sentences / 20.62 hours).

Method	VE Name	VE Params (M)	LM Name	LM Params (M)	Total Params (M)	Modality		Test Set	
						Pose	RGB	B-4	R-L
Gloss-Free Methods (Prior Art)									
MSLU [90]	EffNet	5.3	mT5-Base	582.4	587.7	✓	–	11.42	33.80
SLRT [6] (G-Free)	EffNet	5.3	Transformer	≈30	≈35.3	–	✓	3.03	19.67
GASLT [83]	I3D	13	Transformer	≈30	≈43.0	–	✓	4.07	20.35
GFSLT-VLP [88]	ResNet18	11.7	mBart	680	691.7	–	✓	11.00	36.44
FLa-LLM [10]	ResNet18	11.7	mBart	680	691.7	–	✓	14.20	37.25
Sign2GPT [74]	DinoV2	21.0	XGLM	1732.9	1753.9	–	✓	15.40	42.36
SignLLM [19]	ResNet18	11.7	LLaMA-7B	6738.4	6750.1	–	✓	15.75	39.91
C^2RL [9]	ResNet18	11.7	mBart	680	691.7	–	✓	21.61	48.21
Our Models and Baselines									
Uni-Sign [41] (Pose)	GCN	5.3	mT5-Base	582.4	587.7	✓	–	25.61	54.92
Uni-Sign [41] (Pose+RGB)	EffNet+GCN	9.7	mT5-Base	582.4	592.1	✓	✓	26.36	56.51
Geo-Sign (Hyperbolic Pooled)	GCN+Geo	5.8	mT5-Base	582.4	588.21	✓	–	27.17	57.75
Geo-Sign (Hyperbolic Token)	GCN+Geo+Attn	6.7	mT5-Base	582.4	589.1	✓	–	**27.42**	**57.95**

The MACs from the learnable linear transformations within the hyperbolic projection layers constitute a very small fraction ($< 0.01\%$) of the total model MACs. The bulk of MACs originates from the mT5 model (profiled at ≈ 66.29 GMACs) and the ST-GCN modules (profiled at ≈ 49.93 GMACs). We should note, however, that standard profilers (like DeepSpeed's MAC counter) primarily quantify MACs from common operations like convolutions and linear layers. The computational cost of specialised geometric functions within `geoopt` (e.g., `manifold.dist`, `expmap0`, `logmap0`, Möbius arithmetic) is not explicitly broken out as distinct hyperbolic operation MACs. These functions often involve sequences of elementary operations that are not all MAC-based (e.g., square roots, divisions, trigonometric functions like `artanh` or `tanh`). Thus, their computational load may be underestimated by MAC counters and is often better reflected in measured latency.

Latency Analysis: Latency figures clearly reveal the primary computational overhead introduced by the hyperbolic components during training:

- Baseline Uni-Sign (Pose): ≈ 416 ms forward latency per batch.
- Geo-Sign (Hyperbolic Pooled): ≈ 1630 ms (1.63 s), an increase of ≈ 1214 ms or $\approx 292\%$ over the baseline (approx. $3.9\times$ slowdown).
- Geo-Sign (Hyperbolic Token): ≈ 2550 ms (2.55 s), an increase of ≈ 2134 ms or $\approx 513\%$ over the baseline (approx. $6.1\times$ slowdown).

The substantial increase in training latency, despite modest increases in parameters and profiled MACs from learnable layers, underscores that the geometric operations themselves are the main performance consideration during the training phase. These operations (e.g., geodesic distance, exponential/logarithmic maps, Möbius transformations) are inherently more complex than their Euclidean counterparts. The Token method is notably slower than the Pooled method during training due to its per-token hyperbolic attention.

Importantly, a key advantage of our regularization approach is that these geometric operations and the hyperbolic branch are **not utilised at inference time**. Consequently, Geo-Sign models incur no additional latency increase over the baseline Uni-Sign (Pose) model during inference, preserving efficiency for deployment.

F.1.2 Discussion on Data Efficiency

While not directly evaluated, it is hypothesised that skeletal data's abstraction from visual noise (lighting, background, clothing) can enhance robustness and generalization [75], especially when training data is limited. Hyperbolic geometry further imposes a structural prior on the representation space. This inductive bias could potentially improve data efficiency by guiding the learning process, particularly in scenarios with sparse data, although specific experiments to quantify this effect were not part of the current study. One trade-off of this approach is that we cannot directly leverage

large pre-trained visual encoders as in the case of other RGB approaches, and so pre-training on a sign-specific dataset like CSL-News (1,985 hours, used by Uni-Sign) is essential. However, this pre-training data size is comparable to that used by other SLT methods which use datasets such as How2Sign [15] (2000 hours) or YouTube-ASL [66, 68] (6000 hours). We anticipate that our method would continue to scale well with larger pre-training datasets in other sign languages, though resource constraints prevented evaluation of this aspect.

F.2 Further Technical Implementation Details

This section provides additional details that are pertinent for a full understanding and potential reimplementation of Geo-Sign.

- **Core Libraries**: Our implementation relies on PyTorch [56] as the primary deep learning framework. For Transformer models, we utilize the HuggingFace Transformers library. All hyperbolic geometry operations and Riemannian optimization are handled by the Geoopt library [36]. For distributed training and profiling, DeepSpeed is employed.
- **Hyperparameter Tuning Strategy**: Key hyperparameters specific to the hyperbolic components, such as the initial curvature c, the initial loss blending factor α_{init} (referred to as `args.alpha` in code/main paper), and the hyperbolic embedding dimension d_{hyp}, were tuned using a grid search strategy on the CSL-Daily development set. Full hyperparameters are outlined in Table 7.
- **Numerical Stability Measures**:
  - Operations within `geoopt` are performed using `float32` precision to maintain numerical stability, while the rest of the model uses mixed precision.
  - Small epsilon values (e.g., 10^{-5}) are added in denominators and inside logarithms/arctanh functions where appropriate to prevent division by zeros.
  - **Tangent Vector Clipping**: Before applying an exponential map $\exp_{\mathbf{x}}^c(\mathbf{v})$ from a point $\mathbf{x}$ with a tangent vector $\mathbf{v}$, especially $\exp_{\mathbf{0}}^c(\mathbf{v})$, it's crucial to ensure the resulting point remains strictly within the Poincaré ball and that the norm of $\mathbf{v}$ doesn't cause numerical issues in $\tanh(\cdot)$. We apply a clipping strategy as mentioned in Section 3.4 of the main paper:

    $$\mathbf{v}_{\text{clipped}} \leftarrow \frac{\mathbf{v}}{\max(1, \sqrt{c}\|\mathbf{v}\|_2 + \epsilon_{\text{clip}})},$$

    for a small $\epsilon_{\text{clip}} > 0$ (e.g., 10^{-5}). This ensures that the argument to $\tanh$ in $\exp_{\mathbf{0}}^c$ does not become excessively large and that mapped points do not reach or exceed the boundary of the Poincaré ball. The `project=True` flag in `geoopt`'s `expmap` functions also helps enforce this by projecting points back onto the ball if they numerically fall outside.
- **Gradient Clipping**: Standard norm-based gradient clipping is applied to all model parameters during training to stabilize the optimization process.

In Table 7 we provide the full hyper-parameters for the best performing model. The full code will be released following the review process.

F.3 Limitations and Future Work

While offering representational benefits, hyperbolic operations can add computational overhead compared to purely Euclidean ones though this is generally offset by avoiding raw video processing. The optimal choice of hyperbolic model parameters (e.g., curvature strategy) warrants further study. Generalizability to a wider range of sign languages also needs investigation. Promising directions include exploring other hyperbolic models (e.g., Lorentz), developing more sophisticated dynamic curvature adaptation, integrating Geo-Sign's hyperbolic skeletal features into multi-modal frameworks, and applying these geometric principles to other sign language processing tasks like recognition or generation. Further research into the interpretability of learned hyperbolic embeddings could also yield deeper insights into how sign language structure is captured.

Table 7: Hyperparameter summary for Geo-Sign experiments. Values are for the best reported model configuration.

Category	Hyperparameter	Value	Description
General Training Configuration			
	Random Seed	42	Seed for reproducibility
	Training Epochs	40	Number of fine-tuning epochs on CSL-Daily
	Batch Size (per GPU)	8	Micro-batch size per GPU
	Gradient Accumulation Steps	8	Effective batch size becomes $8 \times$ accum_steps $\times$ num_gpus
	Training Precision (`dtype`)	`bf16`	Mixed precision training data type
Data Handling			
	Max Pose Sequence Length	256	Maximum number of frames for pose sequences
	Max Target Text Length (`max_tgt_len`)	100	Max new tokens for generation during evaluation
Optimizer (Euclidean: ST-GCN, mT5, Linear Layers)			
	Optimizer Type (`opt`)	AdamW	[45]
	Learning Rate (`lr`)	3×10^{-5}	For Euclidean parameters (AdamW)
	AdamW β_1, β_2 (`opt-betas`)	[0.9, 0.999]	Exponential decay rates for moment estimates
	AdamW ϵ (`opt-eps`)	1×10^{-8}	Term for numerical stability
	Weight Decay (`weight-decay`)	0.01	L2 penalty for Euclidean parameters
	LR Scheduler (`sched`)	Cosine Annealing	
	Warmup Epochs (`warmup-epochs`)	5	Number of epochs for LR warm-up
	Minimum LR (`min-lr`)	1×10^{-6}	Lower bound for LR in scheduler
	Gradient Clipping Norm	1.0	Max norm for gradients
Optimizer (Hyperbolic: Manifold Parameters, Projections)			
	Optimizer Type	RAdam	Riemannian Adam
	Learning Rate (`hyp_lr`)	1×10^{-3}	For hyperbolic parameters (RAdam)
Model Architecture			
	ST-GCN Output Dimension (`gcn_out_dim`)	256	Output dimension of ST-GCN part streams
	mT5 Projection Dimension (`hidden_dim`)	768	Target dimension for projecting GCN features to match mT5
Hyperbolic Regularization			
	Hyperbolic Embedding Dimension (d_{hyp}, `hyp_dim`)	256	Dimension of embeddings in Poincaré ball
	Initial Curvature (c_{init}, `init_c`)	1.5	Initial value for learnable curvature c (for best model)
	Loss Blend α_{init} (`alpha`)	0.70	Initial blending factor for $\mathcal{L}_{\text{CE}}$ vs $\mathcal{L}_{\text{hyp\_reg}}$ (for best model)
	Text Comparison Mode (`hyp_text_cmp`)	`token`	Strategy for aligning pose with text tokens (Token Method)
	Hyperbolic Contrastive Loss $\mathcal{L}_{\text{hyp\_reg}}$:		
	Temperature (τ)	Learnable	Temperature for scaling distances in contrastive loss
	Margin (m)	Learnable	Additive margin for negative pairs in contrastive loss
	Label Smoothing (`label_smoothing_hyp`)	0.2	Label smoothing for hyperbolic contrastive loss (InfoNCE)
Loss Functions			
	CE Loss Label Smoothing (`label_smoothing`)	0.2	Label smoothing for mT5 cross-entropy loss
Distributed Training (DeepSpeed)			
	ZeRO Optimization Stage (`zero_stage`)	2	DeepSpeed ZeRO Stage for memory efficiency
	Offload to CPU (`offload`)	False	Whether to offload optimizer/params to CPU

F.4 Qualitative Results

Additional Figures: Figure 4 (similar to aspects shown in Figure 2 of the main paper, concerning learned embedding distributions) illustrates the dynamic utilization of the hyperbolic manifold by showing the average geodesic distance of different pose part embeddings from the origin during training. Notably, features corresponding to hand articulations, which often carry fine-grained lexical information, tend to migrate towards the periphery of the Poincaré disk. This suggests that the model leverages the increased representational capacity in high-curvature regions to distinguish subtle hand-based signs.

Furthermore, Figure 5 (again, related to Figure 2 of the main paper, specifically the UMAP projections) provides a PCA-reduced visualization of the learned hyperbolic pose part embeddings projected onto the 2D Poincaré disk for 1000 poses. This plot reveals a structured distribution where body features cluster near the origin (a more Euclidean-like region suitable for broader semantics), while hand and face features are more dispersed, with hand features populating regions further towards the boundary. This geometric organization, reflecting a learned kinematic hierarchy, likely contributes to the improved discriminability and, consequently, the enhanced translation quality demonstrated in the following examples. These visualizations support the hypothesis that the geometric biases induced by hyperbolic space aid in forming more effective representations for sign language translation.

Translation Results: In this section, we provide an overview of translation samples generated by Geo-Sign . All predictions are from our best-performing "Token" model. First, in Table 8, we show examples of prediction errors with analysis and a general measure of semantic similarity (introduced for readability, not a quantitative metric). English translations are automatically generated and then verified by a native Chinese speaker. We observe that translation quality with respect to semantics is generally high, though our method, like many SLT systems, can sometimes miss pronouns or struggle with complex tenses. In Table 9, we showcase examples where our approach generates

perfect or near-perfect translations. Finally, in Table 10, we select some examples to compare our model's output with that of the Uni-Sign (Pose) baseline. These comparisons illustrate improvements in semantic meaning and accuracy, consistent with the quantitative gains in ROUGE and BLEU-4 scores reported in the main paper.

Table 8: Examples of Prediction Errors and Analysis from Geo-Sign (Token Method)

Prediction	**Ground Truth**	**Analysis of Error**	**Semantic Similarity**
她 今 年 5 0 岁 。 (She is 50 years old.)	他 今 年 四 岁 。 (He is 4 years old.)	Pronoun error: 她 (she) vs. 他 (he). Number error: "5 0" (50) vs. 四 (four). The prediction gets the topic (age) but is wrong on subject and specific age.	Partial (topic: age)
今 天 星 期 五 。 (Today is Friday.)	今 天 星 期 几 ? (What day of the week is it today?)	Statement vs. Question: Prediction states a specific day. GT asks for the day. Character error: 五 (five) vs. 几 (how many/which).	Partial (topic: day of week)
你 什 么 时 候 认 识 小 张 ? (When did you meet Xiao Zhang?)	你 和 小 张 什 么 时 候 认 识 的 ? (When did you AND Xiao Zhang meet?)	Missing words: Prediction lacks "和" (and) and the particle "的". This subtly changes the meaning from a one-way recognition to a mutual acquaintance.	High
我 要 去 超 市 买 椅 子 。 (I want to go to the supermarket to buy a chair.)	我 要 去 超 市 买 椅 子 , 你 去 吗 ? (I want to go to the supermarket to buy a chair, are you going?)	Missing clause/question: Prediction omits the follow-up question "你 去 吗 ?" (are you going?).	High (core statement identical)
下 午 你 们 要 去 做 什 么 ? (What are you [plural] going to do in the afternoon?)	他 们 下 午 要 做 什 么 ? (What are they going to do in the afternoon?)	Pronoun error: 你 们 (you plural) vs. 他 们 (they).	High
下 午 你 们 需 要 做 什 么 ? (What do you [plural] need to do this afternoon?)	他 们 下 午 要 做 什 么 ? (What are they going to do this afternoon?)	Pronoun error: 你 们 (you plural) vs. 他 们 (they). Word choice: 需 要 (need) vs. 要 (going to/want to) - subtle semantic shift, GT is more natural for general plans.	High
大 家 觉 得 什 么 时 候 去 买 椅 子 ? (When does everyone think we should go buy chairs?)	他 们 想 什 么 时 候 去 买 椅 子 ? (When do they want to go buy chairs?)	Subject error: 大 家 (everyone) vs. 他 们 (they). Verb choice: 觉 得 (feel/think) vs. 想 (want/think).	High
我 手 表 不 见 了 。 (My watch is missing.)	这 块 手 表 是 你 的 吗 ? (Is this watch yours?)	Different intent: Prediction states a loss. GT asks about ownership of a present watch. Both are about watches but different scenarios.	Medium (topic: watch)
你 手 表 多 少 钱 ? (How much is your watch?)	这 块 手 表 多 少 钱 买 的 ? (How much did you buy this watch for?)	Missing context/words: Prediction is a bit abrupt. GT is more complete with "这 块" (this) and "买 的" (bought for).	High
我 发 现 了 他 的 偶 像 。 (I discovered his idol.)	你 看 见 我 的 杯 子 吗 ? (Did you see my cup?)	Completely different semantic intent and topic. Prediction is about an idol, GT is about a missing cup.	Very Low

Continued on next page

Table 8 – continued from previous page

Prediction	Ground Truth	Analysis of Error	Semantic Similarity
爸爸的房间里大了。(It has become big in dad's room / Dad's room has become bigger.)	左边的房间是我爸爸妈妈的,他们的房间很大。(The room on the left is my parents', their room is very big.)	Garbled/incomplete prediction: The prediction is grammatically awkward and misses the entire context of the GT.	Low
公司离家远,他为什么打车去公司?(The company is far from home, why does he take a taxi to the company?)	公司离家很远,她为什么不打车?(The company is very far from home, why doesn't she take a taxi?)	Pronoun error: 他 (he) vs. 她 (she). Logic error: Prediction asks why he does take a taxi, GT asks why she doesn't.	Medium
阴天说什么话?天气什么的,明天有事。(What to say on a cloudy day? Weather something, have things to do tomorrow.)	阴天,电视上说多云,怎么了?明天有事?(Cloudy day, TV says it's overcast, what's up? Got plans tomorrow?)	Nonsensical/Garbled prediction: Prediction is very disjointed and doesn't make sense, while GT is a coherent conversation about weather and plans.	Low
桌子上有饮料,你想喝什么?(There are drinks on the table, what do you want to drink?)	桌上放着很多饮料,你喝什么?(There are many drinks on the table, what do you want to drink?)	Slight phrasing difference: "桌子上有" (On the table there are) vs. "桌上放着很多" (On the table are placed many). GT is slightly more natural. Prediction is still good.	High
我刚才在家里找了一个桌子,不是找了。(I just looked for a table at home, not looked for.)	你去房间找找,是不是刚才放在桌子上了?(Go look in the room, was it just placed on the table?)	Different speaker and intent: Prediction is a confused statement about searching. GT is a directive and question to someone else.	Low
一个人的癌症会变得很可能。(A person's cancer will become very possible.)	人体的许多器官都可能发生癌变。(Many organs of the human body can become cancerous.)	Vague and unnatural prediction: "变得很可能" is awkward. GT is precise about "organs" and "癌变" (cancerous change).	Medium
老年人通过斑马线时可以走斑马线,而不走汽车。(When elderly people cross the crosswalk, they can use the crosswalk, and not walk cars.)	一位老人正在慢慢地穿过斑马线,等待的司机却不耐烦地按起了喇叭。(An old man was slowly crossing the crosswalk, but the waiting driver impatiently honked the horn.)	Nonsensical and irrelevant prediction: "而不走汽车" (and not walk cars) makes no sense. The GT describes a specific scenario.	Very Low

Table 9: Examples of Correct Predictions by Geo-Sign (Token Method)

Reference (Ground Truth)	Our Model Prediction (Perfect Match)
‘今天我想吃面条。‘ (Today I want to eat noodles.)	‘今天我想吃面条。‘ (Today I want to eat noodles.)
‘苹果是你买的吗¿ (Did you buy the apples?)	‘苹果是你买的吗¿ (Did you buy the apples?)

Continued on next page

Table 9 – continued from previous page

Reference (Ground Truth)	Our Model Prediction (Perfect Match)
‘我昨天有点累。‘ (I was a bit tired yesterday.)	‘我昨天有点累。‘ (I was a bit tired yesterday.)
‘吃完午饭要多吃点水果。‘ (Eat more fruit after lunch.)	‘吃完午饭要多吃点水果。‘ (Eat more fruit after lunch.)
‘我的妻子感冒了,我开车带她去医院。‘ (My wife has a cold, I will drive her to the hospital.)	‘我的妻子感冒了,我开车去医院。‘ (My wife has a cold, I will drive her to the hospital.)
‘我们会通过短信的方式来联系你。‘ (We will contact you via text message.)	‘我们会通过短信的方式来联系你。‘ (We will contact you via text message.)
‘我们将采用抽查的方式来进行检查。‘ (We will use random checks for inspection.)	‘我们将采用抽查的方式来进行检查。‘ (We will use random checks for inspection.)
‘你要把握好自己人生的方向。‘ (You need to grasp the direction of your own life.)	‘你要把握好自己人生的方向。‘ (You need to grasp the direction of your own life.)
‘病历是禁止涂抹、修改的。‘ (Medical records are not allowed to be smeared or altered.)	‘病历是禁止涂抹、修改的。‘ (Medical records are not allowed to be smeared or altered.)
‘他抛下家人,带着家中财物逃走了。‘ (He abandoned his family and fled with the family’s belongings.)	‘他抛下家人,带着家中财物逃走了。‘ (He abandoned his family and fled with the family’s belongings.)
‘这间玻璃作坊有一百年历史了。‘ (This glass workshop has a hundred years of history.)	‘这间玻璃作坊有一百年历史了。‘ (This glass workshop has a hundred years of history.)

Table 10: Comparative Analysis: Geo-Sign (Token) vs. Uni-Sign (Pose) - Selected Examples

Reference (Ground Truth)	Geo-Sign (Token) Prediction	Uni-Sign (Pose) Prediction
‘他每天回来都很累。‘ (He is very tired every day when he comes back.)	‘他每天来很累。‘ (He comes very tired every day.)	‘他每天来得及很累。‘ (He has enough time [to be/and is] very tired every day.)
‘小张,那个女生是你们公司的吗?你对她了解吗¿ (Xiao Zhang, is that girl from your company? Do you know her?)	‘小张那个女生是你公司的吗¿ (Xiao Zhang, is that girl from your company?)	‘那个小张是这家公司负责人,你了解吗¿ (That Xiao Zhang is the person in charge of this company, do you understand/know?)
‘阴天,电视上说多云,怎么了?明天有事¿ (Cloudy day, TV says it’s overcast, what’s up? Got plans tomorrow?)	‘阴天说什么话?天气什么的,明天有事。‘ (What to say on a cloudy day? Weather something, have things to do tomorrow.)	‘阴阳怪气地讲着天赋,不知不觉就发生了什么。‘ (Sarcastically talking about talent, something happened unknowingly.)
‘但是你一点也没瘦,你做什么运动了¿ (But you haven’t lost any weight, what exercise have you been doing?)	‘但是你没有太吃饱,你去做什么运动¿ (But you didn’t eat too full, what exercise are you going to do?)	‘但是你已经吃不消了,你能做什么呢¿ (But you already can’t stand it, what can you do?)

Continued on next page

Table 10 – continued from previous page

Reference (Ground Truth)	Geo-Sign (Token) Prediction	Uni-Sign (Pose) Prediction
‘手机选不到靠近窗户的位置。‘ (The mobile phone cannot select a seat near the window.)	‘靠近窗户的位置是不能选择的。‘ (Seats near the window cannot be selected.)	‘坐在靠近窗户的位置是不能取手机的。‘ (Sitting in a seat near the window, you cannot take out your mobile phone.)
‘他对自己一直高标准严要求。‘ (He has always had high standards and strict requirements for himself.)	‘他对自己有着严格的标准要求。‘ (He has strict standard requirements for himself.)	‘他对自己最严格的标准提出了更高的要求。‘ (He put forward higher requirements for his strictest standards.)
‘这位厨师制作的甜品,全部受欢迎。‘ (The desserts made by this chef are all popular.)	‘厨师的作品很受欢迎。‘ (The chef’s work is very popular.)	‘厨师在设作品时非常受欢迎。‘ (The chef is very popular when designing works.)

G Code Listings

These code examples provide an overview of key components in the architecture to help improve readability of the paper.

1. Defines the skeleton topology and a row-normalised adjacency tensor `A`.

```python
def hop_distance(num_nodes, edges, max_hop=1):
    """Shortest path length (<= max_hop) for every pair of nodes."""
    adj = np.zeros((num_nodes, num_nodes))
    for i, j in edges:
        adj[i, j] = adj[j, i] = 1
    hop = np.full_like(adj, np.inf, dtype=float)
    for d in range(max_hop + 1):
        hop[np.linalg.matrix_power(adj, d) > 0] = d
    return hop

class Graph:
    def __init__(self, layout='hand', strategy='uniform', max_hop=1):
        self._init_edges(layout)
        self.hop = hop_distance(self.num_nodes, self.edges, max_hop)
        self.A   = self._adjacency(strategy)

    # --- edge lists ---------------------------------------------------
    def _init_edges(self, layout):
        if layout in ('left', 'right'):                  # hand (21
            joints)
            self.num_nodes = 21
            fingers = [[0,1,2,3,4],[0,5,6,7,8],[0,9,10,11,12],
                       [0,13,14,15,16],[0,17,18,19,20]]
            links   = [(i, i) for i in range(21)]
            links  += [(f[i], f[i+1]) for f in fingers for i in range(
                len(f)-1)]
            self.edges, self.center = links, 0
        elif layout == 'body':                           # torso +
            arms
            self.num_nodes = 9
            torso = [(0,i) for i in range(1,5)]
            arms  = [(3,5),(5,7),(4,6),(6,8)]
            self.edges, self.center = [(i,i) for i in range(9)] +
                torso + arms, 0
        elif layout == 'face_all':
            self.num_nodes = 16
            ring  = [(i,(i+1)%16) for i in range(16)]
            self.edges, self.center = [(i,i) for i in range(16)] +
                ring, 8
        else:
            raise ValueError(f'Unknown layout: {layout}')

    # --- adjacency ----------------------------------------------------
    def _adjacency(self, strategy):
        A = (self.hop <= 1).astype(float)                # neighbours
            1-hop
        if strategy == 'uniform':
            A = A / (A.sum(1, keepdims=True) + 1e-6)
        elif strategy == 'distance':                     # 1 / hop
            distance
            A = 1 / (self.hop + 1e-6);  A[A == np.inf] = 0
            A = A / (A.sum(1, keepdims=True) + 1e-6)
        return torch.tensor(A, dtype=torch.float32).unsqueeze(0)
```

2. ST-GCN definition. `GCNUnit` applies K spatial kernels; `STGCNBlock` adds a temporal conv and an optional residual path.

```python
class GCNUnit(nn.Module):
    def __init__(self, Cin, Cout, A, stride=1, K=None, adaptive=True):
        super().__init__()
        self.K = K or A.shape[0]                    # #adjacency kernels
        self.A = nn.Parameter(A.clone()) if adaptive else A
        self.conv = nn.Conv2d(Cin, Cout*self.K, (1,1))
        self.bn   = nn.BatchNorm2d(Cout)
        self.act  = nn.ReLU(inplace=True)

    def forward(self, x):                           # x: (N,Cin,T,V)
        N, _, T, V = x.shape
        x = self.conv(x).view(N, self.K, -1, T, V)
        x = torch.einsum('nkctv,kvw->nctw', x, self.A)    # spatial agg
        return self.act(self.bn(x))

class STGCNBlock(nn.Module):
    def __init__(self, Cin, Cout, A, t_kernel=3, stride=1, residual=
        True):
        super().__init__()
        self.gcn = GCNUnit(Cin, Cout, A)
        pad = (t_kernel-1)//2
        self.tcn = nn.Sequential(
            nn.Conv2d(Cout, Cout, (t_kernel,1), (stride,1), (pad,0)),
            nn.BatchNorm2d(Cout))
        self.res = (nn.Identity() if Cin==Cout and stride==1
                    else nn.Conv2d(Cin, Cout, 1, (stride,1)))
        self.act = nn.ReLU(inplace=True)

    def forward(self, x):
        return self.act(self.tcn(self.gcn(x)) + self.res(x))
```

3. Body–to–part residual: body features broadcast to hands / face.models.py – residual context

```python
body_ctx = None
for part in ('body','left','right','face_all'):
    x = self.proj_linear[part](src_input[part]).permute(0,3,1,2)
    x = self.gcn_spatial[part](x)
    if part == 'body':
        body_ctx = x.detach()                       # freeze context
    else:
        joint = body_ctx[..., idx_map[part]]        # select joint
        x = x + joint.unsqueeze(-1)                 # broadcast to V
    out[part] = self.gcn_temporal[part](x)
```

4. Project Euclidean vector to the Poincaré ball.

```python
class HyperbolicProjection(nn.Module):
    def __init__(self, d_in, d_out, manifold):
        super().__init__()
        self.manifold = manifold
        self.proj = nn.Linear(d_in, d_out)
        self.log_scale = nn.Parameter(torch.zeros(()))    # ln(scale)

    def forward(self, x):
        t = self.proj(x) * self.log_scale.exp()           # tangent vec
        return self.manifold.expmap0(t, project=True)
```

5. Weighted Frechet mean (Algorithm 1).

```python
def frechet_mean(pts, w, M, max_iter=50, tol=1e-5, eta=1.0):
    """pts: (N,B,D), w: (N,B) or (N,)  ->  mu: (B,D)."""
```

```
    w = w.unsqueeze(-1) / (w.sum(0, keepdim=True) + 1e-8)
    mu = pts[0].clone()
    for _ in range(max_iter):
        v  = (w * M.logmap(mu.unsqueeze(0), pts)).sum(0)
        mu_next = M.expmap(mu, eta*v, project=True)
        if (M.dist(mu_next, mu) < tol).all(): break
        mu = mu_next
    return mu
```

6. Sentence-level text embedding (pooled method).

```
mask = txt_mask.unsqueeze(-1).float()                  # (B,T,1)
sent = (emb * mask).sum(1) / mask.sum(1).clamp_min(1)  # mean-pool
h_text = self.hyp_proj_text(sent)
```

7. Hyperbolic attention (token method).

```
h_tok = self.hyp_proj_text(tok_emb)                    # (B,T,D)
q     = h_pose.unsqueeze(2)                            # (B,P,1,D)
k     = self.manifold.mobius_add(
          self.manifold.mobius_matvec(W_key, h_tok.unsqueeze(1)),
              b_key)
logits = -self.manifold.dist(q, k)                     # (B,P,T)
logits.masked_fill_(~tok_mask.unsqueeze(1), -1e9)
alpha  = F.softmax(logits / tau_attn, -1)              # weights
ctx    = self.manifold.weighted_midpoint(h_tok.unsqueeze(1), alpha,
    [2])
```

8. InfoNCE loss calculated in hyperbolic space.

```
class HyperbolicContrastiveLoss(nn.Module):
    def __init__(self, M, tau0=0.5, m0=0.1):
        super().__init__()
        self.M = M
        self.log_tau = nn.Parameter(torch.logit(torch.tensor(tau0/2)))
        self.m       = nn.Parameter(torch.tensor(m0))

    def forward(self, a, b):                           # (B,D) pairs
        d = self.M.dist(a.unsqueeze(1), b.unsqueeze(0))
        s = -d / (torch.sigmoid(self.log_tau)*2 + 0.01)
        s -= (~torch.eye(len(a), dtype=torch.bool, device=a.device)) *
             self.m.clamp_min(0)
        target = torch.arange(len(a), device=a.device)
        return F.cross_entropy(s, target)
```

9. Manifold with learnable curvature initialisation.

```
self.manifold = geoopt.PoincareBall(c=cfg.init_c, learnable=True)
```

10. Dynamic alpha for loss blending.

```
progress = self.global_step / max(1, self.total_steps)
alpha_base  = cfg.alpha_init + 0.05 * progress            # <= 0.9
alpha_learn = 0.2 * torch.sigmoid(self.alpha_logit)
alpha       = (alpha_base + alpha_learn).clamp(0.1, 0.99)
loss        = alpha * ce_loss + (1 - alpha) * hyp_loss
```