The Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI-22)

Adaptive Poincaré Point to Set Distance for Few-Shot Classification

Rongkai Ma[1], Pengfei Fang[2,3*], Tom Drummond[4], Mehrtash Harandi[1,3]

[1]Monash University
[2]The Australian National University
[3]DATA61-CSIRO, Australia
[4]The University of Melbourne
{rongkai.ma, mehrtash.harandi}@monash.edu, pengfei.fang@anu.edu.au, tom.drummond@unimelb.edu.au

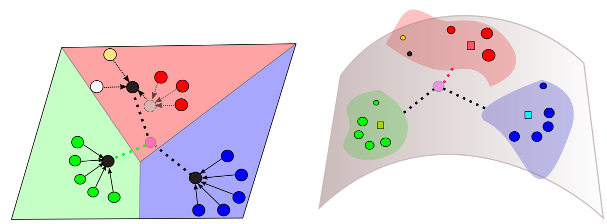
Abstract

Learning and generalizing from limited examples, *i.e.*, few-shot learning, is of core importance to many real-world vision applications. A principal way of achieving few-shot learning is to realize an embedding where samples from different classes are distinctive. Recent studies suggest that embedding via hyperbolic geometry enjoys low distortion for hierarchical and structured data, making it suitable for few-shot learning. In this paper, we propose to learn a context-aware hyperbolic metric to characterize the distance between a point and a set associated with a learned set to set distance. To this end, we formulate the metric as a weighted sum on the tangent bundle of the hyperbolic space and develop a mechanism to obtain the weights adaptively, based on the constellation of the points. This not only makes the metric local but also dependent on the task in hand, meaning that the metric will adapt depending on the samples that it compares. We empirically show that such metric yields robustness in the presence of outliers and achieves a tangible improvement over baseline models. This includes the state-of-the-art results on five popular few-shot classification benchmarks, namely *mini*-ImageNet, *tiered*-ImageNet, Caltech-UCSD Birds-200-2011 (CUB), CIFAR-FS, and FC100.

Introduction

In the modern context of machine learning, deep neural networks (DNNs) have enjoyed enormous success by leveraging the rich availability of labeled data for supervised training. Despite this, deep supervised learning is primarily limited in terms of scaling towards unseen samples due to the high cost of acquiring large amounts of labeled data. This is in clear contrast to how humans learn, where in many cases, only a handful of training examples are sufficient for generalizing towards unseen samples. Few-Shot Learning (FSL) addresses this critical problem through the development of algorithms that can learn using limited data (Finn, Abbeel, and Levine 2017; Nichol, Achiam, and Schulman 2018; Snell, Swersky, and Zemel 2017; Sung et al. 2018; Vinyals et al. 2016; Ye et al. 2020; Hong et al. 2021; Wang et al. 2020).

Performing FSL well is essential towards creating robust frameworks that can learn with the efficiency of humans.



(a) (b)

Figure 1: (a): For a query sample in "red" class, outliers (*i.e.*, yellow and white circles) drag the prototype (*i.e.*, the black circle) far away from the real cluster center in the embedding space such that the nearest neighbor classifier mis-classifies the query point into "green" class. (b): Our method computes an adaptive point to set distance on the manifold, which is more robust to outliers than prototypes. Best viewed in color.

In many cases, FSL methods deem to learn an embedding space to distinguish samples from different classes. Therein, the embedding space is a multidimensional Euclidean space and is realized via a deep neural network.

Employing hyperbolic geometry to encode data has been shown rewarding, as the volume of space expands exponentially (Ganea, Bécigneul, and Hofmann 2018; Khrulkov et al. 2020). Recent works have shown that a hierarchical structure exists within visual datasets and that the use of hyperbolic embeddings can yield significant improvements over Euclidean embeddings (Khrulkov et al. 2020; Fang, Harandi, and Petersson 2021).

Most existing FSL solutions learn a metric through comparing the distance between a query sample and the class prototypes, often modeled as the mean embeddings of each class. However, this does not take the adverse effects of outliers and noises into consideration (Sun et al. 2019). This severely limits the representation power of embedding-based methods since the outliers may drag the prototype away from the true center of the cluster (see Fig.1(a)). For a more robust approach, we require an adaptive metric, which can faithfully capture the distribution per class, while being robust to outliers and other nuances in data (Fig. 1(b)).

With this in mind, we propose learning a context-

*Corresponding author.

aware hyperbolic metric that characterizes the point to set (dis)similarities. This is achieved through employing a Poincaré ball to model hyperbolic spaces and casting the (dis)similarity as a weighted-sum between a query and a class that is learned adaptively. In doing so, each sample (from the support and query sets) is modeled by a set itself (*i.e.*, a feature map). Therefore, we propose to make use of pairwise distances between elements of two sets, along with a refinement mechanism to disregard uninformative parts of the feature maps. This leads to a flexible and robust framework for the FSL tasks. We summarize our contributions as follows:

- We propose a novel adaptive Poincaré point to set (APP2S) distance metric for the FSL task.
- We further design a mechanism to produce a weight, dependent on the constellation of the point, for our APP2S metric.
- We conduct extensive experiments across five FSL benchmarks to evaluate the effectiveness of the proposed method.
- We further study the robustness of our method, which shows our method is robust against the outliers compared to competing baselines.

Preliminaries

In what follows, we use $\mathbb{R}^n$ and $\mathbb{R}^{m\times n}$ to denote the n-dimensional Euclidean space and space of $m \times n$ real matrices, respectively. The n-dimensional hyperbolic space is denoted by $\mathbb{H}^n_c$. The $\operatorname{arctanh} : (-1, 1) \to \mathbb{R}, \operatorname{arctanh}(x) = \frac{1}{2}\ln(\frac{1+x}{1-x}), |x| < 1$ refers to the inverse hyperbolic tangent function. The vectors and matrices (or 3-D tensors) are denoted by bold lower-case letters and bold upper-case letters throughout the paper.

Riemannian Geometry

In this section, we will give a brief recap of Riemannian geometry. A manifold, denoted by $\mathcal{M}$, is a curved surface, which locally resembles the Euclidean space. The tangent space at $\boldsymbol{x} \in \mathcal{M}$ is denoted by $T_{\boldsymbol{x}}\mathcal{M}$. It contains all possible vectors passing through point $\boldsymbol{x}$ tangentially. On the manifold, the shortest path connecting two points is a geodesic, and its length is used to measure the distances on the manifold.

Hyperbolic Space

Hyperbolic spaces are Riemannian manifolds with constant negative curvature and can be studied using the Poincaré ball model (Ganea, Bécigneul, and Hofmann 2018; Khrulkov et al. 2020). The Poincaré ball $(\mathbb{D}^n_c, g^c)$ is a smooth n-dimensional manifold identified by satisfying $\mathbb{D}^n_c = \{\boldsymbol{x} \in \mathbb{R}^n : c\|\boldsymbol{x}\| < 1, c \geqslant 0\}$[1], where c is the absolute value of the curvature for a Poincaré ball, while the real curvature value is $-c$. The Riemannian metric g^c at $\boldsymbol{x}$ is defined as $g^c = {\lambda^c_{\boldsymbol{x}}}^2 g^E$, where g^E is the Euclidean metric tensor and $\lambda^c_{\boldsymbol{x}}$ is the conformal factor, defined as:

$$\lambda^c_{\boldsymbol{x}} := \frac{2}{1 - c\|\boldsymbol{x}\|^2}. \tag{1}$$

Since the hyperbolic space is a non-Euclidean space, the rudimentary operations, such as vector addition, cannot be applied (as they are not faithful to the geometry). The Möbius gyrovector space provides many standard operations for hyperbolic spaces. Essential to our developments in this work is the Möbius addition of two points $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{D}^n_c$, which is calculated as:

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

The geodesic distance between two points $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{D}^n_c$ can be obtained as:

$$d_c(\boldsymbol{x}, \boldsymbol{y}) = \frac{2}{\sqrt{c}}\operatorname{arctanh}(\sqrt{c}\| - \boldsymbol{x} \oplus_c \boldsymbol{y}\|). \tag{3}$$

Another essential operation used in our model is the hyperbolic averaging. The counterpart of Euclidean averaging in hyperbolic space is the *Einstein mid-point* which has the most simple form in $Klein$ coordinates (another model of the hyperbolic space which is isometric to the Poincaré ball). Thus, we transform the points from Poincaré (*i.e.*, $\boldsymbol{x}_{\mathbb{D}}$) ball model to Klein model (*i.e.*, $\boldsymbol{x}_{\mathbb{K}}$) using the transformation:

$$\boldsymbol{x}_{\mathbb{K}} = \frac{2\boldsymbol{x}_{\mathbb{D}}}{1 + c\|\boldsymbol{x}_{\mathbb{D}}\|^2}. \tag{4}$$

Then, the hyperbolic averaging in Klein model is obtained as:

$$\text{HypAve}(\boldsymbol{x}_1, \dots, \boldsymbol{x}_N) = \sum_{i=1}^{N} \gamma_i \boldsymbol{x}_i / \sum_{i=1}^{N} \gamma_i, \tag{5}$$

where $\gamma_i = \frac{1}{\sqrt{1 - c\|\boldsymbol{x}_i\|^2}}$ are the Lorentz factors. Finally, we transform the coordinates back to Poincaré model using:

$$\boldsymbol{x}_{\mathbb{D}} = \frac{\boldsymbol{x}_{\mathbb{K}}}{1 + \sqrt{1 - c\|\boldsymbol{x}_{\mathbb{K}}\|^2}}. \tag{6}$$

In our work, we make use of the tangent bundle of the $\mathbb{D}^n_c$. The logarithm map defines a function from $\mathbb{D}^n_c \to T_{\boldsymbol{x}}\mathbb{D}^n_c$, which projects a point in the Poincaré ball onto the tangent space at $\boldsymbol{x}$, as:

$$\boldsymbol{\pi}^c_{\boldsymbol{x}}(\boldsymbol{y}) = \frac{2}{\sqrt{c}\lambda^c_{\boldsymbol{x}}}\operatorname{arctanh}(\sqrt{c}\| - \boldsymbol{x} \oplus_c \boldsymbol{y}\|)\frac{-\boldsymbol{x} \oplus_c \boldsymbol{y}}{\| - \boldsymbol{x} \oplus_c \boldsymbol{y}\|}. \tag{7}$$

Point to Set Distance

Let $\mathcal{S} = \{\boldsymbol{s}_1, \dots, \boldsymbol{s}_k\}$ be a set. The distance from a point $\boldsymbol{p}$ to the set $\mathcal{S}$ can be defined in various forms. The min and max distance from a point $\boldsymbol{p}$ to the set $\mathcal{S}$ are two simple metrics, which can be defined as:

$$d^{\text{l}}_{\text{p2s}}(\boldsymbol{p}; \mathcal{S}) = \inf\{d(\boldsymbol{p}, \boldsymbol{s}_i) | \boldsymbol{s}_i \in \mathcal{S}\}, \tag{8}$$

$$d^{\text{h}}_{\text{p2s}}(\boldsymbol{p}; \mathcal{S}) = \sup\{d(\boldsymbol{p}, \boldsymbol{s}_i) | \boldsymbol{s}_i \in \mathcal{S}\}, \tag{9}$$

where inf and sup are the infimum and supremum functions, respectively. Given their geometrical interpretation, $d^{\text{l}}_{\text{p2s}}$ and $d^{\text{h}}_{\text{p2s}}$ define the lower and upper pairwise bounds, and fail to encode structured information about the set. Therefore, we opt for a weighted-sum formalism to measure the distance between a point and a set in § .

[1]In the supplementary material, we provide further details regarding the Poincaré ball model and its properties.

Method

This section will give an overview of the proposed method, followed by a detailed description of each component in our model.

Problem Formulation

We follow the standard protocol to formulate few-shot learning (FSL) with episodic training. An episode represents an N-way K-shot classification problem (*i.e.*, the training set, named support set, includes N classes where each class has K examples). As the name implies, K (*i.e.*, the number of examples per class) is small (*e.g.*, $K = 1$ or 5). The goal of learning is to realize a function $\mathcal{F} : \mathcal{X} \to \mathbb{R}^n$ to embed the support set to a latent and possibly lower-dimensional space, such that query samples can be recognized easily using a nearest neighbour classifier. To be specific, an episode or task $\mathcal{E}_i$ consists of a query set $\mathcal{X}^q = \{(\boldsymbol{X}_i^q, y_i^q)|i = 1, \ldots, N\}$, where $\boldsymbol{X}_i^q$ denotes a query example[2] sampled from class y_i^q, and a support set $\mathcal{X}^s = \{(\boldsymbol{X}_{ij}^s, y_i^s)|i = 1, \ldots, N, j = 1, \ldots, K\}$, where $\boldsymbol{X}_{ij}^s$ denotes the j-th sample in the class y_i^s. The embedding methods for FSL, our solution being one, often formulate training as:

$$\mathcal{F}^* := \arg\min_{\mathcal{F}} \sum_{\boldsymbol{X}_u^q \in \mathcal{X}^q} \delta\big(\mathcal{F}(\boldsymbol{X}_u^q), \mathcal{F}(\mathcal{X}_v^s)\big) \;\text{ s.t. } y_u^q = y_v^s, \tag{10}$$

where δ measures a form of distance between the query and the support samples.

Model Overview

We begin by providing a sketch of our method (see the conceptual diagram in Fig. 2(a) and Fig. 2(b)). The feature extractor network, denoted by $\mathcal{F}$, maps the input to a hyperbolic space in our work. We model every class in the support set by its signature. The signature is both class and episodic-aware, meaning that the signature will vary if the samples of a class or samples in the episode vary. This will enable us to calculate an adaptive distance from the query point to every support-class while being vigilant to the arrangement and constellation of the support samples. We stress that our design is different from many prior works where class-specific prototypes are learned for FSL. For example, in (Khrulkov et al. 2020; Snell, Swersky, and Zemel 2017; Sung et al. 2018), the prototypes are class-specific but not necessarily episodic-aware.

To obtain the signatures for each class in the support set, we project the support samples onto the tangent space of the query point and feed the resulting vectors to a signature generator f_ω. The signature generator realizes a permutation-invariant function and refines and summarizes its inputs to one signature per class. We then leverage a relational network f_ϕ to contrast samples of a class against their associated signature and produce a relational score. To obtain the adaptive P2S distance, we first compute a set to set (S2S) distance between the query feature map and each support feature map using the distance module f_ζ. Moreover, a weighted-sum is calculated using the relational score acting as the weight on the corresponding S2S distance, which serves as the P2S distance.

Given P2S distances, our network is optimized by minimizing the adaptive P2S distance between the query and its corresponding set while ensuring that the P2S distance to other classes (*i.e.*, wrong classes) is maximized.

Adaptive Poincaré Point to Set Distance

In FSL, we are given a small support set of K images, $\mathcal{X}_i^s = \{\boldsymbol{X}_{i1}^s, \ldots, \boldsymbol{X}_{iK}^s\}$ per class y_i^s to learn a classification model. We use a deep neural network to first encode the input to a multi-channel feature map, as $\mathcal{S}_i = \mathcal{F}(\mathcal{X}_i^s)$, with $\mathcal{S}_i = \{\boldsymbol{S}_{i1}, \ldots, \boldsymbol{S}_{iK} | \boldsymbol{S}_{ij} \in \mathbb{R}^{H \times W \times C}\}$, where H, W, and C indicate the height, width, and channel size of the instance feature map. Each feature map consists of a set of patch descriptors (local features), which can be further represented as $\boldsymbol{S}_{ij} = \{\boldsymbol{s}_{ij}^1, \ldots, \boldsymbol{s}_{ij}^{HW} | \boldsymbol{s}_{ij}^r \in \mathbb{R}^C\}$.

In our work, we train the network to embed the representation in the Poincaré ball; thus, we need to impose a constraint on patch descriptors at each spatial location $\boldsymbol{s}_{ij}^r$ as follows:

$$\boldsymbol{s}_{ij}^r = \begin{cases} \boldsymbol{s}_{ij}^r & \text{if } \|\boldsymbol{s}_{ij}^r\| \le \mu \\ \mu \boldsymbol{s}_{ij}^r / \|\boldsymbol{s}_{ij}^r\| & \text{if } \|\boldsymbol{s}_{ij}^r\| > \mu, \end{cases} \tag{11}$$

where μ is the norm upper bound of the vectors in the Poincaré ball. In our model, we choose $\mu = (1 - \epsilon)/c$, where c is the curvature of the Poincaré ball and ϵ is a small value that makes the system numerically stable. The same operation applies to the query sample, thereby obtaining an instance feature map for the query sample $\boldsymbol{Q} = \{\boldsymbol{q}^1, \ldots, \boldsymbol{q}^{HW}\}, \boldsymbol{q}^r \in \mathbb{D}_c^C$. Then the P2S distance between the query sample $\boldsymbol{Q}$ and the support set per class $\mathcal{S}_i$ can be calculated using Eq. (8) or Eq. (9). However, those two metrics only determine the lower or upper bound of P2S distance, thereby ignoring the structure and distribution of the set to a great degree. To make better use of the distribution of samples in a set, we propose the adaptive P2S distance metric as:

$$d_{\mathrm{p2s}}^{\mathrm{adp}}(\boldsymbol{Q}; \mathcal{S}_i) := \frac{\sum_{j=1}^{K} w_{ij} d(\boldsymbol{Q}, \boldsymbol{S}_{ij})}{\sum_{j=1}^{K} w_{ij}}, \tag{12}$$

where w_{ij} is the adaptive factor for $d(\boldsymbol{Q}, \boldsymbol{S}_{ij})$. We refer to the distance in Eq. (12) as Adaptive Poincaré Point to Set (APP2S) distance, hereafter.

In Eq. (12), we need to calculate the distance between two feature maps (*i.e.*, $d(\boldsymbol{Q}, \boldsymbol{S}_{ij})$). In doing so, we formulate a feature map as a set (*i.e.*, $\{\boldsymbol{q}^1, \ldots, \boldsymbol{q}^{HW}\}$ and $\{\boldsymbol{s}_{ij}^1, \ldots, \boldsymbol{s}_{ij}^{HW}\}$), such that a set to set (S2S) distance can be obtained. One-sided Hausdorff and two-sided Hausdorff distances (Huttenlocher, Klanderman, and Rucklidge 1993) are two widely used metrics to measure the distance between sets. However, these two metrics are sensitive to

[2]Without losing generality, we use one sample per class as a query for presenting our method. In practice, each episode contains multiple samples for the query image per class.

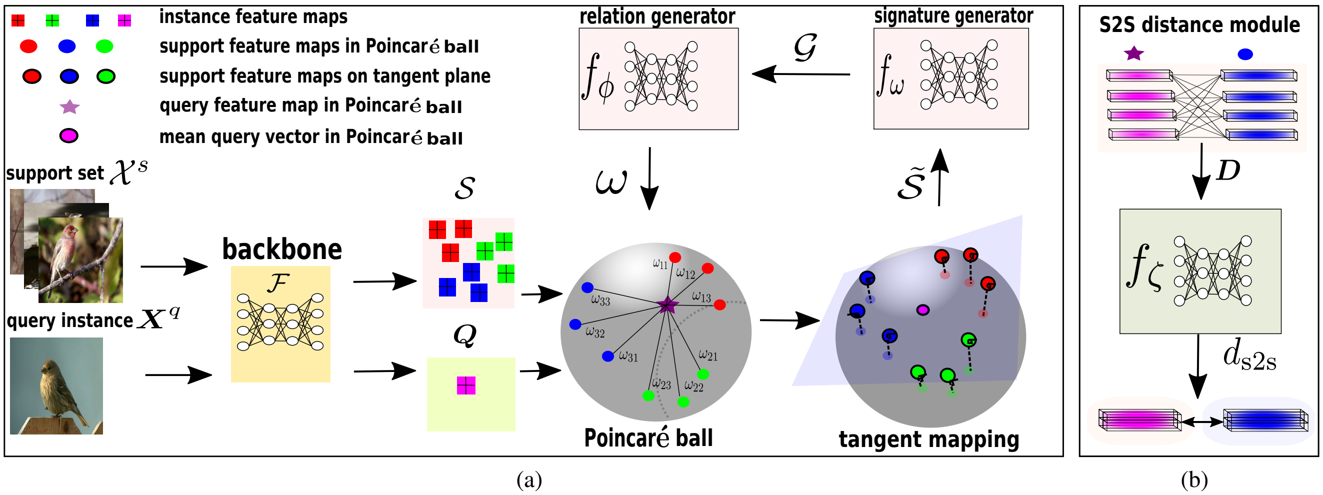


Figure 2: (a): The overall pipeline of our method. Given an episode, we use a backbone network to extract and map the inputs to a hyperbolic space. We then project the support samples onto the tangent plane of the query point and employ a refinement function f_ω to obtain the class and episode-aware signature of every class in the support set. This is followed by a mapping f_ϕ that weighs the importance of each sample of the support set w.r.t. their corresponding class. This enables us to calculate an adaptive point to set distance towards inference. (b): The S2S distance module. We calculate a pair-wise distance between each feature vector of the two feature maps as the input of the distance network f_ζ. Then the f_ζ outputs a S2S distance, which is further used to compute the adaptive P2S.

outliers (Huttenlocher, Klanderman, and Rucklidge 1993; Ibanez et al. 2008). To alleviate this issue, we propose to learn the S2S distance by a network f_ζ. We first calculate the pair-wise distance between two sets as $\boldsymbol{D}(\boldsymbol{Q}, \boldsymbol{S}_{ij}) \in \mathbb{R}^{HW \times HW}$, where each element in $\boldsymbol{D}$ is obtained by $d_{h,w} = d_c(\boldsymbol{q}^h, \boldsymbol{s}_{ij}^w)$ using Eq. (3), where $h = 1, \ldots, HW$ and $w = 1, \ldots, HW$. Then we use a neural network to further learn the distance between two feature maps (see Fig. 2(b)), which is given by:

$$d_{\mathrm{s2s}}(\boldsymbol{Q}, \boldsymbol{S}_{ij}) := f_\zeta(\boldsymbol{D}). \tag{13}$$

Comparing to the Hausdorff distance (Conci and Kubrusly 2018) (see supplementary material), our set to set distance is more flexible and learned through the optimization process.

To further obtain the weights of APP2S (*i.e.*, w_{ij}), we make use of the tangent space of the query sample. We first compute a mean query vector $\bar{\boldsymbol{q}}$ over the spatial dimensions of the feature map $\boldsymbol{Q}$ using Eq. (4)- Eq. (6). Then, we project the samples in the support set to the tangent space of the mean query vector (see Fig. 2(a)), using the logarithm map as:

$$\tilde{\mathcal{S}} = \boldsymbol{\pi}_{\bar{\boldsymbol{q}}}^c(\mathcal{S}), \tag{14}$$

where $\tilde{\mathcal{S}}$ indicates the projected support set on the tangent space at $\bar{\boldsymbol{q}}$. For the i-th class, we can obtain a set of feature maps: $\tilde{\mathcal{S}}_i = \{\tilde{\boldsymbol{S}}_{i1}, \ldots, \tilde{\boldsymbol{S}}_{iK}\}$[3]. To obtain a meaningful weight w_{ij}, we first propose a signature generator, which jointly refines sample representations in the support set and summarizes the set representation per class as the class signature. As shown in Fig. 2(a), the signature generator receives the projected support set $\tilde{\mathcal{S}} = \{\tilde{\mathcal{S}}_1, \ldots, \tilde{\mathcal{S}}_N\}$ as input and refines them for the follow-up task (*i.e.*, obtaining the weights for the APP2S). We denote the output of the refinement module by $\hat{\mathcal{S}} = f_\omega(\tilde{\mathcal{S}})$ ($\hat{\mathcal{S}} = \{\hat{\mathcal{S}}_1, \ldots, \hat{\mathcal{S}}_N\}$, $\hat{\mathcal{S}}_i = \{\hat{\boldsymbol{S}}_{i1}, \ldots, \hat{\boldsymbol{S}}_{iK}\}$). One can understand the refinement function as learning the context of the support set by seeing all the samples, thereby highlighting the discriminative samples and restraining the non-informative samples such as outliers for all the samples. Then the signature for each class is obtained by summarizing as: $\bar{\boldsymbol{S}}_i = \sum_{j=1}^K \hat{\boldsymbol{S}}_{ij}/K$.

[3]The projected feature map is also composed by the vectors at each spatial location $\tilde{\boldsymbol{S}}_{ij} = \{\tilde{\boldsymbol{s}}_{ij}^1, \ldots, \tilde{\boldsymbol{s}}_{ij}^{HW}\}$

Remark 1 *Our proposed set-signature generator f_ω is similar to the set-to-set function in FEAT (Ye et al. 2020), in the sense that both functions perform self-attention over the input features. However, the fundamental difference is that our module exploits the relation between the spatial feature descriptors of all samples in a support set* (e.g., $\tilde{\boldsymbol{s}}_{ij}^r$)*, instead of prototypes as proposed in FEAT (Ye et al. 2020), which possibly gives the model more flexibility to encode meaningful features.*

Given sample features in a class $\tilde{\mathcal{S}}_i = \{\tilde{\boldsymbol{S}}_{i1}, \ldots, \tilde{\boldsymbol{S}}_{iK}\}$ and the corresponding class signature $\bar{\boldsymbol{S}}_i$, we use a relation generator (*i.e.*, f_ϕ in Fig. 2(a)) to compare the relationship between an individual feature map and the class signature. In doing so, we first concatenate the individual feature maps and their class signature along the channel dimension to obtain a hybrid representation, as:

$$\boldsymbol{G}_{ij} = \mathrm{CONCAT}(\tilde{\boldsymbol{S}}_{ij}, \bar{\boldsymbol{S}}_i). \tag{15}$$

Given the hybrid representation $\boldsymbol{G}_{ij}$, the relation generator produces a relation score as: $w_{ij} = f_\phi(\boldsymbol{G}_{ij})$. This score will serve as the adaptive factor for the APP2S distance metric in Eq. (12). Note that the hybrid representation for the whole support and a support-class set are denoted by $\mathcal{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_N\}$ and $\mathcal{G}_i = \{\boldsymbol{G}_{i1}, \ldots, \boldsymbol{G}_{iK}\}$, respectively. Algorithm 1 summarizes the process of training our APP2S metric for FSL.

Remark 2 *The point to set distance defined by Eq.* (12) *is different from that in MatchingNet (Vinyals et al. 2016). MatchingNet formulates all the samples in the support set as a set. In contrast, we treat the samples in a class as the set, which makes our adaptive point to set distance fully contextual aware of the whole support set (by the set-signature) and encodes the distribution of each class.*

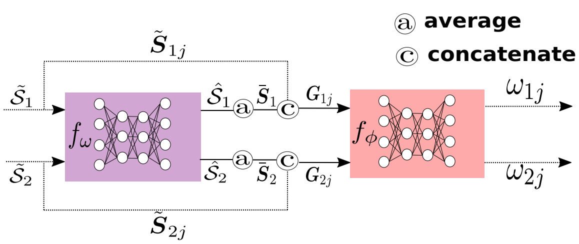


Figure 3: The information flow of the signature generator and the relation generator. The signature generator f_ω receives the projected support set $\{\tilde{\mathcal{S}}_1, \tilde{\mathcal{S}}_2\}$ as a bag and outputs the refined representation per sample such that each element (*i.e.*, $\tilde{\boldsymbol{S}}_{ij}$) sees all the other elements of the support set. Then the refined representations per class/set $\hat{\mathcal{S}}_1$ and $\hat{\mathcal{S}}_2$ are averaged to obtain the signature per class/set $\bar{\boldsymbol{S}}_1$ and $\bar{\boldsymbol{S}}_2$. Finally, we concatenate one projected sample $\tilde{\boldsymbol{S}}_{1j}$ and $\tilde{\boldsymbol{S}}_{2j}$ with the corresponding class signature $\bar{\boldsymbol{S}}_1$ and $\bar{\boldsymbol{S}}_2$, repectively, and feed it into relation generator f_ϕ to produce the adaptive factors ω_{1j} and ω_{2j}.

Algorithm 1: Train network using adaptive Poincaré point to set distance

Input: An episodes $\mathcal{E}$, with their associated support set $\mathcal{X}^s = \{(\boldsymbol{X}^s_{ij}, y^s_i) | i = 1, \ldots, N, j = 1, \ldots, K\}$ and a query sample $\boldsymbol{X}^q$

Output: The optimal parameters for $\mathcal{F}$, f_ω, f_ς, and f_ϕ

Map $\mathcal{X}^s$ and $\boldsymbol{X}^q$ into Poincaré ball

Obtain the tangent support set $\tilde{\mathcal{S}}$ using Eq. (7)

$\hat{\mathcal{S}} = f_\omega(\tilde{\mathcal{S}})$ ▷ the refined support set

for i in $\{1, ..., N\}$ **do**

  $\bar{\boldsymbol{S}}_i = \sum_{j=1}^{K} \hat{\boldsymbol{S}}_{ij}/K$ ▷ the set signature

  $\boldsymbol{G}_{ij} = \mathrm{CONCAT}(\tilde{\boldsymbol{S}}_{ij}, \bar{\boldsymbol{S}}_i)$ ▷ the hybrid representation

  $\omega_{ij} = f_\phi(\boldsymbol{G}_{ij})$ ▷ the weight

  Compute point to set distance and set to set distance using Eq. (12) and Eq. (13)

end for

Optimize the model using Eq. (10)

Related Work

In this section, we discuss the literature on few-shot learning and highlight those that motivate this work. Generally, there are two main branches on the few-shot learning literature, optimization-based and metric-based methods. The optimization-based methods (Antoniou, Edwards, and Storkey 2019; Chen et al. 2019; Finn, Abbeel, and Levine 2017; Flennerhag et al. 2019; Franceschi et al. 2018; Nichol, Achiam, and Schulman 2018), such as MAML and Reptile (Finn, Abbeel, and Levine 2017; Nichol, Achiam, and Schulman 2018), aim to learn a set of initial model parameters that can adapt to new tasks quickly using backpropagation in the episodic regime, without severe overfitting. However, this group of methods usually adopt a bi-level optimization setting to optimize the initial parameters, which is computationally expensive during inference.

On the other hand, our proposed method is closer to metric-based methods (Simon et al. 2020; Snell, Swersky, and Zemel 2017; Sung et al. 2018; Vinyals et al. 2016; Ye et al. 2020; Zhang et al. 2020; Tang et al. 2020; Ma et al. 2021), which target to realize an embedding: $\mathbb{R}^M \rightarrow \mathbb{R}^D$ to represent images in semantic space equipped with an appropriate distance metric such that different categories are distinctive. Matching Network (Vinyals et al. 2016) determines the query labels by learning a sample-wise distance along with a self-attention mechanism that produces a fully contextualized embedding over samples. Prototypical Network (Snell, Swersky, and Zemel 2017) takes a step further from a sample-wise to a class-wise metric, where all the samples of a class are averaged into a prototype to represent the class in the embedding space. Relation Network (Sung et al. 2018) and CTM (Li et al. 2019a) replace the handcrafted metric with a network to encode the non-linear relation between the class representations and the query embedding. Ye *et al.* (Ye et al. 2020) propose adopting a transformer to learn the task-specific features for few-shot learning. Zhang *et al.* (Zhang et al. 2020) adopt the Earth Mover's Distance as a metric to compute a structural distance between representation to obtain the labels for the query images. Simon *et al.* (Simon et al. 2020) propose to generate a dynamic classifier via using subspace. Along this line of research, most of the previous methods utilize the global feature vectors as representations. However, several recent works have demonstrated that utilizing the local feature maps can further boost performance. Therefore, we follow these works (Doersch, Gupta, and Zisserman 2020; Zhang et al. 2020; Wertheimer, Tang, and Hariharan 2021; Lifchitz et al. 2019; Li et al. 2019b) to develop our model.

However, the majority of the aforementioned metric-based works employ various metrics within Euclidean space. Ganea *et al.* (Ganea, Bécigneul, and Hofmann 2018) have proved that embedding via hyperbolic geometry enjoys low distortion for hierarchical and structured data (*e.g.*, trees) and developed the hyperbolic version of the feed-forward neural networks and recurrent neural networks (RNN). Moreover, a recent work (Khrulkov et al. 2020) has shown that the vision tasks can largely benefit from hyperbolic embeddings, which inspires us to further develop algorithms with hyperbolic geometry.

Experiments

Datasets

In this section, we will empirically evaluate our approach across five standard benchmarks, *i.e.*, *mini*-ImageNet (Ravi and Larochelle 2016), *tiered*-ImageNet (Ren et al. 2018), Caltech-UCSD Birds-200-2011 (CUB) (Wah et al. 2011), CIFAR-FS (Bertinetto et al. 2018) and Fewshot-CIFAR100 (FC100) (Oreshkin, López, and Lacoste 2018). Full details of the datasets and implementation are described in the supplementary material. In the following, we will briefly describe our results on each dataset.

Main Result

We evaluate our methods for 100 epochs, and in each epoch, we sample 100 tasks (episodes) randomly from the test set, for both 5-way 1-shot and 5-way 5-shot settings. Following the standard protocol (Simon et al. 2020), we report the mean accuracy with 95% confidence interval.

***mini*-ImageNet**. As shown in Table 1, we evaluate our model using ResNet-12 and ResNet-18 as the backbones on *mini*-ImageNet. Between them, ResNet-12 produces the best results. In addition, our model also outperforms recent state-of-the-art models in most of the cases. Interestingly, our model further outperforms hyperbolic ProtoNet by 7.77% and 7.11% for 5-way 1-shot and 5-way 5-shot with ResNet-18, respectively. With ResNet-12, we outperform the hyperbolic ProtoNet by 5.60% and 7.29% for 5-way 1-shot and 5-way 5-shot, respectively.

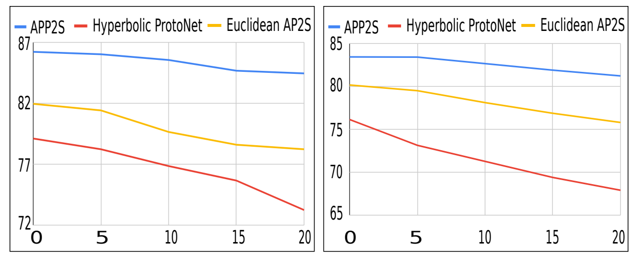
***tiered*-ImageNet**. We further evaluate our model on *tiered*-ImageNet with ResNet backbones. The results in Table 1 indicate that with ResNet-12, our model outperforms the hyperbolic ProtoNet by 4.62% and 7.12% for 5-way 1-shot and 5-way 5-shot, respectively, and achieves state-of-the-art results for inductive few-shot learning.

CIFAR-FS and FC100. As the results in Table 2 suggested, our model also achieves comparable performance with the relevant state-of-the-state methods on this dataset, with ResNet-12 backbone, which vividly shows the superiority of our method.

CUB. We use ResNet-18 as our backbone to evaluate our method on the CUB dataset. Table 3 shows that our model improves the performance over baseline by 3.94% and 4.88% for 5-way 1-shot and 5-way 5-shot settings, respectively. Besides, our model achieves 77.64% and 90.43% for 5-way 1-shot and 5-way 5-shot settings on this dataset, which outperforms state-of-the-art models (*i.e.*, DeepEMD (Zhang et al. 2020) and P-transfer (Shen et al. 2021)) and achieve competitive performance on this dataset.

Robustness to Outliers

To further validate the robustness of our method, we conduct experiments in the presence of outliers in the form of mislabelled images. In the first study, we add a various number of outliers (*e.g.*, 1, 2, 3, 4), whose classes are disjoint to the support-class, to each class of the support set. We performed this study with ResNet-12 backbone on the 5-way 5-shot setting on *tiered*-ImageNet. Fig. 4(a) shows that the performances of hyperbolic ProtoNet degrade remarkably. On the contrary, both our APP2S and Euclidean AP2S are robust to outliers, which shows the superiority of our adaptive metric. Comparing to Euclidean AP2S, APP2S is even more robust (see the slope of Fig. 4(a)) and performs consistently even in the presence of 20 outliers. This suggests that integrating our proposed adaptive metric and hyperbolic geometry can further bring robustness to our framework. In the second study (shown in Fig. 4(b)), we conduct the same experiments on *mini*-ImageNet. The results show a similar trend as the previous one, which further proves the effectiveness of our proposed method.



(a) (b)

Figure 4: Robustness analysis. Horizontal axis: The number of outliers. Vertical axis: Accuracy. (a): The performance vs. the number of outliers on *tiered*-ImageNet. (b): The performance vs. the number of outliers on *mini*-ImageNet.

Ablation Study

We further conduct the ablation study to verify the effectiveness of each component in our method on the *tiered*-ImageNet dataset using the ResNet-12 backbone.

Experiments Set-Up. For setting (ii) in Table 4, we disable the relation module f_ϕ and signature generator f_ω. The P2S distance can be obtained by Eq. (12) and Eq. (13) with equal weights (*i.e.*, 1). Moreover, we enable the relation generator f_ϕ but not the signature generator in setting (iii). We use the class prototype instead of the signature for this experiment. We enable both f_ϕ and f_ω and use the Euclidean distances for setting (iv). In the end, we enable the Poincaré ball but disable the f_ζ for setting (v). In terms of implementation of (v), the backbone is designed to output a feature vector instead of a feature map, such that the P2S distance can be directly computed by Eq. (3) and Eq. (12).

Effectiveness of Point to Set Distance. In this experiment, we first evaluate the effectiveness of the P2S distance by comparing to its point to point (P2P) distance counterpart (*i.e.*, hyperbolic ProtoNet). From Table 4, we could observe that the P2S distance can learn a more discriminative embedding space than P2P distance (*i.e.*, (i) vs. (ii)), and the adaptive P2S can further bring performance gain to our application (*i.e.*, (ii) vs. (iii)). This observation shows the potential of our P2S distance setting in the FSL task.

Effectiveness of Signature Generator. We further evaluate another essential component in our work, *i.e.*, the signature generator, which refines the entire support set and produces a signature per class. As shown in Table 4 (*i.e.*, (iii) and (vi)),

Model	Backbone	*mini*-ImageNet		*tiered*-ImageNet	
		1-shot	5-shot	1-shot	5-shot
ProtoNet (Snell, Swersky, and Zemel 2017)	ResNet-12	60.37 ± 0.83	78.02 ± 0.57	61.74 ± 0.77	80.00 ± 0.55
MatchingNet (Vinyals et al. 2016)	ResNet-12	63.08 ± 0.80	75.99 ± 0.60	68.50 ± 0.92	80.60 ± 0.71
MetaOptNet (Lee et al. 2019)	ResNet-12	62.64 ± 0.61	78.63 ± 0.46	65.99 ± 0.72	81.56 ± 0.53
DeepEMD (Zhang et al. 2020)	ResNet-12	65.91 ± 0.82	82.41 ± 0.56	71.16 ± 0.87	86.03 ± 0.58
P-transfer (Shen et al. 2021)	ResNet-12	64.21 ± 0.77	80.38 ± 0.59	-	-
GLoFA (Lu, Ye, and Zhan 2021)	ResNet-12	66.12 ± 0.42	81.37 ± 0.33	69.75 ± 0.33	83.58 ± 0.42
DMF (Xu et al. 2021)	ResNet-12	$\mathbf{67.76 \pm 0.46}$	82.71 ± 0.31	71.89 ± 0.52	85.96 ± 0.35
Hyperbolic ProtoNet (Khrulkov et al. 2020)	ResNet-12	$*60.65 \pm 0.18$	$*76.13 \pm 0.21$	$*67.38 \pm 0.14$	$*79.11 \pm 0.22$
Ours (APP2S)	ResNet-12	66.25 ± 0.20	$\mathbf{83.42 \pm 0.15}$	$\mathbf{72.00} \pm 0.22$	$\mathbf{86.23 \pm 0.15}$
LwoF (Gidaris and Komodakis 2018)	WRN-28-10	60.06 ± 0.14	76.39 ± 0.11	-	-
wDAE-GNN (Gidaris and Komodakis 2019)	WRN-28-10	61.07 ± 0.15	76.75 ± 0.11	68.18 ± 0.16	83.09 ± 0.12
LEO (Rusu et al. 2018)	WRN-28-10	61.76 ± 0.08	77.59 ± 0.12	66.33 ± 0.05	82.06 ± 0.08
Su *et al*. (Su, Maji, and Hariharan 2020)	ResNet-18	-	76.60 ± 0.70	-	78.90 ± 0.70
AFHN (Li et al. 2020)	ResNet-18	62.38 ± 0.72	78.16 ± 0.56	-	-
Neg-Cosine (Liu et al. 2020)	ResNet-18	62.33 ± 0.82	80.94 ± 0.59	-	-
Hyperbolic ProtoNet (Khrulkov et al. 2020)	ResNet-18	$*57.05 \pm 0.16$	$*74.20 \pm 0.14$	$*66.20 \pm 0.12$	$*76.50 \pm 0.13$
Ours (APP2S)	ResNet-18	64.82 ± 0.12	81.31 ± 0.22	70.83 ± 0.15	84.15 ± 0.29

Table 1: Few-shot classification accuracy and 95 % confidence interval on *mini*-ImageNet and *tiered*-ImageNet with ResNet backbones. "*" notes the result obtained by the self-implemented network.

Model	Backbone	CIFAR-FS		FC100	
		1-shot	5-shot	1-shot	5-shot
TEAM (Qiao et al. 2019)	ResNet-12	70.40	81.30	-	-
ProtoNet (Snell, Swersky, and Zemel 2017)	ResNet-12	72.20 ± 0.70	83.50 ± 0.50	37.50 ± 0.60	52.50 ± 0.60
TADAM (Oreshkin, López, and Lacoste 2018)	ResNet-12	-	-	40.10 ± 0.40	56.10 ± 0.40
DeepEMD (Zhang et al. 2020)	ResNet-12	-	-	46.47 ± 0.78	63.22 ± 71
Hyperbolic ProtoNet (Khrulkov et al. 2020)	ResNet-12	$*70.27 \pm 0.22$	$*80.98 \pm 0.16$	$*36.04 \pm 0.18$	$*51.60 \pm 0.18$
Ours (APP2S)	ResNet-12	$\mathbf{73.12 \pm 0.22}$	$\mathbf{85.69 \pm 0.16}$	$\mathbf{47.64 \pm 0.21}$	$\mathbf{63.56 \pm 0.22}$

Table 2: Few-shot classification accuracy and 95 % confidence interval on CIFAR-FS and FC100 with ResNet-12 backbones. "*" notes the result obtained by the self-implemented network.

Model	1-shot	5-shot
MatchingNet	72.36 ± 0.90	83.64 ± 0.60
ProtoNet	71.88 ± 0.91	87.42 ± 0.48
DeepEMD ♣	76.65 ± 0.83	88.69 ± 0.50
P-transfer ♣	73.88 ± 0.87	87.81 ± 0.48
Hyperbolic ProtoNet	$*73.70 \pm 0.22$	$*85.55 \pm 0.13$
Ours (APP2S)	$\mathbf{77.64 \pm 0.19}$	$\mathbf{90.43 \pm 0.18}$

Table 3: Few-shot classification accuracy and 95 % confidence interval on CUB. "*" notes the result obtained by the self-implemented network. "♣" denotes the method using ResNet-12 as the backbone, otherwise ResNet-18.

we could observe that our method benefits from the signature generator, which shows that the signature of each class could help to generate an informative weight for individual feature map within the same class.

Effectiveness of Hyperbolic Geometry. We also implement our model in the Euclidean space to verify the effectiveness of our method. The row (iv) and (vi) in Table 4 vividly show that the representation in the Poincaré ball has a richer embedding than that in Euclidean spaces.

Effectiveness of Set to Set Distance. The comparison between (v) and (vi) shows that our set to set distance generator associated with the feature map outputs richer information than using a feature vector to directly compute the APP2S.

ID	Model	5-shot
(i)	Hyperbolic ProtoNet	79.11 ± 0.22
(ii)	Hyperbolic P2S w/o f_ϕ	83.14 ± 0.17
(iii)	Hyperbolic P2S w/ f_ϕ	84.88 ± 0.17
(iv)	Euclidean AP2S	81.96 ± 0.18
(v)	APP2S w/o f_ζ	84.12 ± 0.13
(vi)	**APP2S**	$\mathbf{86.23 \pm 0.15}$

Table 4: The ablation study of our model, we start from the hyperbolic ProtoNet (Khrulkov et al. 2020) towards APP2S.

Conclusion

In this paper, we propose a novel adaptive Poincaré point to set (APP2S) distance metric for the few-shot learning, which can adapt depending on the samples at hands. Empirically, we showed that this approach is expressive with both hyperbolic geometry and Euclidean counterpart. Our model improves the performances over baseline models and achieves competing results on five standard FSL benchmarks.

References

Antoniou, A.; Edwards, H.; and Storkey, A. 2019. How to train your MAML. In *International Conference on Learning Representations*.

Bertinetto, L.; Henriques, J. F.; Torr, P.; and Vedaldi, A. 2018. Meta-learning with differentiable closed-form solvers. In *International Conference on Learning Representations*.

Chen, W.-Y.; Liu, Y.-C.; Kira, Z.; Wang, Y.-C. F.; and Huang, J.-B. 2019. A Closer Look at Few-shot Classification. In *International Conference on Learning Representations*.

Conci, A.; and Kubrusly, C. S. 2018. Distance Between Sets - A survey. *Advances in Mathematical Sciences and Applications*.

Doersch, C.; Gupta, A.; and Zisserman, A. 2020. Crosstransformers: spatially-aware few-shot transfer. *arXiv preprint arXiv:2007.11498*.

Fang, P.; Harandi, M.; and Petersson, L. 2021. Kernel Methods in Hyperbolic Spaces. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 1126–1135.

Flennerhag, S.; Rusu, A. A.; Pascanu, R.; Visin, F.; Yin, H.; and Hadsell, R. 2019. Meta-Learning with Warped Gradient Descent. In *International Conference on Learning Representations*.

Franceschi, L.; Frasconi, P.; Salzo, S.; Grazzi, R.; and Pontil, M. 2018. Bilevel programming for hyperparameter optimization and meta-learning. In *International Conference on Machine Learning*, 1568–1577. PMLR.

Ganea, O.; Bécigneul, G.; and Hofmann, T. 2018. Hyperbolic neural networks. In *Advances in Neural Information Processing Systems*.

Gidaris, S.; and Komodakis, N. 2018. Dynamic few-shot visual learning without forgetting. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Gidaris, S.; and Komodakis, N. 2019. Generating classification weights with gnn denoising autoencoders for few-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Hong, J.; Fang, P.; Li, W.; Zhang, T.; Simon, C.; Harandi, M.; and Petersson, L. 2021. Reinforced Attention for Few-Shot Learning and Beyond. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Huttenlocher, D. P.; Klanderman, G. A.; and Rucklidge, W. J. 1993. Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Ibanez, L.; Audette, M.; Yeo, B.; Golland, P.; Tustison, N.; and Gee, J. 2008. The Use of Robust Local Hausdorff Distances in Accuracy Assessment for Image Alignment of Brain MRI. *Insight Journal*.

Khrulkov, V.; Mirvakhabova, L.; Ustinova, E.; Oseledets, I.; and Lempitsky, V. 2020. Hyperbolic image embeddings. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Lee, K.; Maji, S.; Ravichandran, A.; and Soatto, S. 2019. Meta-learning with differentiable convex optimization. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Li, H.; Eigen, D.; Dodge, S.; Zeiler, M.; and Wang, X. 2019a. Finding task-relevant features for few-shot learning by category traversal. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Li, K.; Zhang, Y.; Li, K.; and Fu, Y. 2020. Adversarial Feature Hallucination Networks for Few-Shot Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Li, W.; Wang, L.; Xu, J.; Huo, J.; Gao, Y.; and Luo, J. 2019b. Revisiting local descriptor based image-to-class measure for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7260–7268.

Lifchitz, Y.; Avrithis, Y.; Picard, S.; and Bursuc, A. 2019. Dense classification and implanting for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9258–9267.

Liu, B.; Cao, Y.; Lin, Y.; Li, Q.; Zhang, Z.; Long, M.; and Hu, H. 2020. Negative margin matters: Understanding margin in few-shot classification. In *European Conference on Computer Vision*, 438–455. Springer.

Lu, S.; Ye, H.-J.; and Zhan, D.-C. 2021. Tailoring Embedding Function to Heterogeneous Few-Shot Tasks by Global and Local Feature Adaptors. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 8776–8783.

Ma, R.; Fang, P.; Avraham, G.; Zuo, Y.; Drummond, T.; and Harandi, M. 2021. Learning Instance and Task-Aware Dynamic Kernels for Few Shot Learning. *arXiv preprint arXiv:2112.03494*.

Nichol, A.; Achiam, J.; and Schulman, J. 2018. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Oreshkin, B.; López, P. R.; and Lacoste, A. 2018. Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in Neural Information Processing Systems*.

Qiao, L.; Shi, Y.; Li, J.; Wang, Y.; Huang, T.; and Tian, Y. 2019. Transductive episodic-wise adaptive metric for few-shot learning. In *IEEE International Conference on Computer Vision*.

Ravi, S.; and Larochelle, H. 2016. Optimization as a model for few-shot learning. In *International Conference on Learning Representations*.

Ren, M.; Triantafillou, E.; Ravi, S.; Snell, J.; Swersky, K.; Tenenbaum, J. B.; Larochelle, H.; and Zemel, R. S. 2018. Meta-Learning for Semi-Supervised Few-Shot Classification. In *International Conference on Learning Representations*.

Rusu, A. A.; Rao, D.; Sygnowski, J.; Vinyals, O.; Pascanu, R.; Osindero, S.; and Hadsell, R. 2018. Meta-Learning with Latent Embedding Optimization. In *International Conference on Learning Representations*.

Shen, Z.; Liu, Z.; Qin, J.; Savvides, M.; and Cheng, K.-T. 2021. Partial Is Better Than All: Revisiting Fine-tuning Strategy for Few-shot Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 9594–9602.

Simon, C.; Koniusz, P.; Nock, R.; and Harandi, M. 2020. Adaptive Subspaces for Few-Shot Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Snell, J.; Swersky, K.; and Zemel, R. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*.

Su, J.-C.; Maji, S.; and Hariharan, B. 2020. When does self-supervision improve few-shot learning? In *European Conference on Computer Vision*, 645–666. Springer.

Sun, S.; Sun, Q.; Zhou, K.; and Lv, T. 2019. Hierarchical attention prototypical networks for few-shot text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, 476–485.

Sung, F.; Yang, Y.; Zhang, L.; Xiang, T.; Torr, P. H.; and Hospedales, T. M. 2018. Learning to compare: Relation network for few-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Tang, H.; Li, Z.; Peng, Z.; and Tang, J. 2020. BlockMix: meta regularization and self-calibrated inference for metric-based meta-learning. In *Proceedings of the 28th ACM International Conference on Multimedia*, 610–618.

Vinyals, O.; Blundell, C.; Lillicrap, T.; Wierstra, D.; et al. 2016. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*.

Wah, C.; Branson, S.; Welinder, P.; Perona, P.; and Belongie, S. 2011. The Caltech-UCSD Birds-200-2011 Dataset. Technical report.

Wang, Y.; Xu, C.; Liu, C.; Zhang, L.; and Fu, Y. 2020. Instance credibility inference for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12836–12845.

Wertheimer, D.; Tang, L.; and Hariharan, B. 2021. Few-Shot Classification With Feature Map Reconstruction Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8012–8021.

Xu, C.; Fu, Y.; Liu, C.; Wang, C.; Li, J.; Huang, F.; Zhang, L.; and Xue, X. 2021. Learning Dynamic Alignment via Meta-filter for Few-shot Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5182–5191.

Ye, H.-J.; Hu, H.; Zhan, D.-C.; and Sha, F. 2020. Few-shot learning via embedding adaptation with set-to-set functions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Zhang, C.; Cai, Y.; Lin, G.; and 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*.