International Journal of Computer Vision
https://doi.org/10.1007/s11263-023-01834-6

Poincaré Kernels for Hyperbolic Representations

Pengfei Fang[1,2] · **Mehrtash Harandi**[3] · **Zhenzhong Lan**[4] · **Lars Petersson**[5]

Received: 13 February 2022 / Accepted: 6 June 2023

Abstract
Embedding data in hyperbolic spaces has proven beneficial for many advanced machine learning applications. However, working in hyperbolic spaces is not without difficulties as a result of its curved geometry (*e.g.*, computing the Fréchet mean of a set of points requires an iterative algorithm). In Euclidean spaces, one can resort to kernel machines that not only enjoy rich theoretical properties but that can also lead to superior representational power (*e.g.*, infinite-width neural networks). In this paper, we introduce valid kernel functions for hyperbolic representations. This brings in two major advantages, 1. kernelization will pave the way to seamlessly benefit the representational power from kernel machines in conjunction with hyperbolic embeddings, and 2. the rich structure of the Hilbert spaces associated with kernel machines enables us to simplify various operations involving hyperbolic data. That said, identifying valid kernel functions on curved spaces is not straightforward and is indeed considered an open problem in the learning community. Our work addresses this gap and develops several positive definite kernels in hyperbolic spaces (modeled by a Poincaré ball), the proposed kernels include the rich universal ones (*e.g.*, Poincaré RBF kernel), or realize the multiple kernel learning scheme (*e.g.*, Poincaré radial kernel). We comprehensively study the proposed kernels on a variety of challenging tasks including *few-shot learning*, *zero-shot learning*, *person re-identification*, *deep metric learning*, *knowledge distillation* and *self-supervised learning*. The consistent performance gain over different tasks shows the benefits of the kernelization for hyperbolic representations.

Keywords Hyperbolic spaces · Kernelization · Poincaré kernels · Few-shot learning · Zero-shot learning · Person re-identification · Deep metric learning · Knowledge distillation · Self-supervised learning

1 Introduction

This paper studies kernel methods for hyperbolic representations. Specifically, we propose a family of positive definite (pd) kernel functions to map the embeddings in hyperbolic spaces, to be specific Poincaré ball, into Reproducing Kernel Hilbert Spaces (RKHSs), which enables us to seamlessly benefit from kernel machines to analyze hyperbolic spaces.

In the machine learning community, the Euclidean space has been the "workhorse" for feature embeddings of input data (*e.g.*, image or text). This is mainly because the high-dimensional vector space is a natural generalization from the familiar three-dimensional space we live in and performing basic operations for comparison (*e.g.*, calculating distances and similarities) is easy and straightforward. However, embedding in Euclidean spaces can harm and distort the structured data, thereby losing the complex geometric information inherently present in the data. For example, the Euclidean space fails to encode the hierarchical information in graph-structured data (Liu et al., 2019).

✉ Pengfei Fang
fangpengfei@seu.edu.cn ; Pengfei.Fang@anu.edu.au

Mehrtash Harandi
mehrtash.harandi@monash.edu

Zhenzhong Lan
lanzhenzhong@westlake.edu.cn

Lars Petersson
Lars.Petersson@data61.csiro.au

1 School of Computer Science and Engineering, Southeast University, Nanjing 210096, People's Republic of China

2 MOE Key Laboratory of Computer Network and Information Integration (Southeast University), Nanjing 210096, People's Republic of China

3 Department of Electrical and Computer Systems Engineering, Monash University, Melbourne, VIC 3800, Australia

4 School of Engineering, Westlake Univerisity, Hangzhou 310013, People's Republic of China

5 Data61-CSIRO, Canberra, ACT 2601, Australia

Published online: 26 June 2023

Several recent studies in many advanced machine learning applications (such as natural language processing, computer vision, or graph learning) suggest that embedding the data using hyperbolic geometry can be beneficial as compared to the common practice of using Euclidean geometry. This includes tasks such as textual entailment (Ganea et al., 2018), machine translation (Gulcehre et al., 2019), language-visual reasoning (Gulcehre et al., 2019), image classification and retrieval (Khrulkov et al., 2020), 3-D shape recognition (Chen et al., 2020), graph classification (Liu et al., 2019) and recommender systems (Tran et al., 2020), to name a few.

The hyperbolic space is characterized by a constant negative sectional curvature (in contrast to the flat structure of the Euclidean space), and does not satisfy Euclid's parallel postulate. One intriguing property of hyperbolic spaces is their capacity of encoding hierarchical data, as the volume of hyperbolic space expands exponentially (Hamann, 2011), thereby increasing their representation power.[1] Although several studies have successfully employed the hyperbolic geometry for inference (Ganea et al., 2018; Khrulkov et al., 2020; Cho et al., 2019), the difficulties of working with such non-linear spaces still overwhelm their wider use. For example, while averaging in Euclidean geometry is straightforward, its counterpart in hyperbolic space is approximated by the Fréchet mean. Computing the Fréchet mean requires an iterative algorithm and could easily become costly (Karcher, 1977; Lou et al., 2020). This motivates us to develop kernels to make it possible to seamlessly benefit and employ kernel machines towards analyzing hyperbolic data.

To make use of kernel machines, one needs to have a pd kernel function at its disposal. Loosely speaking, a kernel function is a measure of similarity. Many familiar kernels in the Euclidean space are defined as functions of the Euclidean distance (which is indeed the geodesic distance of the space). Take the RBF kernel $k(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\xi d^2(\boldsymbol{x}, \boldsymbol{y}))$ as an example. This might imply that valid pd kernels in curved spaces, the hyperbolic space being one, can be constructed once the geodesic distance is known. Unfortunately, this is not the case as shown in Jayasumana et al. (2015); Feragen et al. (2015) (*c.f.*, theorem 6.2 in Jayasumana et al. (2015)), because such curved spaces are not isometric to flat Euclidean spaces. Interestingly, the difficulty of defining pd kernels on curved spaces is now considered an open problem in machine learning (Feragen & Hauberg, 2016).

In our preliminary study (Fang et al., 2021a), we address the design challenge of pd kernels for hyperbolic representations. To be specific, we leverage the Poincaré ball to model the hyperbolic space and propose four valid pd Poincaré kernels, including the simple linear-like kernel as well as the universal ones. The pd properties of the proposed kernels are also proved mathematically. To evaluate the power of the proposed kernels, we also conduct experiments on several vision tasks and employ the kernels along deep neural networks (DNNs) to attain rich models for inference. Empirically, we observe the kernelization for hyperbolic representations brings performance gain considerably over the baseline model.

Despite the significant improvement from the proposed single kernels in our preliminary study (Fang et al., 2021a), tuning such kernel functions may become cumbersome in practice. In another word, it is not sufficient for an intelligent system, endowed with such a kernel, to deal with a variety of learning tasks. For example, the appropriate kernel varies for different datasets. Furthermore, tuning the kernel parameter (*e.g.*, the bandwidth for RBF kernel) is indeed effort-consuming and requires a lot of domain knowledge of tasks (Wang et al., 2021). A possible method to mitigate the issues is the *multiple kernel learning* (MKL) scheme, which learns the combination of base kernels from data (Rakotomamonjy et al., 2008; Wang et al., 2021). MKL is flexible and efficient as it automates kernel learning, such that the form of the learned kernel can fit well for the task at hand, without extensively selecting appropriate kernels and tuning the kernel parameter. This inspires us to benefit from the MKL scheme, which designs a general formulation, constituting multiple weighted kernels, in the Poincaré ball. To the best of our knowledge, this is the very first attempt where the MKL scheme is implemented for hyperbolic representations. Table 1 shows the formulation of the proposed kernels, including the one with the MKL scheme.

This manuscript extends our preliminary study in several aspects. *First*, on top of the kernels proposed in (Fang et al., 2021a), we investigate a new kernel, namely, *Poincaré radial kernel*, in conjunction with its theoretical analysis. It has been widely recognized in the community that kernel selection and tuning are not easy. While the MKL scheme effectively addresses the issues via automating to learn the kernel formulation from data, without tuning the kernel. The proposed Poincaré radial kernel can form the MKL scheme and adaptively learn the combination weights for base kernels. Empirically, the Poincaré radial kernel attains overall better performance than other kernels. *Second*, two more challenging machine learning tasks (*i.e.*, *deep metric learning* and *self-supervised learning*) are adopted to evaluate Poincaré kernels. In doing so, we kernelize the triplet loss and contrastive loss for deep metric learning and self-supervised learning respectively. To the authors' best knowledge, the kernelization of triple loss and contrastive loss has not been studied in the existing literature. *Finally*, in our initial version of this work, we investigate a good practice of hyperbolic geometry along with DNNs, as a side contribution (mapping the embedding into Poincaré directly, instead of mapping from the tangent space in existing works (Khrulkov et al.,

[1] In practice, such hierarchical structure can be revealed by the geodesic distance of two points.

Table 1 Summary of the proposed positive definite kernels in hyperbolic spaces and their properties. "MKL" indicates the multiple kernel learning scheme

Kernel	Formulation: $k(z_i, z_j)$	Condition	Properties	MKL
$f_{\mathbb{D}}(z) = \tanh^{-1}(\sqrt{c}\|z\|)\frac{z}{\sqrt{c}\|z\|}$, $c > 0$ and $z \in \mathbb{D}_c^n$				
Poincaré tangent kernel	$k^{\tan}(z_i, z_j) = \langle f_{\mathbb{D}}(z_i), f_{\mathbb{D}}(z_j)\rangle$	–	pd	✗
Poincaré RBF kernel	$k^{\mathrm{rbf}}(z_-, z_j) = \exp\left(-\xi\|f_{\mathbb{D}}(z_i) - f_{\mathbb{D}}(z_j)\|^2\right)$	$\xi > 0$	pd, universal	✗
Poincaré Laplace kernel	$k^{\mathrm{lap}}(z_i, z_j) = \exp\left(-\xi\|f_{\mathbb{D}}(z_i) - f_{\mathbb{D}}(z_j)\|\right)$	$\xi > 0$	pd, universal	✗
Generalized Poincaré Laplace kernel	$k^{\mathrm{glap}}(z_i, z_j) = \exp\left(-\xi\|f_{\mathbb{D}}(z_i) - f_{\mathbb{D}}(z_j)\|^{2\alpha}\right)$	$\xi > 0, 0 < \alpha < 1$	pd, universal	✗
Poincaré binomial kernel	$k^{\mathrm{bin}}(z_i, z_j) = \left(1 - \langle f_{\mathbb{D}}(z_i), f_{\mathbb{D}}(z_j)\rangle\right)^{-\alpha}$	$\alpha > 0$	pd, universal	✗
$g_{\mathbb{D}}(z) = \frac{f_{\mathbb{D}}(z)}{\|f_{\mathbb{D}}(z)\|}$, $f_{\mathbb{D}}(z) = \tanh^{-1}(\sqrt{c}\|z\|)\frac{z}{\sqrt{c}\|z\|}$, $c > 0$ and $z \in \mathbb{D}_c^n$				
Poincaré Radial Kernel	$k^{\mathrm{rad}}(z_i, z_j) = \sum_{m=0}^{\infty} a_m \langle g_{\mathbb{D}}(z_i), g_{\mathbb{D}}(z_j)\rangle^m$ $+a_{-1}\left([\![k_{-1}(z_i, z_j) = 1]\!] - [\![k_{-1}(z_i, z_j) = -1]\!]\right)$ $+a_{-2}[\![k_{-2}(z_i, z_j) \in \{-1, 1\}]\!]$	$a_m \geq 0$, $\sum_{m=-2}^{\infty} a_m < \infty$	pd, universal	✔

2020; Chen et al., 2020)). To better understand why such practice works, we also include a toy experiment to visualize the *embedding quality* in the Poincaré ball, which reasonably shows that our kernels make better use of geometry constraints.

The code will be made available freely to the research community.

2 Related Work

Our work mainly focuses on integrating geometry in deep learning frameworks and kernel methods over the manifold. In this section, we briefly give an overview of related works.

2.1 Geometric Constraint Learning

In the deep learning era, the representation power of raw data can be improved by integrating geometric constraints in deep neural networks. That is, the network benefits from the underlying property of the geometry, thereby pushing the network to encode complex structures of data. In Fang et al. (2021b); Simon et al. (2020), a couple of images can be represented by a set or a subspace. In SVDNet, the orthogonality constraint enforces the fully connected layer lying on the Grassmannian manifold, which de-correlates the features among entries (Sun et al., 2017). Constraining the trajectory of models in the Grassmannian manifold benefits the continual learning to prevent catastrophic forgetting (Simon et al., 2021). The works in Liu et al. (2017); Meng et al. (2019) also show that embedding in a spherical space is particularly effective for similarity learning (*e.g.*, face verification, clustering, metric learning) compared to using Euclidean spaces.

In recent years, hyperbolic geometry has gained substantial interest thanks to its tree-like nature, and the ability to encode hierarchical relationships in data. Generalizing the basic operations in the Euclidean geometry, the very first work develops hyperbolic layers in neural networks (Ganea et al., 2018). The following works further show the success of hyperbolic embeddings for graph-structured data, language data, visual data as well as 3-D data (Liu et al., 2019; Gulcehre et al., 2019; Khrulkov et al., 2020; Chen et al., 2020). Such works enjoy the powerful hyperbolic representation by exploiting the data hierarchy. More complex structures of data are also studied in Gu et al. (2019); Skopek et al. (2020), which represents the data in a mixed-curvature geometry. The success of the data embedding in non-flat spaces shows that learning the data distribution/structure is important in building the discriminative data representation. Despite its significant performance gain in existing works, the hyperbolic embedding is projected from the tangent space, it thus cannot fully utilize the property of the hyperbolic geometry in the sense that every presentation is approximated, which flattens the geometry, especially for the points which are away from the origin.

2.2 Kernel Methods

Kernel methods have been studied extensively and proven its success in a broad range of machine learning approaches, *e.g.*, SVM, PCA and clustering (Hofmann et al., 2008). The main idea of kernel methods is to project the input samples, to a high-dimensional (or even infinite-dimensional) Reproducing Kernel Hilbert Space (RKHS), where the projected data can be analyzed with linear models. To avoid explicit lifting to RKHS, the kernel trick provides a simple way to generate the similarity measure of pairs in RKHS. Following this line of research, various kernel formulations are defined (Hofmann et al., 2008), *e.g.*, polynomial kernel, RBF kernel, Laplace kernel, *etc*. A natural generalization over the use of such a single and fixed kernel is on learning the adaptive kernel functions, *e.g.*, multiple kernel learning (MKL) scheme (Rakotomamonjy et al., 2008; Lanckriet et al., 2004). In MKL, the final kernel formulation is a conic combination of base kernels and the weights are learned from the data, such that the learned kernel machine can match with the data to the utmost extent (Rakotomamonjy et al., 2008; Wang et al., 2021).

As of late, attempts to boost the representational power of structured data by generalizing the kernel methods to non-linear geometries have gained increasing attention. The common strategy to define a valid pd kernel on non-Euclidean geometries is to adopt a proper distance metric. In Jayasumana et al. (2013), the authors propose the main theoretical framework to design the Gaussian kernel on symmetric positive definite matrices. The proposed theory is further verified to develop the Gaussian kernel on the Grassmann manifold (Jayasumana et al., 2015). More kernels for the Grassmann manifold are studied in Harandi et al. (2014). In Harandi et al. (2014), the pd Grassmannian kernels are proposed by adopting the equivalent embedding functions. The kernels using the Fisher information metric are developed for the persistence diagrams in Le and Yamada (2018). In Jayasumana et al. (2014), the radial kernel for a series of compact manifolds (*e.g.*, n-sphere, Grassmann manifold and shape manifold) is also developed. The closest study to our work is the work of Cho et al. (2019), which formulates the support vector machine (SVM) in hyperbolic spaces. To facilitate the nonlinear decision boundaries, the kernel SVM for the hyperbolic space is also introduced in Cho et al. (2019). However, the proposed kernel in Cho et al. (2019) is not pd, such that it does not have theoretical properties of pd kernel. In another word, the indefinite kernel the proposed indefinite kernel is not a universal kernel and hence violates the universal approximation property (Micchelli et al., 2006). In addition, training

the indefinite kernel is not easy as it requires stabilizing the loss value (Ong et al., 2004).

In contrast to existing works, our work develops the theoretical framework for positive definite kernels on hyperbolic geometry. As a complementary concept to the indefinite kernel, our work kernelizes the hyperbolic space, and thus embeds hyperbolic data into a high, possibly infinite, dimensional Hilbert space, such that the resulting representations benefit from kernel machines. In the remainder of this paper, we will present the developed theory and evaluate the algorithms across different challenging applications.

3 Preliminaries and Background

3.1 Notations

Formally, let $\mathbb{H}^n$, $\mathbb{R}^n$, $\mathbb{R}^{m\times n}$ and $\mathcal{H}$ denote n-dimensional hyperbolic spaces, n-dimensional Euclidean spaces, spaces of $m\times n$ real-valued matrices and Hilbert spaces. The symbol $\mathbb{N}_0$ is a set of positive integer and 0, defined as $\mathbb{N}_0 := \mathbb{N}\cup 0$. Throughout the paper, the matrices and vectors are denoted by bold capital letters (*e.g.*, $\boldsymbol{X}$) and bold lower-case letters (*e.g.*, $\boldsymbol{x}$), respectively. The transpose of a matrix (*e.g.*, $\boldsymbol{X}$) or a vector (*e.g.*, $\boldsymbol{x}$) is denoted by the superscript $\top$, *e.g.* $\boldsymbol{X}^\top$ or $\boldsymbol{x}^\top$. The sigmoid function is defined as $\text{sigmoid}(\cdot):\mathbb{R}\to\mathbb{R}$, $\text{sigmoid}(x) := \frac{1}{1+e^{-x}}$. The hyperbolic tangent function is defined as $\tanh(\cdot):\mathbb{R}\to\mathbb{R}$, $\tanh(x):=\frac{e^{2x}-1}{e^{2x}+1}$ and its inverse is defiend as $\tanh^{-1}(\cdot):\mathbb{R}\to\mathbb{R}$, $\tanh^{-1}(x) := \frac{1}{2}\ln\left(\frac{1+x}{1-x}\right)$, $|x|<1$. The Iverson bracket for the mathematical statement $\boldsymbol{e}$, denoted by $[\![\boldsymbol{e}]\!]$, is defined by:

$$[\![\boldsymbol{e}]\!] = \begin{cases} 1 \text{ if } \boldsymbol{e} \text{ is true,} \\ 0 \text{ otherwise.} \end{cases} \tag{1}$$

3.2 Hyperbolic Geometry

An n-dimensional hyperbolic space $\mathbb{H}^n$ is a Riemannian manifold with a constant negative curvature (Absil et al., 2007). Following the common practice of employing the hyperbolic geometry as embedding space, we use the Poincaré ball $\mathbb{D}$ to work with the hyperbolic space. The Poincaré ball is a model of n-dimensional hyperbolic geometry in which all points are embedded within an n-dimensional sphere (or inside a circle in the 2D case which is called the Poincaré disk model). Formally, the Poincaré ball model, with curvature $-c$ ($c>0$), is defined as a manifold $\mathbb{D}_c^n = \{\boldsymbol{z}\in\mathbb{R}^n : c\|\boldsymbol{z}\|<1\}$, with the Riemannian metric $g_c^{\mathbb{D}}(\boldsymbol{z}) = \lambda_c^2(\boldsymbol{z})\cdot g^E$, in which $\lambda_c(\boldsymbol{z})$ is the conformal factor, defined as $\frac{2}{1-c\|\boldsymbol{z}\|^2}$, and $g^E = \mathbf{I}_n$ is the Euclidean metric tensor. Furthermore and to facilitate vector operations, the Möbius gyrovector space may come in handy. The *Möbius addition* for $\boldsymbol{z}_i, \boldsymbol{z}_j \in \mathbb{D}_c^n$ is defined as:

$$\boldsymbol{z}_i \oplus_c \boldsymbol{z}_j = \frac{(1+2c\langle \boldsymbol{z}_i,\boldsymbol{z}_j\rangle + c\|\boldsymbol{z}_j\|^2)\boldsymbol{z}_i + (1-c\|\boldsymbol{z}_i\|^2)\boldsymbol{z}_j}{1+2c\langle \boldsymbol{z}_i,\boldsymbol{z}_j\rangle + c^2\|\boldsymbol{z}_i\|^2\|\boldsymbol{z}_j\|^2}. \tag{2}$$

In the Poincaré ball, the *geodesic distance*[2] for two points $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$ on $\mathbb{D}_c^n$ is:

$$d_c(\boldsymbol{z}_i,\boldsymbol{z}_j) = \frac{2}{\sqrt{c}}\tanh^{-1}(\sqrt{c}\|-\boldsymbol{z}_i\oplus_c\boldsymbol{z}_j\|). \tag{3}$$

For a point $\boldsymbol{z}\in\mathbb{D}_c^n$, the tangent space at $\boldsymbol{z}$, denoted by $T_{\boldsymbol{z}}\mathbb{D}_c^n$, is an inner product space, which contains the tangent vector with all possible directions at $\boldsymbol{z}$. The Riemannian metric $g_c^{\mathbb{D}}$ at point $\boldsymbol{z}$ is a positive definite symmetric bilinear function on $T_{\boldsymbol{z}}\mathbb{D}_c^n$ as $g_c^{\mathbb{D}}(\boldsymbol{z}) : (T_{\boldsymbol{z}}\mathbb{D}_c^n \times T_{\boldsymbol{z}}\mathbb{D}_c^n)\to\mathbb{R}$. In other word, the tangent space at $\boldsymbol{x}$ is a Euclidean space, and the scale factor is decided by the conformal factor $\lambda_c(\boldsymbol{z})$.

The *exponential map* provides a way to project a point $\boldsymbol{p}\in T_{\boldsymbol{z}}\mathbb{D}_c^n$ to the Poincaré ball $\mathbb{D}_c^n$, as follows:

$$\Gamma_{\boldsymbol{z}}(\boldsymbol{p}) = \boldsymbol{z}\oplus_c\left(\tanh\left(\sqrt{c}\frac{\lambda_c(\boldsymbol{z})\cdot\|\boldsymbol{p}\|}{2}\right)\frac{\boldsymbol{p}}{\sqrt{c}\|\boldsymbol{p}\|}\right). \tag{4}$$

The inverse process of the exponential map is termed *logarithm map*, which projects a point $\boldsymbol{q}\in\mathbb{D}_c^n$, to the tangent plane of $\boldsymbol{z}$ (*i.e.*, $T_{\boldsymbol{z}}\mathbb{D}_c^n$), and is given as:

$$\Upsilon_{\boldsymbol{z}}(\boldsymbol{q}) = \frac{2}{\sqrt{c}\lambda_c(\boldsymbol{z})}\tanh^{-1}(\sqrt{c}\|-\boldsymbol{z}\oplus_c\boldsymbol{q}\|)\frac{-\boldsymbol{z}\oplus_c\boldsymbol{q}}{\|-\boldsymbol{z}\oplus_c\boldsymbol{q}\|}. \tag{5}$$

Note that $\Upsilon_{\boldsymbol{z}}(\Gamma_{\boldsymbol{z}}(\boldsymbol{p})) = \boldsymbol{p}\in T_{\boldsymbol{z}}\mathbb{D}_c^n$. Both the exponential and the logarithm maps are injective functions in the Poincaré model. In this paper, we leverage the scaled Euclidean space in the identity tangent plane to define the Poincaré kernels for hyperbolic spaces.

4 Poincaré Kernels for Hyperbolic Representations

In this section, we propose positive definite (pd) kernels in hyperbolic spaces. Essentially, we are interested in identifying a bivariate function $k(\cdot,\cdot):(\mathbb{D}_c^n\times\mathbb{D}_c^n)\to\mathbb{R}$, which

[2] The geodesic is the shortest path between two points. Its length is termed geodesic distance. For example, the geodesic of the Euclidean space is a straight line connecting two points, and it becomes the well-known Euclidean distance. On the contrary, the geodesic of the n-sphere is the curve along the sphere, such that the geodesic distance is the length of the curve.

represents an inner product in a Reproducing Kernel Hilbert Space (RKHS). Obviously, not all bivariate functions constitute valid kernels, meaning that they do not necessarily realize an RKHS.

Also, popular kernels in Euclidean spaces cannot lead to meaningful solutions as they are not faithful to the geometry of the hyperbolic spaces. Embedding hyperbolic points into an RKHS is not only theoretically appealing but can also result in practical benefits due to the intriguing properties of RKHSs. Though the indefinite kernels are developed in hyperbolic spaces (Cho et al., 2019), we believe the development of pd kernels is also necessary for the reason that pd kernels are core to many developments in machine learning. That is, the pd property is essential for various algorithms such as Gaussian process (Hofmann et al., 2008), two-sample tests in RKHS (Gretton et al., 2012) and many more. Focusing on deep learning and as an example, the NTK (Jacot et al., 2018) relies on the pd property. Add to this, the recent work (Domingos, 2020), where again developments make use of the pd property.

In this paper, we make use of the tangent space of the hyperbolic geometry to define a set of valid pd kernels. We start by formally defining a pd kernel.

Definition 1 (Positive Definite Kernels (Berg et al., 1984)) Let $\mathcal{Z}$ be a non-empty set. A symmetric function $k(\cdot,\cdot) : (\mathcal{Z} \times \mathcal{Z}) \to \mathbb{R}$ is a positive definite (pd) kernel on the set $\mathcal{Z}$ if and only if $\sum_{i,j=1}^{m} c_i c_j k(z_i, z_j) \geq 0$ for any $m \in \mathbb{N}$, $z_i \in \mathcal{Z}$ and $c_i \in \mathbb{R}$.

Essential to our work is the following lemma;

Lemma 1 *Let $\mathcal{Z}$ be a non-empty set. Consider a function $f(\cdot) : \mathcal{Z} \to \mathbb{R}^n$, that maps each element of $\mathcal{Z}$ to $\mathbb{R}^n$. Then,*

$$k(z_i, z_j) = \left\langle f(z_i), f(z_j) \right\rangle$$

is a pd kernel on $\mathcal{Z}$.

Proof The proof of this lemma follows immediately from Definition 1. To see this, define

$$\boldsymbol{F}_{n\times m} := \left[f(z_1), f(z_2), \cdots, f(z_m) \right] .$$

Now, notice that

$$\sum_{i,j=1}^{m} c_i c_j k(z_i, z_j) = \boldsymbol{c}^\top \boldsymbol{K} \boldsymbol{c} = \boldsymbol{c}^\top \boldsymbol{F}^\top \boldsymbol{F} \boldsymbol{c} = \|\boldsymbol{F}\boldsymbol{c}\|^2 \geq 0 .$$

The $\left[\boldsymbol{K}_{m\times m}\right]_{i,j} = k(z_i, z_j)$ is called the gram matrix. □

Based on Lemma 1, we propose to make use of $f_{\mathbb{D}}(\cdot) : \mathbb{D}_c^n \to \mathbb{R}^n$ defined as,

$$f_{\mathbb{D}}(z) := \tanh^{-1}(\sqrt{c}\|z\|) \frac{z}{\sqrt{c}\|z\|}, \tag{6}$$

to develop valid pd kernels on $\mathbb{D}_c^n$. The function $f_{\mathbb{D}}(\cdot)$ enjoys various unique properties. First note that the function is bijective and $f_{\mathbb{D}}(z) = \Upsilon_{\mathbf{0}}(z)$. The next theorem establishes an important property and justifies our choice here better.

Theorem 1 (Curve Length Equivalence) *A curve in $\mathbb{D}_c^n$ is a continuous function $\gamma(\cdot) : [0, 1] \to \mathbb{D}_c^n$; joining the starting point $\gamma(0)$ to the end point $\gamma(1)$. Define the distance induced by $f_{\mathbb{D}}$ as*

$$d_e\left(z_i, z_j\right) := \left\| f_{\mathbb{D}}(z_i) - f_{\mathbb{D}}(z_j) \right\|. \tag{7}$$

The length of any given curve γ is the same under d_e and the geodesic distance d_c up to a scale of $1/\tilde{\lambda}_c$, where $\tilde{\lambda}_c = 2$ is the conformal factor at the origin.

Proof The proof is relegated to the Supplementary Material of our paper due to space limitations. □

Having $f_{\mathbb{D}}(\cdot)$ at our disposal, we are now ready to define the Poincaré kernels in hyperbolic spaces.

4.1 Poincaré Tangent Kernel

The simplest pd kernel resembles the linear kernel in Euclidean spaces and is defined as

$$\boxed{k^{\tan}(z_i, z_j)} = \langle f_{\mathbb{D}}(z_i), f_{\mathbb{D}}(z_j) \rangle. \tag{8}$$

We call this kernel Poincaré tangent kernel as it can be understood as the linear kernel in the identity tangent space of the Poincaré ball. This kernel is attractive as it is parameter-less, making it ideal for fast prototyping. The proof of positive-definiteness of the hyperbolic tangent kernel follows directly from Lemma 1.

4.2 Poincaré RBF Kernel

The Gaussian RBF kernel is a popular universal kernel in Euclidean spaces. In $\mathbb{R}^n$, the RBF kernel can be written as $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\xi \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2)$, $\xi > 0$, where the metric is the squared Euclidean distance in $\mathbb{R}^n$. Taking into account the properties of the RBF kernel (Christmann & Steinwart, 2008), it is very desirable to extend this kernel to hyperbolic spaces. One may assume that replacing the Euclidean distance by the geodesic distance (*i.e.*, Eq. (3)) can lead to a valid pd kernel. This, unfortunately, is not the case as shown by the toy example below.

Example 1 Consider a 3-dimensional Poincaré ball with curvature $c = 0.1$ (*i.e.*, $\mathbb{D}_{0.1}^3$) and the following points in $\mathbb{D}_{0.1}^3$:

$$z_1 = \begin{bmatrix} 0.1885 \\ 0.2330 \\ 0.9526 \end{bmatrix}, z_2 = \begin{bmatrix} 0.6586 \\ 0.2053 \\ 0.0894 \end{bmatrix}, z_3 = \begin{bmatrix} 0.3017 \\ 0.4155 \\ 0.5357 \end{bmatrix}, z_4 = \begin{bmatrix} 0.2388 \\ 0.8290 \\ 0.3790 \end{bmatrix}.$$

The gram matrix (*i.e.*, $\exp(-\xi d_c^2(z_i, z_j))$ for $\xi = 0.01$) for these four points has a negative eigenvalue of -3.0605×10^{-5}.

Further to the counterexample above, the RBF kernel derived from the geodesic distance is shown to be pd iff the space is isometric to the Euclidean space per the following theorem.

Theorem 2 (Theorem 6.2 in Jayasumana et al. (2015)) *Let $\mathcal{M}$ be a complete Riemannian manifold and $d_{\mathcal{M}}$ be the induced geodesic distance on the manifold. The Gaussian RBF kernel $k(\cdot,\cdot) : (\mathcal{M} \times \mathcal{M}) \to \mathbb{R} : k(\boldsymbol{m}_i, \boldsymbol{m}_j) := \exp(-\xi d_{\mathcal{M}}^2(\boldsymbol{m}_i, \boldsymbol{m}_j))$ is positive definite for all $\xi > 0$ if and only if the Riemannian manifold $\mathcal{M}$ is isometric to some Euclidean space $\mathbb{R}^n$.*

According to Theorem 2, it is theoretically impossible to obtain a valid RBF kernel using geodesic distance on hyperbolic spaces.[3] Given the above, we propose to make use of $d_e(\cdot,\cdot)$ and define the Poincaré RBF kernel as

$$\boxed{k^{\text{rbf}}(z_i, z_j)} = \exp\left(-\xi \| f_{\mathbb{D}}(z_i) - f_{\mathbb{D}}(z_j)\|^2\right). \tag{9}$$

To show that the form in Eq. (9) is a valid pd kernel, we first define negative definite (nd) kernels.

Definition 2 (Negative Definite Kernels (Berg et al., 1984)) Let $\mathcal{Z}$ be a non-empty set. A symmetric function $k(\cdot,\cdot) : (\mathcal{Z} \times \mathcal{Z}) \to \mathbb{R}$ is a negative definite (nd) kernel on the set $\mathcal{Z}$ if and only if $\sum_{i,j=1}^{m} c_i c_j k(z_i, z_j) \leq 0$ for any $m \in \mathbb{N}$, $z_i \in \mathcal{Z}$ and $c_i \in \mathbb{R}$ with $\sum_{i=0}^{m} c_i = 0$.

Note the difference between pd and nd kernels. For nd kernels, an additional condition (*i.e.*, $\sum_{i=0}^{m} c_i = 0$) is required. The following lemma shows that $d_e^2(\cdot,\cdot) = \| f_{\mathbb{D}}(z_i) - f_{\mathbb{D}}(z_j)\|^2$ is indeed nd.

Lemma 2 *Let $\mathcal{Z}$ be a non-empty set. An injective function $f(\cdot) : \mathcal{Z} \to \mathbb{R}^n$, maps each vector in $\mathcal{Z}$ onto an inner product space $\mathbb{R}^n$. Then $k(z_i, z_j) := \| f(z_i) - f(z_j)\|^2$ is negative definite.*

Proof The proof is relegated to the Supplementary Material of our paper due to space limitations. □

The following important theorem establishes the connection between positive definite kernels and negative definite kernels.

[3] If a manifold $\mathcal{M}$ is isometric to some Euclidean spaces $\mathbb{R}^n$, then the geodesic distance on $\mathcal{M}$ is the Euclidean distance in $\mathbb{R}^n$. However, it is impossible to find an isometry between $\mathbb{D}_c^n$ and $\mathbb{R}^n$ because of the difference in the curvature of two geometries.

Theorem 3 (Berg et al., 1984) *Let $\mathcal{Z}$ be a non-empty set and $k(\cdot,\cdot) : (\mathcal{Z} \times \mathcal{Z}) \to \mathbb{R}$ be a kernel. The exponential type kernel $k(z_i, z_j) = \exp(-\xi \Phi(z_i, z_j))$ is positive definite for all $\xi > 0$ if and only if $\Phi(\cdot,\cdot)$ is negative definite.*

Stating the fact that $d_e^2(\cdot,\cdot)$ is nd along with Theorem 3 concludes our claim that the Poincaré RBF kernel defined in Eq. (9) is pd.

4.3 Poincaré Laplace Kernel

The Laplace kernel is another widely used universal kernel in Euclidean spaces, formulated as $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\xi \|\boldsymbol{x}_i - \boldsymbol{x}_j\|)$, $\xi > 0$. When extending the Laplace kernel to hyperbolic spaces, we use the following theorem to build a nd kernel for hyperbolic spaces.

Theorem 4 (Berg et al., 1984) *If $k : (\mathcal{Z} \times \mathcal{Z}) \to \mathbb{R}$ is negative definite and satisfies $k(z_i, z_j) \geq 0$, then k^α is also negative definite for $0 < \alpha < 1$.*

Combining Theorem 3 and Theorem 4, and choosing $\alpha = \frac{1}{2}$, we could obtain the Poincaré Laplace kernel as

$$\begin{aligned} \boxed{k^{\text{lap}}(z_i, z_j)} &= \exp\left(-\xi d_e(f_{\mathbb{D}}(z_i), f_{\mathbb{D}}(z_j))\right) \\ &= \exp\left(-\xi \| f_{\mathbb{D}}(z_i) - f_{\mathbb{D}}(z_j)\|\right). \end{aligned} \tag{10}$$

A more general form of the Poincaré Laplace kernel (*i.e.*, generalized Poincaré Laplace kernel) can be further derived as:

$$\boxed{k^{\text{glap}}(z_i, z_j)} = \exp\left(-\xi \| f_{\mathbb{D}}(z_i) - f_{\mathbb{D}}(z_j)\|^{2\alpha}\right), \tag{11}$$

where $0 < \alpha < 1$.

4.4 Poincaré Binomial Kernel

In addition to the exponential type kernels, we further construct a Poincaré binomial kernel. To obtain the Poincaré binomial kernel, we make use of the following lemma.

Lemma 3 *Let $\mathcal{Z}$ be a non-empty set. An injective function $f : \mathcal{Z} \to \mathbb{R}^n$, maps each vector in $\mathcal{Z}$ onto an inner product space $\mathbb{R}^n$. Then $k(z_i, z_j) := \left(1 - \langle f(z_i), f(z_j)\rangle\right)^{-\alpha}$ defines a binomial kernel on $\mathcal{Z}$ when $\alpha > 0$ and $\| f(z)\| < 1$.*

Proof According to Lemma 4.8 of Christmann and Steinwart (2008), if the function $k(\cdot,\cdot)$ can be decomposed by a full Taylor series with each term being non-negative, then we can claim $k(\cdot,\cdot)$ is a valid pd kernel. Let $t = \langle f(z_i), f(z_j)\rangle$, the binomial series $k(z_i, z_j) = (1-t)^{-\alpha} = \sum_{n=0}^{\infty} \binom{-\alpha}{n} (-1)^n t^n$ holds for all $|t| < 1$, where the binomial coefficient $\binom{\beta}{n} := \prod_{i=1}^{n} (\beta - i + 1)/i$. It can be seen $\binom{-\alpha}{n} (-1)^n > 0$ when $\alpha > 0$, which indicates the binomial kernel has a non-negative and full Taylor series. □

According to the Lemma 3, we could obtain the Poincaré binomial kernel as

$$\boxed{k^{\text{bin}}(z_i, z_j)} = \left(1 - \langle f_{\mathbb{D}}(z_i), f_{\mathbb{D}}(z_j)\rangle\right)^{-\alpha}, \ \alpha > 0. \tag{12}$$

Also, given the non-negativeness and full Taylor series in the above proof, we can further claim that the Poincaré binomial kernel satisfies the necessary and sufficient condition of being universal, shown in Corollary 4.57 of Christmann and Steinwart (2008).

4.5 Poincaré Radial Kernel

The above proposed universal kernels suffer from the difficulty of tuning the hyper-parameters (*e.g.*, ξ and α in Table 1) to attain good performances. One can address this issue by employing multiple kernel learning (MKL) scheme, which automates the kernel learning and learns an optimal kernel function per task. In this section, we will develop a learnable kernel, termed Poincaré radial kernel, which is a combination of simple inner product kernels. Instead of using Eq. (6), we leverage the mapping:

$$g_{\mathbb{D}}(z) := \frac{f_{\mathbb{D}}(z)}{\|f_{\mathbb{D}}(z)\|}, \tag{13}$$

to develop our Poincaré radial kernel. This mapping can be understood as mapping the points in the Poincaré ball to a n-sphere in the tangent plane of identity. It's a natural choice that mapping points to the sphere benefits the real-practice in conjunction with the CNNs (Liu et al., 2017; Hao et al., 2019).

Given the mapping $g_{\mathbb{D}}(\cdot)$, we first formulate the Poincaré cosine kernel as the basic kernel unit in MKL, as follows:

$$k^{\cos}(z_i, z_j) = \langle g_{\mathbb{D}}(z_i), g_{\mathbb{D}}(z_j)\rangle. \tag{14}$$

Referring to Lemma 1, one can easily prove that $k^{\cos}(\cdot,\cdot)$ is pd in the Poincaré ball $\mathbb{D}_c^n$. We then define other components by the following closure properties of pd kernels (Jayasumana et al., 2014; Berg et al., 1984).

1. If two kernels k_1 and k_2 are pd, then so is $k_1 k_2$, and therefore k_1^n, for all $n \in \mathbb{N}$.
2. If all kernel in a point-wise convergent sequence $k_1, k_2, \ldots$ are pd, then their point-wise limit $k = \lim_{i\to\infty} k_i$ is also pd.
3. If two kernels k_1 and k_2 are pd, then so is their conic combination $a_1 k_1 + a_2 k_2$, where $a_1, a_2 \geq 0$.

From the 1st property and the trivial result that $k^0 = 1$ is pd, we have:

$$k_m^{\cos}(z_i, z_j) = \langle g_{\mathbb{D}}(z_i), g_{\mathbb{D}}(z_j)\rangle^m, \tag{15}$$

where $m \in \mathbb{N}_0$. The 2nd property can be employed to identify the following two kernels:

$$k_{-1}^{\cos}(z_i, z_j) = \begin{cases} 1 & \text{if } g_{\mathbb{D}}(z_i) = g_{\mathbb{D}}(z_j), \\ -1 & \text{if } g_{\mathbb{D}}(z_i) = -g_{\mathbb{D}}(z_j), \\ 0 & \text{otherwise}, \end{cases} \tag{16}$$

and

$$k_{-2}^{\cos}(z_i, z_j) = \begin{cases} 1 & \text{if } g_{\mathbb{D}}(z_i) = \pm g_{\mathbb{D}}(z_j), \\ 0 & \text{otherwise}, \end{cases} \tag{17}$$

with the 3rd property, one can develop the pd kernels via combining the kernels in Eqs. (15), (16) and (17). Specifically, the following theorem gives us theoretical support for our development of Poincaré radial kernel.

Remark 1 Eq. (16) resembles an indicator function in the sense that it rewards if $g_{\mathbb{D}}(z_i)$ matches $g_{\mathbb{D}}(z_j)$. It also penalizes if $g_{\mathbb{D}}(z_i)$ sits opposite to $g_{\mathbb{D}}(z_j)$ on the sphere realized by Eq. (13) in the tangent space at origin of $\mathbb{D}^n$. In contrast, Eq. (17) rewards both cases equally.

Theorem 5 (Theorem 4.4 in Jayasumana et al. (2014)) *Let $(\mathcal{Z}, d)$ be a metric space and $\mathcal{S}_{\mathcal{H}}$ be the unit sphere in a real Hilbert space $\mathcal{H}$. If there exists a function $g(\cdot) : \mathcal{Z} \to \mathcal{S}_{\mathcal{H}}$ that is a scaled isometry between $(\mathcal{Z}, d)$ and $(\mathcal{H}, \|.\|)$, then any kernel $k(\cdot,\cdot)$ of the form:*

$$k(z_i, z_j) = \sum_{m=-2}^{\infty} a_m k_m^{\cos}(g(z_i), g(z_j)), \tag{18}$$

where $\sum_m a_m < \infty$ and $a_m \geq 0$ for all m, is pd and radial on $(\mathcal{Z}, d)$. Furthermore, if $g(\cdot) : \mathcal{Z} \to \mathcal{S}_{\mathcal{H}}$ is surjective, all pd radial kernels on $(\mathcal{Z}, d)$ are of this form.

Having the induced cosine-type kernels and Theorem 5 at hand, our Poincaré radial kernel can be formulated as:

$$\begin{aligned} k^{\text{rad}}(z_i, z_j) = &\sum_{m=0}^{\infty} a_m k_m^{\cos}(z_i, z_j) \\ &+ a_{-1}\left([\![k_{-1}(z_i, z_j) = 1]\!] - [\![k_{-1}(z_i, z_j) = -1]\!]\right) \\ &+ a_{-2}[\![k_{-2}(z_i, z_j) \in \{-1, 1\}]\!], \end{aligned} \tag{19}$$

where $a_m \geq 0$ and $\sum_{m=-2}^{\infty} a_m < \infty$. $[\![\cdot]\!]$ indicates the Iverson bracket.

Plugging Eq. (14) into Eq. (19), we can obtain the final formulation of the Poincaré radial kernel, as:

$$\boxed{k^{\text{rad}}(z_i, z_j)} = \sum_{m=0}^{\infty} a_m \langle g_{\mathbb{D}}(z_i), g_{\mathbb{D}}(z_j)\rangle^m + a_{-1}\left([\![k_{-1}(z_i, z_j) = 1]\!] - [\![k_{-1}(z_i, z_j) = -1]\!]\right) + a_{-2}[\![k_{-2}(z_i, z_j) \in \{-1, 1\}]\!]. \tag{20}$$

To ensure the learned Poincaré radial kernel remains pd during the training process, one needs to constrain the weights (a_m in Eq. (20)) to have positive values. This can be achieved by imposing an activation function on the weights. The commonly used activation functions such as ReLU$(\cdot)$, softmax$(\cdot)$, and sigmoid$(\cdot)$ can be used for this purpose. Following the practice in Jayasumana et al. (2021), we adopt the sigmoid$(\cdot)$ function to ensure this constraint, *i.e.*, $a_m := \text{sigmoid}(a_m)$. This choice is also justified in Sect. 5.8.

Also, in Eq. (20), the Poincaré radial kernel contains infinite series of base kernels. While in practice, a few first terms can properly approximate the kernel. In another word, we need to choose an optimal number of kernels M and throughout the paper, the number is set to $M = 50$, which will be justified in Sect. 5.

Remark 2 Eq. (6) can provide a simple way to construct an MKL function as:

$$k(z_i, z_j) = \sum_{m=0}^{M} a_m \langle f_{\mathbb{D}}(z_i), f_{\mathbb{D}}(z_j)\rangle^m \ . \tag{21}$$

For $a_m \geq 0$, this kernel is pd. However, empirically we observe that a DNN optimized with the above kernel $k(\cdot, \cdot)$ may become instable in convergence and does not necessarily lead to improved performances. This is verified in the few-shot learning (FSL) task. Specifically, in the *mini*ImageNet dataset, the recognition accuracy of the network is degraded by 2.61% and 1.23% in the 5-way 1-shot and 5-way 5-shot settings, respectively. This motivates the development of the Poincaré radial kernel using Eq. (13).

Remark 3 Noted that both Poincaré tangent kernel in Eq. (8) and Poincaré cosine kernel in Eq. (14) are inner product kernels in $T_{\mathbf{0}}\mathbb{D}_c^n$. In another word, Poincaré cosine kernel is the normalized version of Poincaré tangent kernel, and can be understood as a cosine similarity in the identity tangent plane for $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. In addition, the Poincaré cosine kernel is the basic kernel unit for the Poincaré radial kernel (see Eq. (20)), required by the Theorem 5.

Remark 4 As alluded to earlier, we have made use of the identity tangent space of the Poincaré ball (*i.e.*, $\mathbb{D}_c^n$) to define pd kernels for the hyperbolic spaces. This implies that the kernels are defined using the Lie algebra of $\mathbb{D}_c^n$. Such a construction has been used with success in other manifolds (*e.g.*, SPD as in Jayasumana et al. (2015)).

In this paper, we employ the kernels along with convolutional neural networks (CNNs) to attain rich models for computer vision tasks. The CNNs encode the input data to vectors, distributed in hyperbolic spaces. Then the proposed kernels are further used to train the network.

5 Experiments

In this section, we will conduct comprehensive experiments to verify the superiority of the proposed Poincaré kernels. We first use a toy example to evaluate the embedding quality in the hyperbolic space, learned by kernels. Thereafter, a variety of challenging tasks are adopted to verify the effectiveness and generalization of the proposed algorithms.

5.1 Good Practice of Employing Hyperbolic Geometry

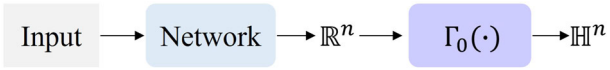
Few works have studied the problem of learning an embedding in hyperbolic spaces (Chen et al., 2020; Khrulkov et al., 2020). However, the existing works generate the vectors in the tangent space at the origin and project to the hyperbolic spaces using $\Gamma_{\mathbf{0}}(\cdot)$[4] mapping (see Fig. 1a). A drawback of this framework is that the hyperbolic geometry is not fully utilized as every representation is flattened at the identity (*i.e.*, origin of Poincaré ball). In other words, only the vectors very close to the origin represent hyperbolic distances. In contrast, and in our experiments, we generate hyperbolic presentations directly in the Poincaré ball, as illustrated in Fig. 1b. To be specific, a neural network encodes the input to a vector representation z, and it can be ensured in the Poincaré ball, with curvature $-c$ ($c > 0$), constrained by the following:

$$z := \begin{cases} z & \text{if } \|z\| \leq \frac{1}{\sqrt{c}} \\ (1-\epsilon)\frac{z}{\sqrt{c}\|z\|} & \text{else,} \end{cases} \tag{22}$$

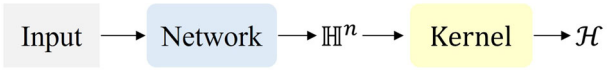
where ϵ is a tiny value used for numerical stability. Throughout this paper, the ϵ value is set to 10^{-3}. The Eq. (22) can be understood as a re-projection that constrains the representation in the Poincaré ball.

To justify the good practice of our choice, we first visualize the embedding quality by a toy example. In this study, we train a simple CNN on the MNIST dataset (LeCun et

[4] Noted the function $\Gamma_{\mathbf{0}}(\cdot)$ realizes the mapping to project the points in the identity tangent space (*i.e.*, $T_{\mathbf{0}}\mathbb{D}_c^n$) into a Poincaré ball (*i.e.*, $\mathbb{D}_c^n$), which is defined as $\Gamma_{\mathbf{0}}(\boldsymbol{x}) = \tanh(\sqrt{c}\|\boldsymbol{x}\|)\frac{\boldsymbol{x}}{\sqrt{c}\|\boldsymbol{x}\|}$ for $\boldsymbol{x} \in T_{\mathbf{0}}\mathbb{D}_c^n$.



(a) Geometry constraints in existing works [6, 45].



(b) Geometry constraints in our work.

Fig. 1 Schematic comparison between existing works and our work in employing constraints from the hyperbolic geometry

al., 1998) over three settings. In the first setting, the network training follows the baseline in Khrulkov et al. (2020), which performs the classification task in the Poincaré ball (as shown in Fig. 1a). The second setting is to train a network following the proposed paradigm without using any kernels. The third setting, following the paradigm of Fig. 1b, trains a network using the simple Poincaré tangent kernel, proposed by our work. Figure 2 illustrates the feature embeddings in a 2-dimensional Poincaré ball under two training settings.

From Fig. 2, we can observe that in both the first (see Fig. 2a) and the third setting (see Fig. 2c), most of the samples are distributed near the boundary. Similar observations were also made in Khrulkov et al. (2020). However, in the second setting (see Fig. 2b), some classes are distributed near the boundary while others are clustered within the Poincaré ball. When comparing the visualization of hyperbolic embeddings from Khrulkov et al. (2020) (see Fig 2a) with our embeddings obtained without using kernels (see Fig. 2b), it is evident that our practice yields more discriminative hyperbolic embeddings. In Khrulkov et al. (2020), the hyperbolic embeddings heavily overlap, and the within-class variance of certain classes (*e.g.*, red and black) is very large. On the other hand, it demonstrates that our approach with the Poincaré tangent kernel (see Fig. 2c) results in more evenly and compactly distributed class clusters, clearly demonstrating the effectiveness of the proposed kernels.

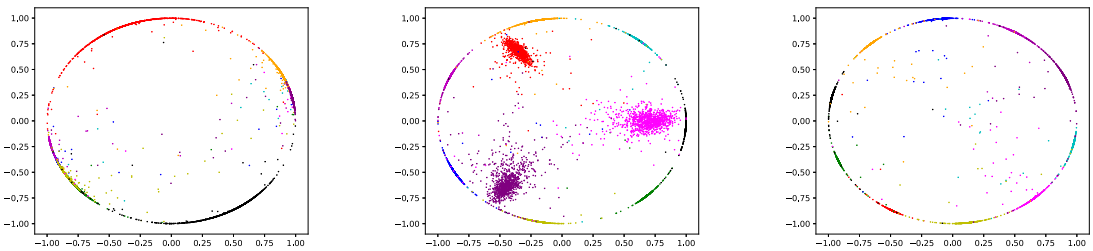
In the remainder of this section, we continue to evaluate the effectiveness of Poincaré kernels on a set of challenging computer vision tasks, *i.e.*, few-shot learning, zero-shot learning, person re-identification, deep metric learning, knowledge distillation and self-supervised learning.

5.2 Few-Shot Learning

Problem setting Few-shot learning (FSL) is required to learn an embedding space, which should be adapted to recognize unseen classes at test time, given only a few samples of each new class (Snell et al., 2017; Sung et al., 2018; Vinyals et al., 2016; Hong et al., 2021). The network is trained in a meta-learning manner (see Fig. 3), which is also known as task-agnostic FSL. In each iteration, we sample an episode of data to train the network. Specifically, this protocol is well-known as N-way K-shot classification, which realizes the N-class recognition task per episode. In our experiments, we follow the general practice (*i.e.*, 5-way 1-shot and 5-way 5-shot) to evaluate the model. We employ the pipeline in the prototypical network (ProtoNet) (Snell et al., 2017) along with the proposed kernels to train the feature extractor.

In the training phase, each episode is composed of a support set $\mathcal{S} = \left\{(\{\boldsymbol{s}_{i,1}, \ldots, \boldsymbol{s}_{i,K}\}, l_i) : i = 1, \ldots, N\right\}$ and a query set $\mathcal{Q} = \{(\boldsymbol{q}_i, l_i) : i = 1, \ldots, N\}$. The prototype of each class is computed by $\hat{\boldsymbol{s}}_i = \frac{1}{K}\sum_{j=1}^{K} \boldsymbol{s}_{i,j}$. Then the prototypical network (ProtoNet) formulates the loss function as:

$$\mathcal{L}_{\text{fsl}}^{\text{E}} = -\frac{1}{N_q}\sum_{i=1}^{N_q} \log\left(\frac{\exp(-\|\boldsymbol{q}_i - \hat{\boldsymbol{s}}^*\|)}{\sum_{j=1}^{N}\exp(-\|\boldsymbol{q}_i - \hat{\boldsymbol{s}}_j\|)}\right), \tag{23}$$



(a) Hyperbolic embeddings from [45].

(b) Hyperbolic embeddings from our work without kernels.

(c) Hyperbolic embeddings from Poincaré tangent kernel.

Fig. 2 Visualization of feature embeddings in hyperbolic spaces learned for MNIST dataset. **a**: Hyperbolic embeddings trained by the pipeline in Khrulkov et al. (2020). **b** Hyperbolic embeddings from our work without kernels. **c**: Hyperbolic embeddings trained by the pipeline in our work. Here, we use the hyperbolic tangent kernel. It shows that the hyperbolic embeddings from our practice are more discriminative than that from Khrulkov et al. (2020), as the hyperbolic embeddings from Khrulkov et al. (2020) are heavily overlapped and the within class variance of some classes (*e.g.*, red and black) are very large. As compared to the one without using kernels in (**b**), our practice with Poincaré tangent kernel in (**c**) makes the class clusters more evenly and compactly, clearly showing the effectiveness of the proposed kernels. Best viewed in color

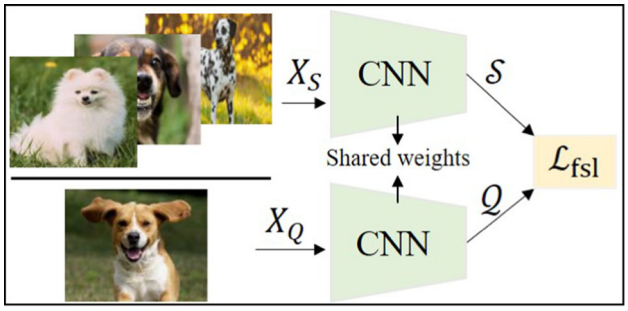


Fig. 3 The pipeline of the deep network for few-shot recognition. $\boldsymbol{X}_S$ and $\boldsymbol{X}_Q$ denote the input images in the support set and query set

where $\boldsymbol{q}_i$ and $\hat{\boldsymbol{s}}^*$ share the same label, and N_q is the number of query samples in one episode.

Noted that $\boldsymbol{q}_i, \hat{\boldsymbol{s}}_i \in \mathbb{R}^n$ for the vanilla ProtoNet, thus the distance used in Eq. (23) is the L_2 distance. Then in the hyperbolic version (*i.e.*, Hyper ProtoNet), where $\boldsymbol{q}_i, \hat{\boldsymbol{s}}_i \in \mathbb{D}_c^n$, the loss is further formulated as:

$$\mathcal{L}_{\text{fsl}}^{\text{H}} = -\frac{1}{N_q}\sum_{i=1}^{N_q}\log\left(\frac{\exp(-d_c(\boldsymbol{q}_i, \hat{\boldsymbol{s}}^*))}{\sum_{j=1}^{N}\exp(-d_c(\boldsymbol{q}_i, \hat{\boldsymbol{s}}_j))}\right), \tag{24}$$

where d_c is the geodesic distance in the Poincaré ball.

We further plug our kernels in the loss functions, as:

$$\mathcal{L}_{\text{fsl}}^{\text{K}} = -\frac{1}{N_q}\sum_{i=1}^{N_q}\log\left(\frac{g(k(\boldsymbol{q}_i, \hat{\boldsymbol{s}}^*))}{\sum_{j=1}^{N} g(k(\boldsymbol{q}_i, \hat{\boldsymbol{s}}_j))}\right), \tag{25}$$

where $k(\cdot,\cdot)$ indicates the kernel, and $\boldsymbol{q}_i, \hat{\boldsymbol{s}}_i \in \mathbb{D}_c^n$. Here, $g(\cdot)$ is exp function if $k(\cdot,\cdot)$ is non-exponential type kernels. Otherwise, $g(\cdot)$ is the identity mapping.

In terms of the feature extractor, we use both Conv-4 (Snell et al., 2017) and ResNet-18 (He et al., 2016) as CNN backbones in our experiments. Moreover, four popular benchmarks, *i.e.*, ***mini*ImageNet** (Deng et al., 2009), **CUB** (Wah et al., 2011), ***tiered*-ImageNet** (Ren et al., 2018) and **Few-shot-CIFAR100** (FC100) (Oreshkin et al., 2018) are adopted to assess our algorithms. The details of datasets are included in the Supplementary Material.

Related work We will review the literature on FSL, primarily based on the metric-learning approaches. In the Matching networks (Vinyals et al., 2016), a sample-wise metric is learned to determine the category of a query. An extension idea develops the class-wise metric (Snell et al., 2017), in which all samples per class is considered as a class descriptor. Explicitly modeling a non-linear relationship is studied in Relation Networks (Sung et al., 2018), such that the latent metric is data-dependent and adaptive. Considering the misalignment issue of objects within images, Zhang *et al.* leverage the optimal transport techniques in the metric space, such that the distance metric can be calculated via optimal patch feature matching (Zhang et al., 2020). The alignment can also be achieved by attention mechanism (Hong et al., 2021) or dynamic filters (Xu et al., 2021). The recent work investigates to model the context information of support set, improving the discrimination of embeddings (Ye et al., 2020).

The close direction to our work is the geometric learning in metric spaces. In Rodríguez et al. (2020), a regularizer is integrated on a manifold, to smooth the update of the embedding. Modeling the support set as a Grassmann manifold can adaptively measure the distance between the query sample and support set (Simon et al., 2020). The hyperbolic geometry is first studied in Khrulkov et al. (2020), revealing that the hierarchical structure in the dataset also benefits the embedding learning.

Results Tables 2 and 3 illustrate empirical results on four datasets. We observe that our algorithms improve the few-shot recognition performance as compared to their hyperbolic counterpart and other advanced methods. In addition, the results from the Poincaré radial kernel, in general, exceed the results from other kernels. For example, in 5-way 1-shot setting, with Conv-4 backbone, the Poincaré radial kernel outperforms the Hyperbolic ProtoNet (Khrulkov et al., 2020) by 2.85, 7.60, 3.52 and 1.65 for *mini*ImageNet, CUB, *tiered*-ImageNet and FC100, respectively. Or in 5-way 5-shot setting, with the same backbone, the Poincaré radial kernel brings the performance gain over the Hyperbolic ProtoNet (Khrulkov et al., 2020) by 4.15, 3.43, 4.91 and 3.06 for *mini*ImageNet, CUB, *tiered*-ImageNet and FC100, respectively, clearly showing the potential and superiority of the universal kernel with MKL protocol. The improvement can also be made in the ResNet backbone, further indicating its generalization.

5.3 Zero-Shot Learning

Problem setting Zero-shot learning (ZSL) aims to identify objects that are unseen during the training phase (Akata et al., 2015a; Xian et al., 2016). Formally, suppose we have a seen set $\mathcal{D}^s$ and an unseen set $\mathcal{D}^u$. Specifically, the seen set, $\mathcal{D}^s = \{(\boldsymbol{v}_i^s, l_i^s, \boldsymbol{a}_i^s), i = 1, \ldots, N^s\}$, contains the visual feature $\boldsymbol{v}_i \in \mathbb{R}^{d_v}$, the semantic feature $\boldsymbol{a}_i \in \mathbb{R}^{d_a}$ for the seen class $l_i^s \in L^s$. Similarly, the unseen set, $\mathcal{D}^u = \{(\boldsymbol{v}_i^u, l_i^u, \boldsymbol{a}_i^u), i = 1, \ldots, N^u\}$, also contains unseen visual feature $\boldsymbol{v}_i^u$, unseen semantic feature $\boldsymbol{a}_i^u$ with the unseen class $l_i^u \in L^u$. It is noted that L^s and L^u should be disjoint, *i.e.*, $L^s \cap L^u = \emptyset$. The pipeline of the network for ZSL is illustrated in Fig. 4.

We then build a baseline network for the scenario of zero-shot recognition. In the training phase, we randomly sample N_b seen visual features as $\boldsymbol{V} = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{N_b}\}$. All the semantic features are projected to the visual space, denoted by $\boldsymbol{E} = \{e(\boldsymbol{a}_1), \ldots, e(\boldsymbol{a}_{|L^s|})\}$, where $|L^s|$ denotes the number of seen classes in the training set. In our implementation, the embedding function (*i.e.*, $e(\cdot)$) is a simple two layer

Table 2 Few-shot classification results on *mini*ImageNet and CUB datasets with 95% confidence interval

Model	Backbone	*mini*ImageNet		CUB	
		5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot
MatchingNet (Vinyals et al., 2016)	Conv-4	43.56 ± 0.84	55.31 ± 0.73	61.16 ± 0.89	72.86 ± 0.70
ProtoNet (Snell et al., 2017)	Conv-4	44.53 ± 0.76	65.77 ± 0.66	51.31 ± 0.91	70.77 ± 0.69
MAML (Finn et al., 2017)	Conv-4	48.70 ± 1.84	63.11 ± 0.92	55.92 ± 0.95	72.09 ± 0.76
RelationNet (Sung et al., 2018)	Conv-4	50.44 ± 0.82	65.32 ± 0.70	62.45 ± 0.98	76.11 ± 0.69
DN4 (Li et al., 2019b)	Conv-4	51.24 ± 0.74	71.02 ± 0.64	53.15 ± 0.84	81.90 ± 0.60
DSN (Simon et al., 2020)	Conv-4	51.78 ± 0.96	68.99 ± 0.69	-	-
Hyper ProtoNet (Khrulkov et al., 2020)	Conv-4	54.43 ± 0.20	72.67 ± 0.15	64.02 ± 0.20	82.53 ± 0.14
Poincaré tangent kernel	Conv-4	55.61 ± 0.21	74.81 ± 0.16	66.14 ± 0.23	82.11 ± 0.15
Poincaré RBF kernel	Conv-4	56.48 ± 0.20	76.09 ± 0.16	70.98 ± 0.22	85.21 ± 0.13
Poincaré Laplace kernel	Conv-4	56.26 ± 0.20	75.35 ± 0.15	68.27 ± 0.23	84.64 ± 0.13
Poincaré binomial kernel	Conv-4	56.82 ± 0.20	75.27 ± 0.15	69.05 ± 0.23	83.00 ± 0.14
Poincaré radial kernel	Conv-4	**57.28 ± 0.18**	**76.82 ± 0.15**	**71.62 ± 0.21**	**85.96 ± 0.14**
Baseline (Chen et al., 2019)	ResNet-18	51.75 ± 0.80	74.27 ± 0.63	65.51 ± 0.87	82.85 ± 0.55
Baseline++ (Chen et al., 2019)	ResNet-18	51.87 ± 0.77	75.68 ± 0.63	67.02 ± 0.77	83.58 ± 0.54
RelationNet (Sung et al., 2018)	ResNet-18	52.48 ± 0.86	69.83 ± 0.68	67.59 ± 0.58	82.75 ± 0.58
MAML (Finn et al., 2017)	ResNet-18	49.61 ± 0.92	65.72 ± 0.77	69.96 ± 1.01	82.70 ± 0.65
MatchingNet (Vinyals et al., 2016)	ResNet-18	52.91 ± 0.88	68.88 ± 0.69	72.36 ± 0.90	83.64 ± 0.60
ProtoNet (Snell et al., 2017)	ResNet-18	54.16 ± 0.82	73.68 ± 0.65	71.88 ± 0.91	86.64 ± 0.51
SNCA (Wu et al., 2018)	ResNet-18	57.80 ± 0.80	72.80 ± 0.70	-	-
Hyper ProtoNet (Khrulkov et al., 2020)	ResNet-18	59.47 ± 0.20	76.84 ± 0.14	72.86 ± 0.22	85.69 ± 0.13
Poincaré tangent kernel	ResNet-18	59.91 ± 0.21	76.65 ± 0.16	73.52 ± 0.22	88.75 ± 0.11
Poincaré RBF kernel	ResNet-18	60.91 ± 0.21	77.12 ± 0.15	75.79 ± 0.21	**89.98 ± 0.11**
Poincaré Laplace kernel	ResNet-18	60.52 ± 0.21	77.33 ± 0.15	74.37 ± 0.21	89.08 ± 0.12
Poincaré binomial kernel	ResNet-18	61.04 ± 0.21	77.01 ± 0.15	74.46 ± 0.22	89.28 ± 0.11
Poincaré radial kernel	ResNet-18	**62.15 ± 0.20**	**77.81 ± 0.15**	**76.02 ± 0.22**	89.64 ± 0.12

† Indicates the network was self-implemented. 1st / 2nd best in "bold" / "(underline)"

Table 3 Few-shot classification results on *tiered*-ImageNet and FC100 datasets with 95% confidence interval.

Model	Backbone	*tiered*-ImageNet		FC100	
		5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot
Hyper ProtoNet† (Khrulkov et al., 2020)	Conv-4	54.44 ± 0.23	71.96 ± 0.20	37.59 ± 0.19	51.76 ± 0.19
Poincaré tangent kernel	Conv-4	54.73 ± 0.22	74.37 ± 0.18	37.66 ± 0.17	52.29 ± 0.18
Poincaré RBF kernel	Conv-4	57.78 ± 0.23	76.11 ± 0.18	38.93 ± 0.18	54.40 ± 0.18
Poincaré Laplace kernel	Conv-4	57.33 ± 0.22	76.48 ± 0.18	37.99 ± 0.17	53.54 ± 0.18
Poincaré binomial kernel	Conv-4	56.72 ± 0.22	75.87 ± 0.18	38.32 ± 0.18	53.50 ± 0.18
Poincaré radial kernel	Conv-4	**57.96 ± 0.22**	**76.87 ± 0.18**	**39.24 ± 0.17**	**54.82 ± 0.18**
Hyper ProtoNet† (Khrulkov et al., 2020)	ResNet-18	62.28 ± 0.23	74.50 ± 0.21	40.64 ± 0.20	52.50 ± 0.30
Poincaré tangent kernel	ResNet-18	63.31 ± 0.23	76.06 ± 0.23	42.18 ± 0.26	54.32 ± 0.32
Poincaré RBF kernel	ResNet-18	64.52 ± 0.22	76.82 ± 0.21	43.84 ± 0.23	56.01 ± 0.30
Poincaré Laplace kernel	ResNet-18	64.38 ± 0.22	77.16 ± 0.21	43.22 ± 0.23	55.47 ± 0.30
Poincaré binomial kernel	ResNet-18	64.12 ± 0.23	76.44 ± 0.23	42.60 ± 0.24	55.08 ± 0.32
Poincaré radial kernel	ResNet-18	**65.33 ± 0.21**	**77.48 ± 0.20**	**44.12 ± 0.20**	**56.28 ± 0.26**

† indicates the network was self-implemented. 1st / 2nd best in "bold" / "(underline)"

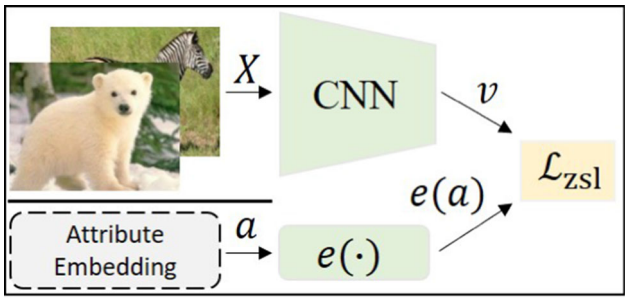


Fig. 4 The pipeline of the deep network for zero-shot learning. $\boldsymbol{X}$ and $\boldsymbol{a}$ denotes input images and attribute descriptors

MLP, with each layer stacking the linear transformation, ReLU activation and batch normalization. Then the network is trained by the following cross-entropy type loss:

$$\mathcal{L}_{\text{zsl}} = -\frac{1}{N_b}\sum_{i=1}^{N_b}\log\left(\frac{\exp\left(-\|(e(\boldsymbol{a}^*)-\boldsymbol{v}_i\|\right)}{\sum_{j=1}^{|L^s|}\exp\left(-\|e(\boldsymbol{a}_j)-\boldsymbol{v}_i\|\right)}\right),$$

where $\boldsymbol{a}^*$ shares the same label with $\boldsymbol{v}_i$. The baseline network is conducted on Euclidean spaces.

Then in our work, the kernelized loss function for the hyperbolic representations (*i.e.*, $e(\boldsymbol{a})$, $\boldsymbol{v} \in \mathbb{D}_c^n$) can be modified as:

$$\mathcal{L}_{\text{zsl}}^{\text{K}} = -\frac{1}{N_b}\sum_{i=1}^{N_b}\log\left(\frac{g\left(k((e(\boldsymbol{a}^*),\boldsymbol{v}_i)\right)}{\sum_{j=1}^{|L^s|}g\left(k(e(\boldsymbol{a}_j),\boldsymbol{v}_i)\right)}\right), \tag{26}$$

where $k(\cdot,\cdot)$ indicates the kernel. Here, $g(\cdot)$ is exp mapping if $k(\cdot,\cdot)$ is non-exponential type kernels. Otherwise, $g(\cdot)$ is the identity mapping.

Four datasets, *i.e.*, **SUN** (Patterson & Hays, 2012), **CUB** (Wah et al., 2011), **AWA1** (Lampert et al., 2013) and **AWA2** (Akata et al., 2015a) are adopted to evaluate our algorithms in the generalized ZSL (G-ZSL) setting. We report the top-1 mean class accuracy (MCA) for both the unseen classes (U) and the seen classes (S) and also calculate the harmonic mean (HM) score, *i.e.* HM $= 2 \times \text{U} \times \text{S}/(\text{U}+\text{S})$. Please refer to the Supplementary Material for more details about the statistics of each dataset.

Related work The task of zero-shot learning (ZSL) connects visual features and semantic features in a unified embedding space (Akata et al., 2015a). Some initial solutions use the low dimensional semantic space as the embedding space, such that the visual feature is projected to the semantic space (Lampert et al., 2013; Frome et al., 2013). However, it may occur the hubness problem. Then alternative solutions are proposed to embed both the semantic features and visual features to a common intermediate space (Akata et al., 2015b; Sung et al., 2018; Zhang & Saligrama, 2015). In recent years, the pipeline that projects semantic features to visual space becomes more popular, for the reason that it can mitigate the hubness problem to a certain degree (Zhang et al., 2017). In Liu et al. (2020), both semantic feature and visual feature are benefited from the hyperbolic geometry in encoding the hierarchical information of the dataset.

Results We first evaluate the effectiveness of our algorithms by comparing them against the baseline. As shown in Table 4, each Poincaré kernel brings a significant improvement to the baseline network. For example, the simplest Poincaré tangent kernel improves the HM value over the baseline by 6.1, 21.6, 21.9 and 14.1 for SUN, CUB, AWA1 and AWA2, respectively. In addition, the powerful Poincaré radial kernel or Poincaré Laplace kernel continues to improve the representation capacity, again showing the superiority of the kernel design for hyperbolic representations.

To further verify the effectiveness of our approach, we continue to compare our methods to a couple of popular ZSL algorithms, including the state-of-the-art non-generative methods (Zhang & Shi, 2019; Li et al., 2019a). We observe that our Poincaré radial kernel or Poincaré Laplace kernel achieve competitive results to the state-of-the-art methods across four datasets. ZSL is a very challenging task as none of the methods in Table 4 achieved the best performance on HM value across all four datasets. That said, it is not easy to distinguish the overall best model over four dataset. Thus, to establish this objectively, we employ the Friedman test[5] Demšar (2006) to compare the algorithms. As shown in the last column of Table 4, the ranking list clearly shows that our methods with the Poincaré radial kernel and the Poincaré Laplace kernel are the best two options in general for the ZSL task.

5.4 Person Re-identification

Problem setting Person re-identification (re-ID) is an important application in the video/multi-camera surveillance task (Fang et al., 2019, 2021c; Ye et al., 2021). It aims to retrieve correct person images from a gallery dataset for the query person of interest. The goal of training a re-ID machine is to learn an embedding space, where the intra- (or inter-) person variance is minimized (or maximized). The feature extractor is trained by a classification task (see Fig. 5). To be specific, given a person image with associated identity (*i.e.*, y), the network first extracts its appearance representation (*i.e.*, $\boldsymbol{f} \in \mathbb{R}^n$). The a fully connected layer (*i.e.*, $\boldsymbol{W}$) is applied to predict the identity of person and a softmax function is used to normalize the output (*i.e.*, $\boldsymbol{p} = \text{softmax}(\boldsymbol{W}^\top \boldsymbol{f})$). The probability of the person $\boldsymbol{f}$ w.r.t. its label y is denoted by $p(y|\boldsymbol{f}) = \frac{\exp(\langle \boldsymbol{w}^*, \boldsymbol{f}\rangle)}{\sum_j^N \exp(\langle \boldsymbol{w}_j, \boldsymbol{f}\rangle)}$. The training will minimize the

[5] The Friedman test is a non-parametric measure for multiple datasets. It ranks the algorithms for each dataset separately and calculates the average ranks for each dataset as a ranking score.

Table 4 Zero-shot recognition results on SUN, CUB, AWA1 and AWA2 datasets

Model	SUN			CUB			AWA1			AWA2			Friedman test (rank)	
	U	S	HM	U	S	HM	U	S	HM	U	S	HM		
LATEM (Xian et al., 2016)	14.7	28.8	19.5	15.2	57.3	24.0	7.3	71.7	13.3	11.5	77.3	20.0	14.00	(14)
DEVISE (Frome et al., 2013)	16.9	27.4	20.9	23.8	53.0	32.8	13.4	68.7	22.4	17.1	74.7	27.8	12.00	(13)
DEM (Zhang et al., 2017)	20.5	34.3	25.6	19.6	57.9	29.2	32.8	84.7	47.3	30.5	86.4	45.1	11.00	(10)
ALE (Akata et al., 2015a)	21.8	33.1	26.3	23.7	62.8	34.4	16.8	76.1	27.5	14.0	81.8	23.9	11.33	(12)
SP-AEN (Chen et al., 2018)	24.9	38.6	30.3	34.7	70.6	46.6	–	–	–	23.3	**90.9**	37.1	9.67	(9)
CRnet (Zhang & Shi, 2019)	34.1	36.5	35.3	45.5	56.8	50.5	58.1	74.7	65.4	52.6	78.8	63.1	3.25	(6)
Kai *et al*. (Li et al., 2019a)	36.3	42.8	39.3	**47.4**	47.6	47.5	**62.7**	77.0	69.1	**56.4**	81.4	**66.7**	3.67	(3)
Liu *et al*. † (Liu et al., 2020)	37.2	41.6	39.2	45.8	50.2	47.9	59.1	80.8	68.3	52.9	86.7	65.7	4.00	(4)
Baseline	22.8	38.0	28.5	18.6	44.6	26.3	29.8	76.4	42.9	25.5	76.4	38.2	11.00	(10)
Poincaré tangent kernel	29.4	42.0	34.6	40.8	**58.1**	47.9	52.3	85.2	64.8	37.1	88.5	52.3	3.67	(7)
Poincaré RBF kernel	37.0	43.3	39.9	44.6	57.8	50.3	59.0	84.6	69.5	42.9	89.5	57.9	3.00	(4)
Poincaré Laplace kernel	35.1	44.2	39.1	46.2	56.1	50.7	60.7	83.5	70.3	54.1	87.1	**66.7**	2.67	(2)
Poincaré binomial kernel	26.9	43.8	33.3	39.8	56.9	46.8	43.7	**88.9**	58.6	39.8	90.5	55.4	7.67	(8)
Poincaré radial kernel	**38.2**	**44.8**	**41.2**	45.8	57.6	**51.0**	60.2	86.7	**71.1**	48.2	90.3	62.8	2.22	(1)

U and S indicate the accuracy for unseen and seen classes, respectively. HM is the harmonic mean of U and S. The 1st / 2nd best are in "bold" / "(underline)", respectively. † indicates self-implemented algorithm

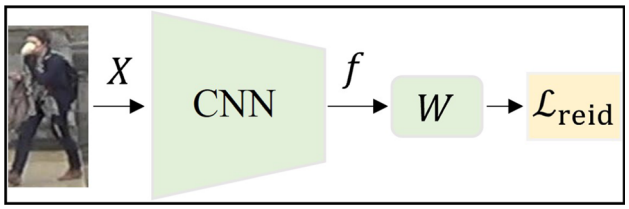


Fig. 5 The pipeline of the deep network for person re-identification. $\boldsymbol{X}$ denotes the input pedestrian images

negative log-probability, as

$$\begin{aligned}\mathcal{L}_{\text{reid}} &= -\log\left(p(y|\boldsymbol{f})\right) \\ &= -\log\left(\frac{\exp(\langle \boldsymbol{w}^*, \boldsymbol{f}\rangle)}{\sum_j^N \exp(\langle \boldsymbol{w}_j, \boldsymbol{f}\rangle)}\right).\end{aligned} \tag{27}$$

The kernelized loss function for $\boldsymbol{f}, \boldsymbol{w} \in \mathbb{D}_c^n$ can further be obtained:

$$\mathcal{L}_{\text{reid}}^{\text{K}} = -\log\left(\frac{g(k(\boldsymbol{w}^*, \boldsymbol{f}))}{\sum_j^N g(k(\boldsymbol{w}_j, \boldsymbol{f}))}\right), \tag{28}$$

where $k(\cdot,\cdot)$ indicates the kernel and $\boldsymbol{w}^*, \boldsymbol{f} \in \mathbb{D}_c^n$. Here, $g(\cdot)$ is exp mapping if $k(\cdot,\cdot)$ is non-exponential type kernels. Otherwise, $g(\cdot)$ is the identity mapping. Since the person images in the test set are unseen during training, we use the penultimate layer of the network as a feature embedding for the person image in the inference phase.

Following the work (Khrulkov et al., 2020), ResNet-50, pre-trained on ImageNet, is employed as a backbone network and we also perform experiments across three dimensions, *i.e*., 32, 64, 128, for the feature representation. Both **Market-1501** (Zheng et al., 2015) and **DukeMTMC-reID** (Ristani et al., 2016) pedestrian datasets are used to evaluate our approaches. We include the statistics of each dataset in the Supplementary Material. We use both mean average precision (mAP) and rank-1 accuracy of cumulative matching characteristic (CMC) to evaluate our algorithms. Different from FSL and ZSL, we use the generalized Poincaré Laplace kernel in the re-ID experiment, as we observe that the generalized Poincaré Laplace kernel achieves fairly good performance compared to the Poincaré Laplace one.

Related work Establishing a highly-discriminative embedding space is the core target for person re-ID task (Zheng et al., 2016; Ye et al., 2021) and many recent works investigate attention mechanisms to locate the discriminative regions within pedestrian images (Li et al., 2018; Fang et al., 2021c; Zhang et al., 2020). In Li et al. (2019), Li *et al*. propose a harmonious attention network, in which a hard attention block and a soft attention block are integrated to attend the informative local and global areas. Works in Wang et al. (2018); Fang et al. (2021c) further develop the full attention framework to preserve the spatial structure information of images. The importance of such structural information is also proved by the consistent regularization over the attention blocks (Zhou et al., 2019). Other auxiliary information, *e.g*., attributes, poses, spatial relations are also adopted to create effective attention mechanisms (Zhang et al., 2020; Tay et al., 2019; Su et al., 2017). Along with the attention mechanism, optimizing over geometry constraints is also studied (Sun

Table 5 Person re-ID results on Market-1501 and DukeMTMC-reID datasets

Model	Dim	Market-1501		DukeMTMC-reID	
		R-1	mAP	R-1	mAP
Euclidean (Khrulkov et al., 2020)	#32	68.0	43.4	57.2	35.7
Hyperbolic (Khrulkov et al., 2020)	#32	75.9	51.9	62.2	39.1
Poincaré tangent kernel	#32	[75.4]	53.3	63.9	42.5
Poincaré RBF kernel	#32	76.0	54.3	**67.3**	**46.3**
g-Poincaré Laplace kernel	#32	78.7	56.3	64.1	40.7
Poincaré binomial kernel	#32	[75.2]	55.0	63.7	44.7
Poincaré radial kernel	#32	**79.6**	**57.8**	66.8	46.1
Euclidean (Khrulkov et al., 2020)	#64	80.5	57.8	68.3	45.5
Hyperbolic (Khrulkov et al., 2020)	#64	84.4	62.7	70.8	48.6
Poincaré tangent kernel	#64	85.8	68.0	**73.9**	54.2
Poincaré RBF kernel	#64	85.2	65.7	73.8	**55.8**
g-Poincaré Laplace kernel	#64	85.4	68.4	73.3	50.6
Poincaré binomial kernel	#64	[83.0]	64.6	71.5	54.0
Poincaré radial kernel	#64	**86.4**	**68.7**	73.6	55.2
Euclidean (Khrulkov et al., 2020)	#128	86.0	67.3	74.1	53.3
Hyperbolic (Khrulkov et al., 2020)	#128	87.8	68.4	76.5	55.4
Poincaré tangent kernel	#128	89.4	74.1	78.6	60.9
Poincaré RBF kernel	#128	88.9	73.5	78.4	62.2
g-Poincaré Laplace kernel	#128	[87.6]	72.4	77.3	59.6
Poincaré binomial kernel	#128	[87.6]	72.0	[75.4]	59.2
Poincaré radial kernel	#128	**90.2**	**74.6**	**79.8**	**63.8**

The value in $\boxed{\cdot}$ denotes the result below the performance in Khrulkov et al. (2020). 1st / 2nd best in "bold" / "(underline)". g-Poincaré Laplace kernel indicates the generalized Poincaré Laplace kernel

et al., 2017; Khrulkov et al., 2020). In SVDNet (Sun et al., 2017), the orthogonality is integrated into the classification layer, thereby decoupling the feature correlations. Modeling the embedding space in the spherical space or hyperbolic space also shows superior properties of the curved geometry (Khrulkov et al., 2020; Hao et al., 2019). The data distribution also contributes to the embeddings, and can be explored via studying the point-to-set distance (Yu et al., 2018) or set-to-set distance (Fang et al., 2021b).

Results We compare the proposed algorithms to methods in Khrulkov et al. (2020). As shown in Table 5, we observe that our algorithms bring positive effects to the retrieval performance on both datasets, especially for the mAP value. In the market-1501 dataset, most of our methods achieve competitive performance compared to Khrulkov et al. (2020), and the Poincaré radial kernel achieves the best performance. For example, the performance gain of R-1 / mAP in 32, 64, 128 dimensions are 3.7 / 5.9, 2.0 / 6.0, 2.4 / 6.2, respectively. However, we also observe that the binomial kernel cannot perform well in different embedding sizes. In the DukeMTMC-reID dataset, our method could outperform its hyperbolic embedding counterpart (Khrulkov et al., 2020) on both R-1 and mAP values. In this dataset, the radial kernel and RBF kernel are the most powerful two kernels. For example, the Poincaré radial kernel improves the R-1 / mAP values over the work (Khrulkov et al., 2020) by 4.6 / 7.0, 2.8 / 6.6 and 3.3 / 8.4 for the dimension of 32, 64 and 128. While the Poincaré RBF kernel improves the values by 4.6 / 7.2, 3.0 / 7.2, and 1.9 / 6.8 for the three dimensions, respectively.

5.5 Deep Metric Learning

Problem setting Similar to the person re-ID task, deep metric learning (DML) is also required to create a latent metric space, which is trained by a popular ranking task using triplet loss (Schroff et al., 2015). Figure 6 demonstrates the pipeline of DML.

The optimizing object of the triplet loss is to minimize the intra-class distance while enlarging the inter-class distance. Given an anchor sample, *i.e.*, $\boldsymbol{X}_i^{\mathrm{a}}$, a possible triplet can be constructed as $\{\boldsymbol{X}_i^{\mathrm{a}}, \boldsymbol{X}_i^{+}, \boldsymbol{X}_i^{-}\}$, where the positive sample $\boldsymbol{X}_i^{+}$ belongs to the same class of the anchor, while the class of the negative sample $\boldsymbol{X}_i^{-}$ is different to that of the anchor. A CNN encodes input images to a latent metric space, with the corresponding feature embeddings, denoted by $\{\boldsymbol{f}_i^{\mathrm{a}}, \boldsymbol{f}_i^{+}, \boldsymbol{f}_i^{-}\}$.

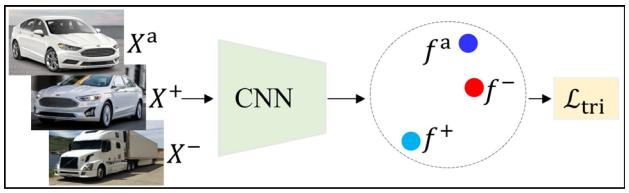


Fig. 6 The pipeline of the network for deep metric learning. The notations $\boldsymbol{X}^{\mathrm{a}}$, $\boldsymbol{X}^{+}$ and $\boldsymbol{X}^{-}$ indicate the anchor sample, positive sample and negative sample, and $\boldsymbol{f}^{\mathrm{a}}$, $\boldsymbol{f}^{+}$ and $\boldsymbol{f}^{-}$ are the associated feature embeddings

Then a triplet loss can be written as:

$$\mathcal{L}_{\mathrm{tri}} = \frac{1}{N_{\mathrm{tri}}} \sum_{i=1}^{N_{\mathrm{tri}}} \left[d_i^{+} - d_i^{-} + \eta \right]_{+}, \tag{29}$$

where $d_i^{+} = \|\boldsymbol{f}_i^{\mathrm{a}} - \boldsymbol{f}_i^{+}\|^2$ and $d_i^{-} = \|\boldsymbol{f}_i^{\mathrm{a}} - \boldsymbol{f}_i^{-}\|^2$. Here, $[y]_{+} = \max(0, y)$ is the hinge loss and $\eta > 0$ is a margin. N_{tri} is the number of triplet in a batch. Its kernelized counterpart is formulated as:

$$\mathcal{L}_{\mathrm{tri}}^{\mathrm{K}} = \frac{1}{N_{\mathrm{tri}}} \sum_{i=1}^{N_{\mathrm{tri}}} \left[-k_i^{+} + k_i^{-} + \eta \right]_{+}, \tag{30}$$

where $k_i^{+} = k(\boldsymbol{f}_i^{\mathrm{a}}, \boldsymbol{f}_i^{+})$ and $k_i^{-} = k(\boldsymbol{f}_i^{\mathrm{a}}, \boldsymbol{f}_i^{-})$. In this study, we set the margin $\eta = 0.3$ for all experiments.

We use Inception-V1 (Szegedy et al., 2015) with batch normalization (Ioffe & Szegedy, 2015) as the feature extractor and evaluate our algorithms on the **Stanford Cars** (CARS-196) dataset (Krause et al., 2013) (see Supplementary Material for its details). Following the common practice in DML, we use normalized mutual information (NMI) and recall@K (R@K) metrics in the evaluation stage.

Related work In general, DML refers to intelligence approaches, which learn data-dependent metric functions (Weinberger & Saul, 2009), and it has been crucial in many computer vision tasks. An early solution is on learning a Mahalanobis metric (Xiang et al., 2008), which embeds the raw data to a Mahalanobis pseudo metric space, thereby inferring the geometrical structure of feature distribution. Modern machine learning techniques study to explicitly optimize the distance metric in the embedding space. In Schroff et al. (2015), triplet loss, proposed by Schroff *et al*., takes into account the relative distance per triplet of samples. The Npair loss makes use of more negatives for an anchor sample (Sohn, 2016). Instead of mining the negative sample in Schroff et al. (2015), the lifted structure loss (Song et al., 2016) samples the negative sample by considering its distance to both the anchor and its positive sample. In addition to the simple Euclidean distance, angular loss learns a scale-invariant similarity metric using the angle at the negative point (Wang et al., 2017). Ustinova *et al*. study the histogram loss, which pushes away the sample with different classes by reducing the overlap between two probability density functions (Ustinova & Lempitsky, 2016).

Table 6 Deep metric learning results on CARS-196 dataset

Model	NMI	R@1	R@2	R@4	R@8
Baseline	56.7	60.1	72.6	81.4	88.9
Poincaré tangent kernel	58.2	62.7	74.8	83.2	90.2
Poincaré RBF kernel	59.1	**63.9**	75.4	83.7	90.4
Poincaré Laplace kernel	58.9	63.1	73.8	82.6	89.3
Poincaré binomial kernel	58.2	62.1	74.3	82.1	89.3
Poincaré radial kernel	**60.4**	63.6	**76.2**	**84.8**	**90.9**

1st / 2nd best in "bold" / "_". g-Poincaré Laplace kernel indicates the generalized Poincaré Laplace kernel

Results Table 6 shows the results in the study of DML. It reveals that our Poincaré kernels can improve the accuracy over the baseline, clearly showing the effectiveness of the kernel methods for hyperbolic representations. Again, in DML, the Poincaré radial kernel achieves the overall best performance. It improves the baseline performance by 3.7 and 3.5 for NMI and R@1 values.

5.6 Knowledge Distillation

Problem setting Knowledge distillation (KD) is an efficient method to train a small student network, under the supervision of a pre-trained larger teacher network (Hinton et al., 2014). Such that the small student network can learn the knowledge from the teacher network and be deployed in the mobile devices. In the teacher-student network, the output of the teacher network acts as ground truth to train a student network (see Fig. 7). For a training image (*e.g.*, $\boldsymbol{X}$), the teacher network generates the prediction scores $\boldsymbol{g} = [g_1, g_2, \ldots, g_N]^{\top}$. Then the student network first extract the feature vector of input image as $\boldsymbol{f} \in \mathbb{R}^n$, and a fully connected layer $\boldsymbol{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_N]$ is used to produce the predication, *i.e.* $\boldsymbol{p} = \mathrm{softmax}(\boldsymbol{W}^{\top}\boldsymbol{f})$ and each p_i is given by:

$$p_i = \frac{\exp(\langle \boldsymbol{w}_i, \boldsymbol{f} \rangle / T)}{\sum_{j=1}^{N} \exp(\langle \boldsymbol{w}_j, \boldsymbol{f} \rangle / T)}, \tag{31}$$

where T is the temperature. Then the KD loss function can be formulated as:

$$\begin{aligned} \mathcal{L}_{\mathrm{kd}} &= -\sum_{i=1}^{N} g_i \log(p_i) \\ &= -\sum_{i=1}^{N} g_i \log\left(\frac{\exp(\langle \boldsymbol{w}_i, \boldsymbol{f} \rangle / T)}{\sum_{j=1}^{N} \exp(\langle \boldsymbol{w}_j, \boldsymbol{f} \rangle / T)} \right). \end{aligned} \tag{32}$$

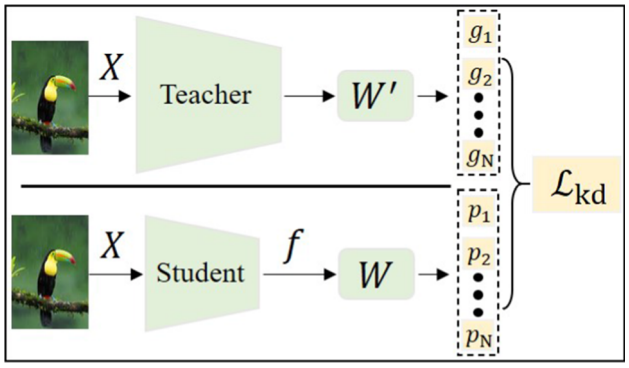


Fig. 7 The pipeline of teacher-student network for knowledge distillation. $\boldsymbol{X}$ denotes the input images

The kernelized KD loss for the hyperbolic representation $\boldsymbol{f} \in \mathbb{D}_c^n$ can be obtained as:

$$\mathcal{L}_{\text{kd}}^{\text{K}} = -\sum_{i=1}^{N} g_i \log\left(\frac{g(k(\boldsymbol{w}_i, \boldsymbol{f})/T)}{\sum_{j=1}^{N} g(k(\boldsymbol{w}_j, \boldsymbol{f})/T)}\right), \tag{33}$$

where $k(\cdot,\cdot)$ indicates the kernel and $\boldsymbol{w}_i, \boldsymbol{f} \in \mathbb{D}_c^n$. Here, $g(\cdot)$ is exp mapping if $k(\cdot,\cdot)$ is non-exponential type kernels. Otherwise, $g(\cdot)$ is the identity mapping. As for the value of temperature T, we stay consistent with the popular choice for $T = 4$ across all experiments (Cho & Hariharan, 2019; Zagoruyko & Komodakis, 2017).

We use the ResNet-20 as a teacher network and a simple 4-layer CNN as a student network. Table 7 illustrates the CNN architectures for teacher and student networks.

We report the results on the **CIFAR-10** and **CIFAR-100** datasets (Krizhevsky, 2009). The details of datasets are summarized in the Supplementary Material. We use the top-1 mean accuracy to evaluate the networks.

Related work Designing an objective that pushes the student network to mimic behaviors from the teacher network is essential in the KD problem with the early attempt being (Ba & Caruana, 2014). In Ba and Caruana (2014), the behavior mimicking is realized by minimizing the L2 distance of predictions. Its extension work by Hinton *et al*. explores the KL-divergence as behavior measurement (Hinton et al., 2014). It also shows that leveraging the predication from a teacher network as a label to supervise the training for a small size student network is better than using the origin one-hot label. Besides behavior in prediction, the student network also learns relational knowledge (*i.e*., distribution or correlation of feature embeddings) from teachers (Park et al., 2019; Peng et al., 2019; Liu et al., 2019). Recent studies also advocate that spatial structure information also matters in KD. Such structure information can be leaned by attention transfer (Zagoruyko & Komodakis, 2017; Wang et al., 2020) or feature similarity preservation (Yim et al., 2017; Tung & Mori, 2019).

Results As shown in Table 8, we can again find that our Poincaré kernels improve the accuracy over the baseline. Specifically, the Poincaré RBF kernel brings the maximum performance gain *i.e*., 3.1, on CIFAR-10, while the Poincaré radial kernel achieves the best performance on CIFAR-100.

5.7 Self-Supervised Learning

Problem setting Self-supervised Learning (SSL) has gained increasing attention in the learning community for its power of learning representations from a large scale of data without manual labeling. We follow the good practice of SimCLR (Chen et al., 2020) for SSL experiments. The pipeline of SimCLR is demonstrated in Fig. 8. For the i-th image in a mini-batch, two different data augmentations are applied, and a network encodes two images to a positive pair of representations, *i.e*., $[\boldsymbol{z}_i, \boldsymbol{z}_i^+]$. In SimCLR, a projection head,[6] denoted by h in Fig. 8, is further used on top of the representations, *i.e*., $\boldsymbol{f}_i = h(\boldsymbol{z}_i)$ and $\boldsymbol{f}_i^+ = h(\boldsymbol{z}_i^+)$. Then the contrastive learning objective can be given by:

$$\mathcal{L}_{\text{cts}} = -\sum_{i=1}^{N} \log \frac{\exp\left(\text{sim}(\boldsymbol{f}_i, \boldsymbol{f}_i^+)/T\right)}{\sum_{j=1}^{N} \exp\left(\text{sim}(\boldsymbol{f}_i, \boldsymbol{f}_j)/T\right)}, \tag{34}$$

where T is the temperature and the $\text{sim}(\cdot,\cdot) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$, $\text{sim}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{\boldsymbol{x}_i^\top \boldsymbol{x}_j}{\|\boldsymbol{x}_i\|\|\boldsymbol{x}_j\|}$.

In contrastive loss, the cosine distance is used as a similarity measure. We can use the proposed kernels to replace sim function in Eq. (34) to kernelize the contrastive loss. The kernelized contrastive loss can be formulated as:

$$\mathcal{L}_{\text{cts}}^{\text{K}} = -\sum_{i=1}^{N} \log \frac{g(k(\boldsymbol{f}_i, \boldsymbol{f}_i^+)/T)}{\sum_{j=1}^{N} g(k(\boldsymbol{f}_i, \boldsymbol{f}_j)/T)}, \tag{35}$$

where $k(\cdot,\cdot)$ indicates the kernel, and $\boldsymbol{f}_i, \boldsymbol{f}_i^+ \in \mathbb{D}_c^n$. Here, $g(\cdot)$ is exp mapping if $k(\cdot,\cdot)$ is non-exponential type kernels. Otherwise, $g(\cdot)$ is the identity mapping. In this study, we set $T = 0.07$ as that in Chen et al. (2020).

We use ResNet-18 as a feature extractor. The protocol of SSL experiments first uses contrastive loss to pre-train the feature extractor. Then the representation quality is evaluated by linear probing. We report the Top-1 test accuracy for SSL on three datasets, *i.e*., **STL-10**, **CIFAR-10** and **CIFAR-100**. The representation power is evaluated by linear probing (Chen et al., 2020). The details of datasets are summarized in the Supplementary Material.

[6] Following SimCLR, the projection head is a 2-layer MLP with ReLU activation (*i.e*., 2048 → 2048 → ReLU → 128).

Table 7 Network architecture for knowledge distillation on CIFAR-10 and CIFAR-100 datasets

Conv layer	Teacher ResNet-20	Student 4-layer CNN
Conv_1	conv, 3×3, 16	conv, 3×3, 16
Conv_2	$\begin{bmatrix}\text{conv}, 1 \times 1, 16\\ \text{conv}, 3 \times 3, 16\end{bmatrix} \times 3$	conv, 3×3, 16
Conv_3	$\begin{bmatrix}\text{conv}, 1 \times 1, 32\\ \text{conv}, 3 \times 3, 32\end{bmatrix} \times 3$	conv, 3×3, 32
Conv_4	$\begin{bmatrix}\text{conv}, 1 \times 1, 64\\ \text{conv}, 3 \times 3, 64\end{bmatrix} \times 3$	conv, 3×3, 64
CIFAR-10 / 100	Global average pooling, 10 / 100-classes, softmax	Global average pooling, 10 / 100-classes, softmax
PNs ($\times 10^{-6}$)	0.272 / 0.278	0.027 / 0.032

PNs indicates the parameter numbers

Table 8 Knowledge distillation results on CIFAR-10 / 100 datasets

Model	CIFAR-10	CIFAR-100
Baseline	80.5	49.9
Poincaré tangent kernel	82.1	50.5
Poincaré RBF kernel	**83.6**	54.4
g-Poincaré Laplace kernel	83.2	53.9
Poincaré binomial kernel	81.6	51.8
Poincaré radial kernel	83.2	**54.9**

1st / 2nd best in "bold" / "(underline)". g-Poincaré Laplace kernel indicates the generalized Poincaré Laplace kernel

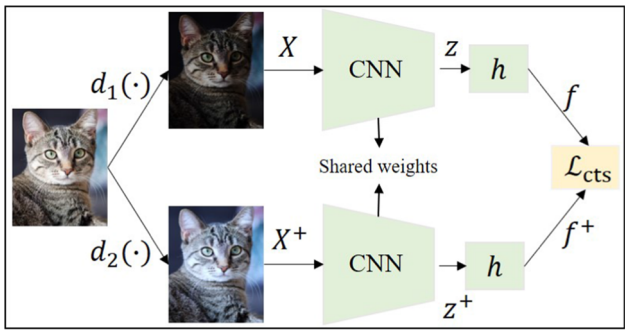


Fig. 8 The pipeline of SimCLR using the contrastive learning scheme. d_1 and d_2 present two different data augmentations, applied to the same image, thereby resulting in a positive pair, *i.e.*, $\boldsymbol{X}$ and $\boldsymbol{X}^+$. On top of the CNN architecture, a projection head h is used to project the feature embedding to a new space for contrastive learning

Related work Learning via contrastive scheme is a natural idea to create image representations, and significantly improves the representation power of SSL (Hjelm et al., 2019; Chen et al., 2020; He et al., 2020; Chen & He, 2021). The success of contrastive learning in SSL heavily relies on methods to build positive samples for each image. In DIM, the positive pair is defined as the global context feature and the local patch feature in images (Hjelm et al., 2019). SimCLR is a simple yet effective framework, where various data augmentations are applied to an image as a positive pair, for SSL (Chen et al., 2020). MoCo uses a momentum-updated encoder to build positive pairs (He et al., 2020). In the Siamese networks, the positive pair is created by simply adding a projection head to the origin feature (Chen & He, 2021).

Table 9 Self-supervised Learning results on STL-10 / CIFAR-10 / 100 datasets

Model	STL-10	CIFAR-10	CIFAR-100
SimCLR	73.62	69.37	40.88
Poincaré tangent kernel	74.23	**71.08**	42.95
Poincaré RBF kernel	74.57	70.43	43.68
g-Poincaré Laplace kernel	72.68	64.21	39.08
Poincaré binomial kernel	70.33	66.72	39.20
Poincaré radial kernel	**74.97**	70.92	**44.03**

The value in $\boxed{\cdot}$ denotes the result below the baseline network. 1st / 2nd best in "bold" / "_". g-Poincaré Laplace kernel indicates the generalized Poincaré Laplace kernel

Results Table 9 shows the empirical results of SSL. We can observe that the SSL is a very challenging task as only proper kernels can improve the performance over the baseline. To be specific, the Poincaré tangent kernel, Poincaré RBF kernel, and Poincaré radial kernel consistently improve representation power on three datasets. The Poincaré radial kernel achieves the maximum performance gain on STL-10 and CIFAR-100, reading as 1.35 and 3.15 respectively, while Poincaré tangent kernel brings the maximum performance gain on CIFAR-10, with the value of 1.71.

However, the Poincaré Laplace kernel and Poincaré, which work well in other embedding learning applications, *e.g.*, FSL, fail to work in SSL. They even degrade the performance of the baseline. This situation indicates that not all kernels with fixed formulation increase the discrimination of features, and it is essential to develop the data-adaptive kernels, like the Poincaré radial kernel in this paper.

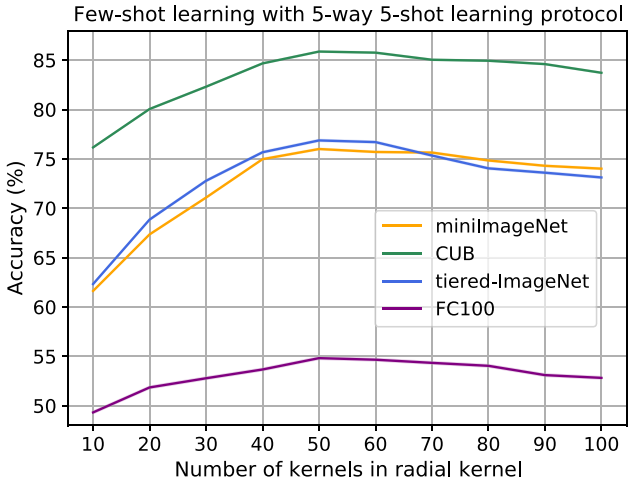


Fig. 9 Evaluation of the number M in Poincaré radial kernel

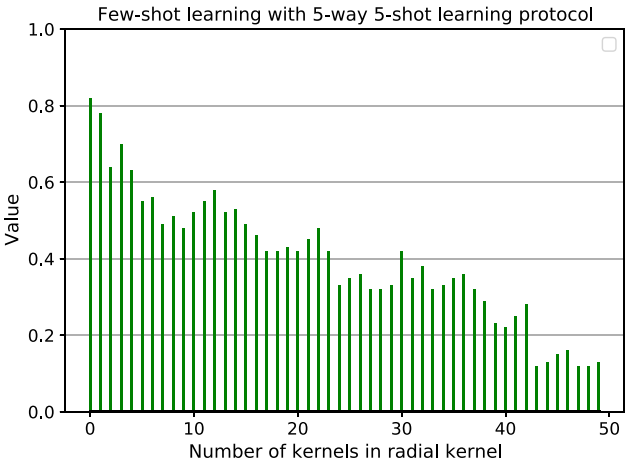


Fig. 10 Visualization of the value of weights, *i.e.*, a_m, in the Poincaré radial kernel

5.8 Further Studies

Number of kernels in the Poincaré radial kernel The formulation of Poincaré radial kernel in Eq. (20) follows the multiple kernel learning (MKL) scheme, with a good property that each kernel component/weight can be learned depending on the dataset. In this experiment, we study the number of kernels in Eq. (20). We consider the few-shot learning task under 5-way 5-shot setting and use Conv-4 as the backbone network. Four datasets, the *mini*ImageNet, CUB, *tired*-ImageNet and FC100, are used in this study. An observation is made in Fig. 9 that when $M = 50$, the Poincaré radial kernel attains better performance consistently across all datasets.

Our study above demonstrates that the proposed Poincaré radial kernel achieves the best performance when $M = 50$. To gain insights into the behavior of the weights, we visualize the weight values a_m in Fig. 10. The plot reveals a long-tailed distribution of weight values with respect to the orders of the kernel components. We note that the higher-order components, despite having smaller weights, play a crucial role in enhancing the accuracy of the similarity measure, thus contributing to the overall performance of the method. Additional visualizations of the weight values for different M are provided in the supplementary material.

Activation function in the Poincaré radial kernel As discussed in Sect. 4.5, an activation function is required to apply to the learned weights in Eq. (20), leading the proposed Poincaré radial kernel being pd. In this context, we study the impact of several candidates of activation functions, e.g., ReLU(·), softmax(·) and sigmoid(·), and report the results in Table 10. This analysis is studied on the few-shot learning task with the *mini*ImageNet dataset. Conv-4 is used as the backbone network. Table 10 shows that the proposed Poincaré radial kernel can bring improvement over the baseline with each of the activation functions, showing that the proposed Poincaré radial kernel can work properly once the pd property is satisfied. Among the activation functions, the sigmoid(·) activation attains a better performance than other activation functions. This observation enables us to choose sigmoid, which normalizes the weights between 0 and 1, as the activation function in the Poincaré radial kernel.

Table 10 Effective of different activation functions

Model	*mini*ImageNet	
	5-way 1-shot	5-way 5-shot
Baseline	54.43 ± 0.21	74.81 ± 0.16
ReLU(·)	56.51 ± 0.19	75.62 ± 0.15
softmax(·)	55.88 ± 0.19	75.18 ± 0.16
sigmoid(·)	$\mathbf{57.28 \pm 0.18}$	$\mathbf{76.82 \pm 0.15}$

This study is conducted on the few-shot learning (FSL) task with the *mini*ImageNet dataset. Conv-4 is used as the backbone network. The best result is in "bold"

Indefinite kernel vs. Positive definite kernel To the best of our knowledge, our work is the first to develop pd kernels in hyperbolic spaces. That said, indefinite hyperbolic kernels are developed in Cho et al. (2019). As for the indefinite kernel, we use the Minkowski inner product kernel, presented in Cho et al. (2019) (see Supplementary Material for details). We have evaluated the performance of our pd kernels and the indefinite kernel for the task of 5-way 5-shot learning across the *mini*ImageNet, CUB, *tired*-ImageNet and FC100 datasets. Figure 11 shows that the performance attained by the indefinite kernel does not match that of pd kernels, clearly showing the potential of pd kernels for hyperbolic representations.

Euclidean spaces vs. Hyperbolic spaces One may wonder how useful the hyperbolic spaces are and their kernels in comparison to simple Euclidean kernels. In the end, the Poincaré

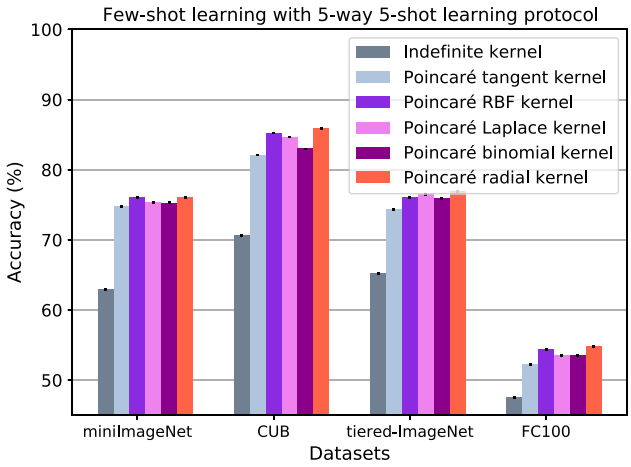


Fig. 11 The performance comparison between the indefinite kernel and pd kernels for hyperbolic representations

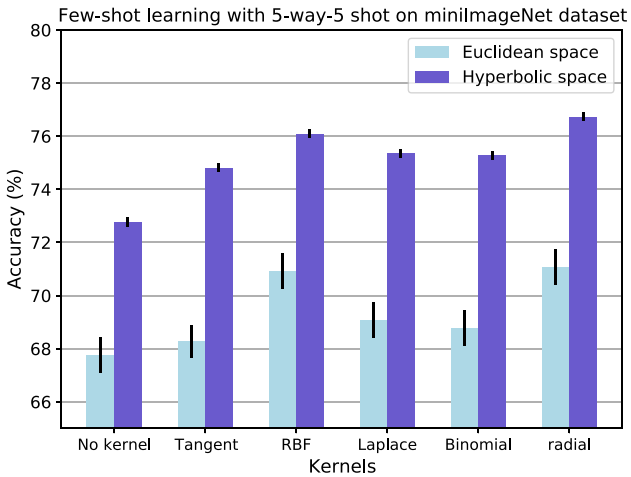


Fig. 12 The performance comparison for kernels on Euclidean spaces and hyperbolic spaces

ball is embedded in n-dimensional Euclidean spaces and hence conventional kernels can be applied seamlessly. In Fig. 12, we compare the proposed kernels against their Euclidean counterparts again on the task of few-shot learning using the *mini*ImageNet dataset. We observe: (1) the kernel machines in both Euclidean spaces and hyperbolic spaces bring performance gain to the deep neural network. (2) The proposed hyperbolic kernels can outperform the vanilla Euclidean kernels significantly, again showing the reasonable design of the proposed kernels.

Remark 5 In this section, extensive experiments are conducted to evaluate the superiority of the proposed Poincaré kernels, as well as the usage of hyperbolic geometry. This can also be justified by the empirical observation that various applications can benefit from a high curvature (*i.e.*, c). For example, in the person re-identification task, the curvature of the Poincaré ball is 10^{-2} in our algorithms, while the work in Khrulkov et al. (2020) sets it to 10^{-5}, which makes the Poincaré ball very flat. We believe our contribution is necessary in the field of deep manifold learning.

6 Conclusion

This paper proposes a family of positive definite kernels to embed hyperbolic representations in Hilbert spaces. To define such kernels, we leverage the identity tangent space of the Poincaré ball and further define valid positive definite kernels in identity tangent spaces. The proposed kernels include powerful universal ones (*i.e.*, the Poincaré RBF kernel, the Poincaré Laplace kernel, the Poincaré binomial kernel, and the Poincaré radial kernel). We evaluate the effectiveness of the kernels in a variety of challenging applications, such as few-shot learning, zero-shot learning, person re-identification, deep metric learning, knowledge distillation and self-supervised learning, and the empirical results have shown positive results for embedding learning via the kernels in hyperbolic spaces. Future works include exploiting the proposed kernels to other applications (*i.e.*, natural language processing and graph neural networks). In addition, we have found that the effectiveness of the kernel is data-dependent and we want to develop a rule for choosing the right kernel for a given data.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11263-023-01834-6.

References

Absil, P. A., Mahony, R., & Sepulchre, R. (2007). *Optimization algorithms on matrix manifolds*. Princeton University Press.

Akata, Z., Perronnin, F., Harchaoui, Z., & Schmid, C. (2015). Label-embedding for image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 5*, 1425–1438.

Akata, Z., Reed, S., Walter, D., Lee, H., & Schiele, B. (2015). Evaluation of output embeddings for fine-grained image classification. In *IEEE computer vision and pattern recognition* (pp. 2927–2936).

Ba, J., & Caruana, R. (2014). Do deep nets really need to be deep? In *Advances in neural information processing systems* (pp. 2654–2662).

Berg, C., Christensen, J. P. R., & Ressel, P. (1984). *Harmonic analysis on semigroups*. Springer.

Chen, J., Qin, J., Shen, Y., Liu, L., Zhu, F., & Shao, L. (2020). Learning attentive and hierarchical representations for 3D shape recognition. In *European conference on computer vision* (pp. 105–122).

Chen, L., Zhang, H., Xiao, J., Liu, W., & Chang, S. F. (2018). Zero-shot visual recognition using semantics-preserving adversarial embedding networks. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 1043–1052).

Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020). A simple framework for contrastive learning of visual representations. In *The 36th international conference on machine learning* (pp. 1597–1607).

Chen, W.Y., Liu, Y.C., Kira, Z., Wang, Y. C., & Huang, J. B. (2019). A closer look at few-shot classification. In *International conference on learning representations* (pp. 1–11).

Chen, X., & He, K. (2021). Exploring simple siamese representation learning. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 15750–15758).

Cho, H., DeMeo, B., Peng, J., & Berger, B. (2019). Large-margin classification in hyperbolic space. In *The 36th international conference on machine learning* (pp. 1832–1840).

Cho, J. H., & Hariharan, B. (2019). On the efficacy of knowledge distillation. In *IEEE/CVF international conference on computer vision* (pp. 1–11).

Christmann, A., & Steinwart, I. (2008). *Support vector machines*. Springer.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research, 5*, 1–30.

Deng, J., Dong, W., Li, R. S. L. J., Li, K., & Li, F. F. (2009). Imagenet: A large-scale hierarchical image database. In *IEEE conference on computer vision and pattern recognition* (pp. 248–255).

Domingos, P. (2020). Every model learned by gradient descent is approximately a kernel machine. arXiv:2012.00152.

Fang, P., Harandi, M., & Petersson, L. (2021). Kernel methods in hyperbolic spaces. In *IEEE/CVF international conference on computer vision* (pp. 10665–10674).

Fang, P., Ji, P., Petersson, L., & Harandi, M. (2021). Set augmented triplet loss for video person re-identification. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision (WACV)* (pp. 464–473).

Fang, P., Zhou, J., Roy, S. K., Ji, P., Petersson, L., & Harandi, M. (2021). Attention in attention networks for person retrieval. In *IEEE transactions on pattern analysis and machine intelligence* (pp. 4626–4641).

Fang, P., Zhou, J., Roy, S.K., Petersson, L., & Harandi, M. (2019). Bilinear attention networks for person retrieval. In *IEEE/CVF international conference on computer vision* (pp. 8030–8039).

Feragen, A., & Hauberg, S. (2016). Open problem: Kernel methods on manifolds and metric spaces. What is the probability of a positive definite geodesic exponential kernel? In *Conference on learning theory* (pp. 1647–1650).

Feragen, A., Lauze, F., & Hauberg, S. (2015). Geodesic exponential kernels: When curvature and linearity conflict. In *IEEE conference on computer vision and pattern recognition* (pp. 3032–3042).

Finn, C., Abbeel, P., & Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *The 34th international conference on machine learning* (pp. 1126–1135).

Frome, A., Corrado, G. S., Shlens, J., Bengio, S., Dean, J., Ranzato, M. A., & Mikolov, T. (2013). Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems* (pp. 2121–2129).

Ganea, O. E., Bécigneul, G., & Hofmann, T. (2018). Hyperbolic neural networks. In *Advances in neural information processing systems* (pp. 5345–5355).

Gretton, A., Borgwardt, K. M., Rasch, M. J., Scholköpf, B., & Smola, A. (2012). A kernel two-sample test. *Journal of Machine Learning Research, 5*, 723–773.

Gu, A., Sala, F., Gunel, B., & Ré, C. (2019). Learning mixed-curvature representations in product spaces. In *International conference on learning representations* (pp. 1–11).

Gulcehre, C., Denil, M., Malinowski, M., Razavi, A., Pascanu, R., Hermann, K. M., Battaglia, P., Bapst, V., Raposo, D., Santoro, A., & de Freitas, N. (2019). Hyperbolic attention networks. In *International conference on learning representations* (pp. 1–11).

Hamann, M. (2011) On the tree-likeness of hyperbolic spaces. arXiv:1105.3925.

Hao, Y., Wang, N., Li, J., & Gao, X. (2019). Hsme: Hypersphere manifold embedding for visible thermal person re-identification. In *The 33rd AAAI conference on artificial intelligence* (pp. 8385–8392).

Harandi, M. T., Salzmann, M., Jayasumana, S., Hartley, R., & Li, H. (2014). Expanding the family of grassmannian kernels: An embedding perspective. In *European conference on computer vision* (pp. 408–423).

He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 9729–9738).

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *IEEE conference on computer vision and pattern recognition* (pp. 770—778).

Hinton, G., Vinyals, O., & Dean, J. (2014). Distilling the knowledge in a neural network. In *Advances in neural information processing systems deep learning workshop*.

Hjelm, R. D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Bachman, P., Trischler, A., & Bengio, Y. (2019). Learning deep representations by mutual information estimation and maximization. In *The international conference on learning representations* (pp. 1–14).

Hofmann, T., Scholkopf, B., & Smola, A. J. (2008). Kernel methods in machine learning. *The Annals of Statistics, 5*, 1171–1220.

Hong, J., Fang, P., Li, W., Zhang, T., Simon, C., Harandi, M., & Petersson, L. (2021). Reinforced attention for few-shot learning and beyond. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 913–923).

Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating deep network training by reducing internal covariate shift. In *The 32nd international conference on machine learning* (pp. 448–456).

Jacot, A., Gabriel, F., & Hongler, C. (2018). Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems* (pp. 8571–8580).

Jayasumana, S., Hartley, R., Salzmann, M., Li, H., & Harandi, M. (2013). Kernel methods on the riemannian manifold of symmetric positive definite matrices. In *IEEE conference on computer vision and pattern recognition* (pp. 73–80).

Jayasumana, S., Hartley, R., Salzmann, M., Li, H., & Harandi, M. (2014). Optimizing over radial kernels on compact manifolds. In *IEEE conference on computer vision and pattern recognition* (pp. 3802–3809).

Jayasumana, S., Hartley, R., Salzmann, M., Li, H., & Harandi, M. (2015). Kernel methods on Riemannian manifolds with gaussian RBF kernels. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 5*, 2464–2477.

Jayasumana, S., Ramalingam, S., & Kumar, S. (2021). Kernelized classification in deep networks. arXiv:2012.09607v2.

Karcher, H. (1977). Riemannian center of mass and mollifier smoothing. *Communications on Pure and Applied Mathematics, 6*, 509–541.

Khrulkov, V., Mirvakhabova, L., Ustinova, E., Oseledets, I., & Lempitsky, V. (2020). Hyperbolic image embeddings. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 6418–6428).

Krause, J., Stark, M., Deng, J., & Fei-Fei, L. (2013). 3D object representations for fine-grained categorization. In *IEEE international conference on computer vision workshops* (pp. 554–561).

Krizhevsky, A. (2009). Learning multiple layers of features from tiny images. In Technical report.

Lampert, C. H., Nickisch, H., & Harmeling, S. (2013). Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 6*, 453–465.

Lanckriet, G. R. G., Cristianini, N., Bartlett, P., Ghaoui, L. E., & Jordan, M. I. (2004). Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research., 6*, 896.

Le, T., & Yamada, M. (2018). Persistence fisher kernel: A Riemannian manifold kernel for persistence diagrams. In *Advances in neural information processing systems* (pp. 10007–10018).

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. In *Proceedings of the IEEE* (pp. 2278—2324).

Li, K., Min, M. R., & Fu, Y. (2019). Rethinking zero-shot learning: A conditional visual classification perspective. In *IEEE/CVF international conference on computer vision* (pp. 3583–3592).

Li, W., Wang, L., Xu, J., Huo, J., Yang, G., & Luo, J. (2019). Revisiting local descriptor based image-to-class measure for few-shot learning. In *IEEE conference on computer vision and pattern recognition* (pp. 7260–7268).

Li, W., Zhu, X., & Gong, S. (2018). Harmonious attention network for person re-identification. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 2285–2294).

Li, W., Zhu, X., & Gong, S. (2019). Scalable person re-identification by harmonious attention. *International Journal of Computer Vision, 5*, 1635–1653.

Liu, Q., Nickel, M., & Kiela, D. (2019). Hyperbolic graph neural networks. In *Advances in neural information processing systems* (pp. 8230–8241).

Liu, S., Chen, J., Pan, L., Ngo, C. W., Chua, T. S., & Jiang, Y. G. (2020). Hyperbolic visual embedding learning for zero-shot recognition. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 9273–9281).

Liu, W., Wen, Y., Yu, Z., Li, M., Raj, B., & Song, L. (2017). Sphereface: Deep hypersphere embedding for face recognition. In *IEEE conference on computer vision and pattern recognition* (pp. 212–220).

Liu, Y., Cao, J., Yuan, B. L. C., Hu, W., Li, Y., & Duan, Y. (2019). Knowledge distillation via instance relationship graph. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 7096–7104).

Lou, A., Katsman, I., Jiang, Q., Belongie, S., Lim, S. N., & Sa, C. D. (2020). Differentiating through the fréchet mean. In *The 37th international conference on machine learning* (pp. 6393–6403).

Meng, Y., Huang, J., Wang, G., Zhang, C., Zhuang, H., Kaplan, L., & Han, J. (2019). Spherical text embedding. In *Advances in neural information processing systems* (pp. 8208–8217).

Micchelli, C. A., Xu, Y., & Zhang, H. (2006). Universal kernels. *Journal of Machine Learning Research, 93*, 2651–2667.

Ong, C. S., Mary, X., Canu, S., & Smola, A. J. (2004). Learning with non-positive kernels. In *The 21st international conference on machine learning* (pp. 1–8).

Oreshkin, B., Rodríguez López, P., & Lacoste, A. (2018). Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in neural information processing systems* (pp. 721–731).

Park, W., Kim, D., Lu, Y., & Cho, M. (2019). Relational knowledge distillation. In *IEEE conference on computer vision and pattern recognition* (pp. 3967–3976).

Patterson, G., & Hays, J. (2012). Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *IEEE conference on computer vision and pattern recognition* (pp. 2751–2758).

Peng, B., Jin, X., Liu, J., Li, D., Wu, Y., Liu, Y., Zhou, S., & Zhang, Z. (2019). Correlation congruence for knowledge distillation. In *IEEE/CVF international conference on computer vision* (pp. 5007–5016).

Rakotomamonjy, A., Bach, F. R., Canu, S., & Grandvalet, Y. (2008). SimpleMKL. *Journal of Machine Learning Research, 6*, 214.

Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J. B., Larochelle, H., & Zemel, R. S. (2018). Meta-learning for semi-supervised few-shot classification. In *International conference on learning representations* (pp. 1–11).

Ristani, E., Solera, F., Zou, R., Cucchiara, R., & Tomasi, C. (2016). Performance measures and a data set for multi-target, multi-camera tracking. In *European conference on computer vision workshop on benchmarking multi-target tracking* (pp. 17–35).

Rodríguez, P., Laradji, I., Drouin, A., & Lacoste, A. (2020). Embedding propagation: Smoother manifold for few-shot classification. In *European conference on computer vision* (pp. 121–138).

Schroff, F., Kalenichenko, D., & Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In *IEEE conference on computer vision and pattern recognition* (pp. 815–823).

Simon, C., Koniusz, P., & Harandi, M. (2021). On learning the geodesic path for incremental learning. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 1591–1600).

Simon, C., Koniusz, P., Nock, R., & Harandi, M. (2020). Adaptive subspaces for few-shot learning. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 4136–4145).

Skopek, O., Ganea, O. E., & Bécigneul, G. (2020). Mixed-curvature variational autoencoders. In *International conference on learning representations* (pp. 1–11).

Snell, J., Swersky, K., & Zemel, R. (2017). Prototypical networks for few-shot learning. In *Advances in neural information processing systems* (pp. 4077–4087).

Sohn, K. (2016). Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems* (vol. 29, pp. 1857–1865).

Song, H. O., Xiang, Y., Jegelka, S., & Savarese, S. (2016). Deep metric learning via lifted structured feature embedding. In *IEEE conference on computer vision and pattern recognition* (pp. 4004–4012).

Su, C., Li, J., Zhang, S., Xing, J., Gao, W., & Tian, Q. (2017). Pose-driven deep convolutional model for person re-identification. In *IEEE/CVF international conference on computer vision* (pp. 3960–3969).

Sun, Y., Zheng, L., Deng, W., & Wang, S. (2017). Svdnet for pedestrian retrieval. In *IEEE international conference on computer vision* (pp. 3800–3808).

Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P. H., & Hospedales, T. M. (2018). Learning to compare: Relation network for few-shot learning. In *IEEE conference on computer vision and pattern recognition* (pp. 1199–1208).

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. In *IEEE conference on computer vision and pattern recognition* (pp. 1–9).

Tay, C. P., Roy, S., & Yap, K. H. (2019). Aanet: Attribute attention network for person re-identifications. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 7134–7143).

Tran, L. V., Tay, Y., Zhang, S., Cong, G., & Li, X. (2020). Hyperml: A boosting metric learning approach in hyperbolic space for recommender systems. In *The 13th international conference on web search and data mining*.

Tung, F., & Mori, G. (2019). Similarity-preserving knowledge distillation. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 1365–1374).

Ustinova, E., & Lempitsky, V. (2016). Learning deep embeddings with histogram loss. In *Advances in neural information processing systems* (vol. 29, pp. 4170–4178).

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., & Wierstra, D. (2016). Matching networks for one shot learning. In *Advances in neural information processing systems* (pp. 3630–3638).

Wah, C., Branson, S., Welinder, P., Perona, P., & Belongie, S. (2011). The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology.

Wang, C., Zhang, Q., Huang, C., Liu, W., & Wang, X. (2018). Mancs: A multi-task attentional network with curriculum sampling for person re-identification. In *European conference on computer vision* (pp. 384–400).

Wang, J., Zhou, F., Wen, S., Liu, X., & Lin, Y. (2017). Deep metric learning with angular loss. In *IEEE international conference on computer vision* (pp. 2612–2620).

Wang, K., Gao, X., Zhao, Y., Li, X., Dou, D., & Xu, C. Z. (2020). Pay attention to features, transfer learn faster CNNS. In *International conference on learning representations* (pp. 1–14).

Wang, T., Zhang, L., & Hu, W. (2021). Bridging deep and multiple kernel learning: A review. *Information Fusion., 5*, 698.

Weinberger, K. Q., & Saul, L. K. (2009). Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research, 6*, 207–244.

Wu, Z., Efros, A. A., & Yu, S. (2018). Improving generalization via scalable neighborhood component analysis. In *European conference on computer vision* (pp. 712–728).

Xian, Y., Akata, Z., Sharma, G., Nguyen, Q., Hein, M., & Schiele, B. (2016). Latent embeddings for zero-shot classification. In *IEEE conference on computer vision and pattern recognition* (pp. 69–77).

Xiang, S., Nie, F., & Zhang, C. (2008). Learning a mahalanobis distance metric for data clustering and classification. *Pattern Recognition, 7*, 3600–3612.

Xu, C., Fu, Y., Liu, C., Wang, C., Li, J., Huang, F., Zhang, L., & Xue, X. (2021). Learning dynamic alignment via meta-filter for few-shot learning. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 5182–5191).

Ye, H. J., Hu, H., Zhan, D. C., & Sha, F. (2020). Few-shot learning via embedding adaptation with set-to-set functions. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 8808–8817).

Ye, M., Shen, J., Lin, G., Xiang, T., Shao, L., & Hoi, S. C. H. (2021). Deep learning for person re-identification: A survey and outlook. In *IEEE transactions on pattern analysis and machine intelligence* (pp. 2872–2893).

Yim, J., Joo, D., Bae, J., & Kim, J. (2017). A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *IEEE conference on computer vision and pattern recognition* (pp. 4133–4141).

Yu, R., Dou, Z., Bai, S., Zhang, Z., Xu, Y., & Bai, X. (2018). Hard-aware point-to-set deep metric for person re-identification. In *European conference on computer vision* (pp. 196–212).

Zagoruyko, S., & Komodakis, N. (2017). Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *International conference on learning representations*.

Zhang, C., Cai, Y., Lin, G., & Shen, C. (2020). Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 12203–12213).

Zhang, F., & Shi, G. (2019). Co-representation network for generalized zero-shot learning. In *The 36th international conference on machine learning* (pp. 7434–7443).

Zhang, L., Xiang, T., & Gong, S. (2017). Learning a deep embedding model for zero-shot learning. In *IEEE conference on computer vision and pattern recognition* (pp. 2021–2030).

Zhang, Z., Lan, C., Zeng, W., Jin, X., & Chen, Z. (2020). Relation-aware global attention for person re-identification. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 3186–3195).

Zhang, Z., & Saligrama, V. (2015). Zero-shot learning via semantic similarity embedding. In IEEE/CVF International Conference on Computer Vision, pp. 4166–4174.

Zheng, L., Shen, L., Tian, L., Wang, S., Wang, J., & Tian, Q. (2015). Scalable person re-identification: A benchmark. In *IEEE international conference on computer vision* (pp. 1116–1124).

Zheng, L., Yang, Y., & Hauptmann, A. G. (2016). Person re-identification: Past, present and future. ArXiv:1610.02984 [cs.CV]

Zhou, S., Wang, F., Huang, Z., & Wang, J. (2019). Discriminative feature learning with consistent attention regularization for person re-identification. In *IEEE/CVF international conference on computer vision* (pp. 8040–8049).