HAPPI: Hyperbolic Hierarchical Prototypes for Image Recognition

Anonymous ICCV-BEW submission

Paper ID 11

Abstract

*Prototypical neural networks have gained prominence in computer vision due to their inherent interpretability, enabling decisions based on representative examples without the need for post-hoc explainability methods. However, many prototype-based models overlook the hierarchical relationships within image features, treating them independently and often resulting in suboptimal performance for tasks requiring complex structural understanding. To address this limitation, we propose HAPPI (**H**ierarchical **A**nd **P**rototy**P**ical **I**mage recognition), a framework that leverages hyperbolic geometry to organize prototypes hierarchically within a Lorentzian manifold. By arranging generic features near the origin of the hyperboloid and specific features farther away, HAPPI enables the learning of generic prototypes for consistent and defining patterns and specific prototypes for fine-grained and variable details, effectively capturing hierarchical relationships in image data. Our approach is model-agnostic and can be applied to various prototypical neural networks and backbones. We evaluate HAPPI across several prototypical model baselines and datasets, demonstrating its versatility and showing that hyperbolic prototypes consistently match or outperform their Euclidean counterparts in quantitative accuracy while providing additional interpretability. Qualitative visualizations reveal that generic prototypes capture consistent, semantically important distinctions between classes. In contrast, specific prototypes capture fine-grained variations within each class, providing essential nuances for detailed classification and enhancing the model's ability to differentiate across multiple levels of abstraction. Our code can be found at github.com/**********.*

1. Introduction

Prototypical neural networks have emerged as a prominent field in computer vision due to their ability to make interpretable and explainable decisions. They hold an advantage over traditional black-box models as they can provide the reasoning behind their results without reliance on post-hoc explainability methods, leading to more trustworthy decision making. Prototypical neural networks such as ProtoPNet [1], ProtoPNet variants [3, 27] and PIP-Net [22] have shown success in a variety of image classification tasks.

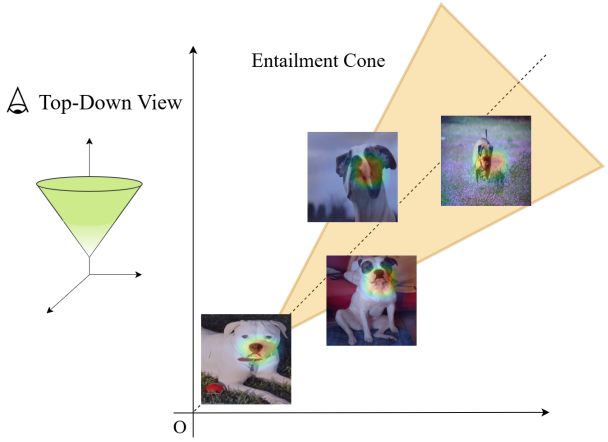

Figure 1. Organization of hierarchical prototypes in hyperbolic space, where generic prototypes that capture defining consistent features, entail specific prototypes that capture intra-class variations such as color, texture, and orientation.

A common shortcoming of many prototype-based models is their tendency to treat image features independently, without considering the hierarchical relationships between them. This limitation can result in suboptimal performance on tasks requiring a deeper understanding of image structure, such as object recognition, scene understanding, and fine-grained classification. In real-world images, features are often organized hierarchically, with more generic features capturing consistent and defining characteristics, while more specific features reflect variable details. Generic features are particularly effective for distinguishing broad categories due to their relevance and distinctiveness across classes. However, when differentiating between visually similar classes—such as distinguishing between dog breeds—specific features become more valuable,

as they capture fine-grained variations necessary for precise classification. Therefore, by neglecting this hierarchy, existing prototypical models may miss critical context needed for accurate predictions.

As illustrated in Fig. 1, organizing prototypes hierarchically allows models to capture both defining and variable class features in a structured manner. Generic prototypes focus on fundamental and consistent traits that reliably characterize a class—for instance, in animal images, they may capture the typical shape of a nose, a paw, or an ear that remains stable across variations. In contrast, specific prototypes highlight within-class diversity by attending to more variable aspects, such as different regions of the body (e.g., a tail instead of a nose) or feature combinations that vary in color, texture, or orientation. By structuring prototypes this way, the model can differentiate classes based on both their defining traits and the finer variations that help distinguish similar-looking instances.

To address this limitation, we introduce **HAPPI**, a **H**ierarchical **A**nd **P**rototy**P**ical **I**mage recognition methodology, which learns prototypes in a hyperbolic space using a Lorentzian manifold instead of Euclidean space. Our method structures the embedding space so that generic features—representing stable and defining characteristics of a class—are positioned near the hyperboloid origin, while specific features—capturing variations within the class—are positioned farther away. This arrangement enables the model to learn generic prototypes that focus on fundamental, class-defining traits, while specific prototypes capture within-class diversity, such as differences in body regions, color patterns, or pose variations. By explicitly modeling this hierarchy, our approach enhances the model's ability to differentiate between classes while also capturing nuanced intra-class variations. While demonstrated on several backbones, our approach has the potential to generalize to other prototypical neural networks and image classifiers. Our contributions are as follows:

- We propose a model-agnostic framework to optimize prototypical neural networks on the Lorentzian manifold in hyperbolic space. We explicitly learn generic and specific prototypes while ensuring hierarchical consistency among prototypes of each class. These prototypes are structured and learned within the hyperboloid to preserve class relationships.
- Extensive experiments across various prototypical architectures and datasets demonstrate that our hyperbolic approach consistently outperforms existing Euclidean-based optimization methods.

2. Related Works

2.1. Prototypical Neural Networks

Prototypical neural networks are inherently interpretable models due to their structure. These models evaluate inputs explicitly based on their similarity to learned discriminative features, or "prototypes", for each class. By visualizing the appearance and location of prototypes, as well as the coefficients relating prototype similarity to the class logits, users can validate the model's decision-making process. ProtoPNet [1] introduced the approach of classification using representative prototypes from each of the classes in the training set and providing visual validation by highlighting feature proximity and localization within input images.

Subsequent work has focused on enhancing interpretability and prototype efficiency. ProtoPool [28] and ProtoPShare [27] reduce the total number of prototypes by sharing them across classes. PIP-Net [22] adopts a self-supervised approach to produce sparse prototypes and human-understandable features, while XProtoNet [13] extends interpretability by allowing different prototype sizes, applying prototypical neural networks successfully to radiology datasets. SPANet [32] further improves explainability by combining part prototypes with semantic concepts, providing clearer interpretations of what each prototype represents. These approaches enhance interpretability by either structuring prototypes more efficiently or aligning them with semantically meaningful concepts, supporting more intuitive and transparent decision-making.

Other approaches have modified the learning formulation to improve performance. TesNet [35] employs a Grassmann manifold to create distinct class subspaces, and ST-ProtoPNet [34] introduces an SVM-like method to learn boundary-supporting prototypes. Prototypical networks have also been integrated into transformer architectures [4], as seen in ProtoPFormer [37], ProtoFormer [5], and most recently, ProtoViT [16]. ProtoTree [21] uses a decision-tree structure to reduce the number of prototype comparisons, enhancing classification efficiency. Recently, ProtoPNeXt [36] demonstrated that cosine similarity and Bayesian tuning could improve ProtoPNet's transparency and performance across various architectures and classification methods.

Beyond interpretability and architectural improvements, recent works have explored hierarchical representations within prototypical networks. MCPNet [33] and HPDR [10] introduce hierarchical prototypes, albeit in different ways. MCPNet learns hierarchical representations through multi-scale prototypes, while HPDR refines feature distributions using hierarchical prototypes in hyperbolic space. These methods leverage hierarchy to better capture structural relationships within data, enhancing representation learning and classification performance.

2.2. Hierarchical Representations

Image datasets inherently exhibit diverse types of hierarchical relationships. One such hierarchy involves clear-ambiguous relationships, where clear images are associated with specific classes, while ambiguous images (e.g., blurred or occluded) are more generic and tend to exhibit features that overlap with multiple classes. Another type of hierarchy is the whole-part relationship, where a global view captures whole objects or scenes, while local views focus on finer details of objects or scene parts. Each global view is associated with multiple local views, allowing local features to be contextualized within the broader scope of the image (e.g. a leaf is part of a tree and a tree is part of a park). Recognizing these hierarchical relationships is essential for understanding the intricate relationships within complex visual objects or scenes [12, 24].

Prior works have proposed explicit methods to represent hierarchical concepts within images. For example, PyramidCLIP [6] captured multi-granular image representations by learning at global (whole image), intermediate (large image patch), and local levels (cropped object images). This approach modeled relationships across these levels using self-supervised learning guided by language, resulting in more separable representation spaces and improved zero-shot classification.

2.3. Hyperbolic Learning

Hyperbolic learning has demonstrated strong potential for capturing hierarchical structures within data [18]. Unlike Euclidean space, which has a flat manifold and struggles to represent hierarchical relationships effectively, hyperbolic manifolds offer a curved geometry that naturally preserves these relationships. It can accurately reflect the tree-like structure of hierarchical data, where leaf nodes are more specialized and represent local features and intermediate nodes represent higher-level features. There are two main approaches to learning visual representations in hyperbolic space: the Poincaré model [11, 12, 17] and the Lorentzian model. The Poincaré model represents hyperbolic space as the interior of a disk (in 2D) or a ball (in higher dimensions) within Euclidean space. Distances and angles in this model are distorted to reflect the negative curvature of hyperbolic geometry. However, the Poincaré approach is challenging to optimize due to its susceptibility to vanishing gradient problems, which hinder the model's convergence [20].

The Lorentzian model is a more recently proposed hyperbolic representation approach that is less prone to vanishing gradient problems, and therefore much easier to optimize [20]. Empirically, approaches using the Lorentzian model have superior performance to those using the Poincaré model [15, 23]. Prior works show that the Lorentzian model leads to a better latent space with hierarchical representations and improvement in image retrieval and classification tasks [2, 26]. Desai et al. [2] used the Lorentz model to enhance vision-language representation learning. Their proposed approach, MERU, learned a contrastive model (i.e. CLIP) in hyperbolic space where hierarchy in visual and language concepts was preserved. MERU adapted the contrastive loss to minimize the proximity between the associated image and text using Lorentzian distance instead of cosine similarity. Additionally, MERU proposed an entailment loss that enables the model to learn that text represents more abstract, generic concepts than corresponding images (e.g., text of 'dog' encompasses many dog images). The entailment loss pushes image embeddings related to a specific text to exist within a cone-shaped space emanating from the text embedding, indicating that the text entails a set of associated images. Inspired by MERU, our work, HAPPI, extends these ideas to prototypical neural networks, tailoring the embedding space and loss functions for prototype-based learning. While prior work has explored hyperbolic prototypes, these approaches typically compute prototypes as the mean of class embeddings in hyperbolic space, rather than explicitly learning class-specific prototypes [7, 8, 12, 14]. In contrast, HAPPI learns distinct, optimizable prototypes directly within the hyperbolic space, ensuring they capture both generic and specific class features beyond simple class averages.

3. Method

3.1. Prototypical Classification Framework

Prototypical methods classify inputs by measuring their similarity to class-specific prototypes. Given an input x, a feature extractor $f(\cdot)$ generates a feature map $f(x) \in \mathbb{R}^{H\times W\times D}$, where H and W are spatial dimensions and D is feature depth. These features are then processed for comparison with prototypes. Let $V_{\text{euc}} \in \mathbb{R}^D$ denote each extracted feature in Euclidean space to be compared with prototypes. Prototypes are denoted as $p_{c,k} \in \mathbb{R}^D$, where c and k represent the class and the index of the prototype within that class, respectively, with typically K prototypes per class. The comparison between V_{euc} and $p_{c,k}$ yields similarity scores $s(V_{\text{euc}}, p_{c,k})$, usually computed using Euclidean distance or cosine similarity. These scores are then fed into a fully connected layer $FC : \mathbb{R}^{K\times C} \rightarrow \mathbb{R}^C$, producing class prediction scores $\hat{y} \in \mathbb{R}^C$, where C is the number of classes. This prototype-based approach enhances interpretability by basing classifications on similarities to known examples, facilitating clearer pattern identification in the data.

3.2. Transition to Hyperbolic Space

Our method extends traditional prototypical networks by projecting the prototypes and extracted features into hyperbolic space, specifically using the Lorentz model of the hy-

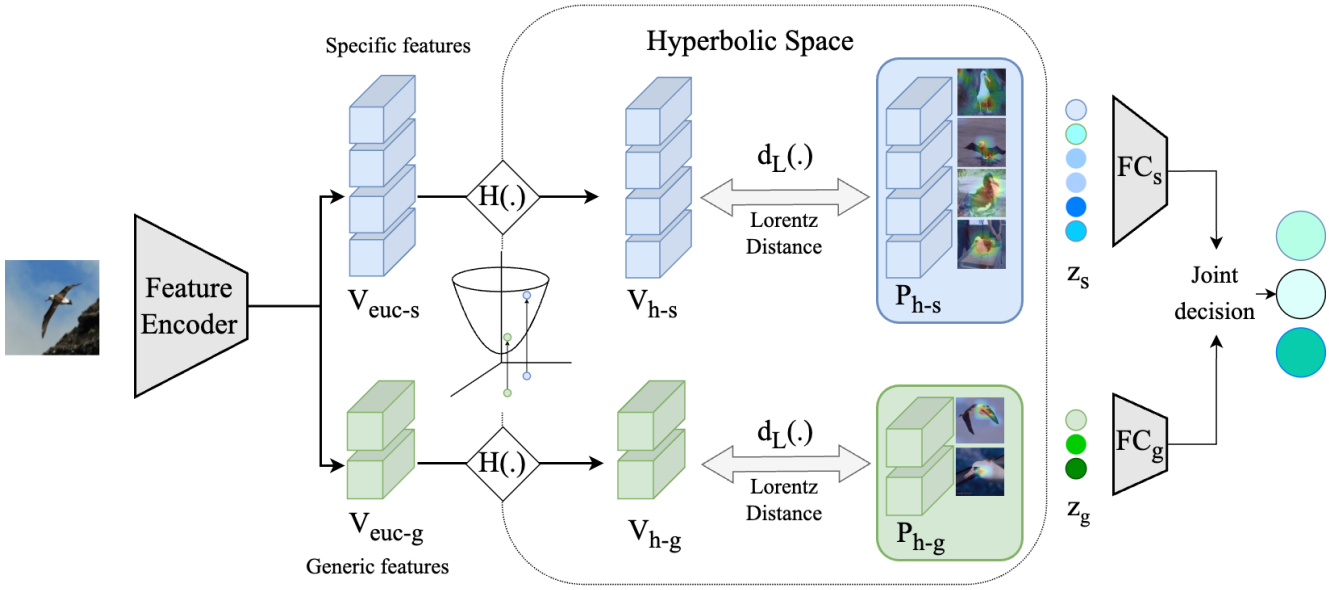

Figure 2. An overview of our proposed approach. We use a given prototypical network's feature encoder to extract both generic and specific features, then lift those features to the hyperboloid using exponential mapping. The lifted features are then compared to their respective prototypes, and similarity scores are generated based on their distances to the prototypes. These similarity scores ultimately result in the activations that are averaged to create the class logits.

perboloid. In this model, the V_{euc} is mapped to a $(D+1)$-dimensional hyperbolic space by adding an additional time dimension. This mapping results in a hyperbolic feature $V_h = [V_{\text{space}}, V_{\text{time}}]$, where $V_{\text{space}} \in \mathbb{R}^D$ represents the spatial components and $V_{\text{time}} \in \mathbb{R}$ is the time dimension [19]. This representation can capture hierarchical relationships and varying levels of detail more effectively than Euclidean space.

To obtain the V_{space}, a mapping operation, shown as $\mathcal{H}(.)$ in Figure 2, projects Euclidean space features V_{euc} onto the hyperboloid. In our case, we use the simplifying assumption from [2] which considers only the exponential map centered at the origin. Under these conditions, the map simplifies to:

$$V_{\text{space}} = \frac{\sinh\left(\sqrt{c}\|V_{\text{euc}}\|\right)}{\sqrt{c}\|V_{\text{euc}}\|} V_{\text{euc}} \tag{1}$$

where c is the negative curvature of the hyperbolic space. Since the hyperbolic features are constrained to reside on the hyperboloid manifold, the time dimension V_{time} can be derived based on V_{space} as follows:

$$V_{\text{time}} = \sqrt{\frac{1}{c} + \|V_{\text{space}}\|^2} \tag{2}$$

As a result, we can effectively map the V_{euc} from Euclidean space into the hyperboloid by calculating both V_{space} and V_{time}. We can then calculate the Lorentzian distance of the resulting features from the learnable prototypes to identify the most prominent prototypes and classify the input image.

For example, let $a = [a_{\text{space}}, a_{\text{time}}]$ and $b = [b_{\text{space}}, b_{\text{time}}]$ denote two points on the hyperboloid. The Lorentzian distance $d_L(a, b)$ is defined as:

$$d_L(a, b) = \frac{1}{\sqrt{c}} \cosh^{-1}(-c\langle a, b\rangle_L), \tag{3}$$

where $\langle \cdot, \cdot \rangle_L$ denotes the Lorentzian inner product. This inner product is defined as:

$$\langle a, b\rangle_L = \langle a_{\text{space}}, b_{\text{space}}\rangle - a_{\text{time}} b_{\text{time}}, \tag{4}$$

where $\langle \cdot, \cdot \rangle$ is the standard Euclidean inner product. This distance metric is foundational in aligning the prototypes with features in the embedding space.

3.3. Hyperbolic-Based Losses and Prototypes

In this section, we introduce the adaptation of prototypical losses and prototype structures to hyperbolic space. We describe the conversion of Euclidean-based clustering and separation losses to their hyperbolic counterparts and introduce generic prototypes for capturing defining and consistent features. We also present an entailment loss inspired by MERU [2] to encourage hierarchical structure in hyperbolic space. Prototype optimization methods (such as clustering and separation losses [1]) are based on hyperbolic distances.

3.3.1. Feature Extraction and Prototype Assignment

We introduce two types of prototypes per class: generic prototypes, which capture defining class features, and specific

prototypes, which capture fine-grained variations such as pose, color, or texture. Generic features are explicitly extracted based on the backbone architecture:

- Transformer-based: We use the [CLS] token to extract generic features, following ProtoPFormer [37].
- CNN-based: Attention maps, interpreted as occurrence maps, are applied to extract generic features, inspired by XProtoNet [13].

In contrast, specific features naturally emerge from the host architecture's feature extraction pipeline and are used directly without additional processing.

For both feature types, we dedicate corresponding prototypes: generic prototypes for generic features and specific prototypes for specific features. Once extracted, we utilize Equations 1 and 2 to project both generic ($V_{\text{euc-g}}$) and specific ($V_{\text{euc-s}}$) features to the hyperbolic space, forming $V_{\text{h-g}}$ and $V_{\text{h-s}}$, respectively. Their corresponding prototypes are also projected, resulting in hyperbolic generic prototypes $P_{\text{h-g}}$ and hyperbolic specific prototypes $P_{\text{h-s}}$.

3.3.2. Prototype-based Classification

Prototype-based classification leverages both generic and specific prototypes to enhance predictive accuracy. We compute the similarities of extracted features to both sets of prototypes. To obtain the final class activation logits, we calculate two separate sets of logits based on these similarities. Each set of similarities is passed through a distinct final layer to produce corresponding logits: z_s for specific prototypes and z_g for generic prototypes. The final activation is a joint decision that is computed as the average of these two sets of logits. This approach allows the model to integrate both generic and specific information for classification, improving its ability to perform effectively across varying levels of granularity.

3.3.3. Clustering and Separation Losses

Clustering and separation losses are designed to encourage the learning of effective prototypes that capture the structure of the data. Clustering loss promotes intra-class compactness by encouraging features to be close to the prototypes of their corresponding classes, while separation loss enhances inter-class separability by pushing features away from prototypes of different classes [1]. Similar to existing prototypical approaches, we cluster extracted specific and generic features that are mapped to the hyperboloid, V_{h}, with the prototypes of the corresponding classes. Likewise, we separate the lifted features from the prototypes of different classes, using the following equations:

$$\mathcal{L}_{\text{clst}} = \min_i d_L(V_t^i, P_i^y), \quad \mathcal{L}_{\text{sep}} = -\mathbb{E}_{c\neq y,i}\left[d_L(V_t), P_i^c)\right], \tag{5}$$

where V_t represents generic (g) or specific (s) feature embeddings, and P_i^c denotes class-specific prototypes. Both clustering and separation criteria, denoted for specific prototypes as $L_{\text{clst}}^{\text{specific}}$ and $L_{\text{sep}}^{\text{specific}}$, and generically as $L_{\text{cluster}}^{\text{generic}}$ and $L_{\text{sep}}^{\text{generic}}$ respectively, utilize the Lorentzian distance, Equation 3, as $d_L(V_{\text{h}}, p)$ where p represents the prototype of interest. This adaptation maintains clustering and separation fidelity in hyperbolic space.

3.3.4. Entailment Loss

Entailment loss is designed to encourage a hierarchical structure in the hyperbolic space by enforcing relationships between specific and generic prototypes. It ensures that each specific prototype is entailed by at least one generic prototype of its class, creating a coherent hierarchy of representations. To encourage this hierarchical encoding, we implement an entailment loss based on MERU's formulation [2]. This is achieved by minimizing the exterior angle of $\text{ext}(V_{\text{h-g}}, V_{\text{h-s}})$ relative to the half-aperture $\text{aper}(V_{\text{h-g}})$:

$$L_{\text{entail}}(V_{\text{h-g}}, V_{\text{h-s}}) = \max(0, \text{ext}(V_{\text{h-g}}, V_{\text{h-s}}) - \text{aper}(V_{\text{h-g}})). \tag{6}$$

The half-aperture of a cone for entailment relationships is calculated as:

$$\text{aper}(a) = \sin^{-1}\left(\frac{2R}{\sqrt{c}\|a_{\text{space}}\|}\right), \tag{7}$$

where R is a constant that controls the boundary conditions near the origin. Furthermore, the exterior angle between two points on the hyperboloid is given by:

$$\text{ext}(a, b) = \cos^{-1}\left(\frac{b_{\text{time}} + a_{\text{time}} \cdot c\langle a, b\rangle_L}{\|a_{\text{space}}\|\sqrt{(c\langle a, b\rangle_L)^2 - 1}}\right). \tag{8}$$

3.3.5. Combined Loss Function

The overall loss function $\mathcal{L}$ uses cross entropy, $\mathcal{L}_{CE}$, for classification and integrates generic and specific clustering and separation losses, entailment loss as shown below:

$$\begin{aligned}\mathcal{L} = \mathcal{L}_{\text{CE}} &+ \lambda_{\text{clst\_g}}\mathcal{L}_{\text{clst}}^{\text{generic}} + \lambda_{\text{sep\_g}}\mathcal{L}_{\text{sep}}^{\text{generic}} \\ &+ \lambda_{\text{clst\_s}}\mathcal{L}_{\text{clst}}^{\text{specific}} + \lambda_{\text{sep\_s}}\mathcal{L}_{\text{sep}}^{\text{specific}} + \lambda_{\text{entail}}\mathcal{L}_{\text{entail}}\end{aligned} \tag{9}$$

where each λ denotes the coefficient corresponding to each loss value.

4. Experiments

4.1. Datasets

We used two different datasets for our evaluations: CUB-200-2011 and Oxford-IIIT Pets.

- **CUB-200-2011 (Caltech-UCSD Birds 200)** [31]: The CUB-200-2011 dataset is a widely-used benchmark for fine-grained classification tasks, focusing specifically on bird species recognition. It consists of 11,788 images across 200 bird species, each with a varying number of

images. The dataset is split into 5,994 images for training and 5,794 images for testing.

- **Oxford-IIIT Pets** [25]: The Oxford-IIIT Pets dataset is used for pet breed classification and includes a balanced mix of cat and dog breeds. It contains 7,349 images spanning 37 breeds, with around 200 images per breed. The dataset is split into 3,680 images for training and 3,669 images for testing.

While each of these datasets has additional attribute annotations such as bounding boxes or segmentations, we only used the labels to train our models. We selected these datasets for their frequent use in prototypical neural network research and their diversity: CUB includes 200 classes, while Pets includes 37 classes.

4.2. Implementation Details

To ensure a comprehensive evaluation, we selected a range of backbone architectures that cover diverse structural approaches. Specifically, we included part-based prototypical neural networks like ProtoPNet and ST-ProtoPNet, which use patch-based prototypes; XProtoNet, which uses adaptable prototype sizes; and ProtoPFormer, which uses transformer-based backbone. For CNN-based evaluations, all models were tested with a ResNet-50 backbone [9], while different transformer-based models were also evaluated to assess performance across architectural types. Additionally, a black-box version of the backbone classification model was evaluated for comparison, as shown in the results section. We also evaluated MCPNet [33], a state-of-the-art hierarchical prototypical model in Euclidean space, and PIPNet [22], a state-of-the-art prototypical model. All evaluations were performed using 10 specific prototypes per class, a standard choice in prototypical neural network literature, along with 2 generic prototypes per class, determined empirically. The entailment loss coefficient, λ_{entail}, was set to 0.1 and R was set to 0.1. The loss weights, λ, for the remaining components were selected to match the values used in the baseline model. Input images were resized to 224x224, with shear and flip transformations for data augmentation, while test images were resized to 224x224 without additional cropping to maintain consistency.

For ProtoPFormer, we adopted their approach of using the CLS token as a generic prototype and image tokens as specific prototypes. Instead of ProtoPFormer's prototypical part concentration (PPC) loss, we implemented our clustering and separation loss functions. While PPC loss focuses on concentrating prototypes on distinct, centralized representative parts for each class, our clustering and separation losses also enforce the learning of distinct and representative prototypes, with a formulation that translates more effectively to hyperbolic space. For ProtoPNet and XProtoNet, we report two accuracy metrics: end-to-end (E2E) training accuracy where the prototypes and features are optimized jointly, and multi-stage training accuracy. Multi-stage training consists of initial prototype warm-up epochs, followed by joint training, and then final layer optimization for additional epochs. To ensure a fair comparison, we retained the original backbone feature learning rates from each model's source paper. Further implementation details can be found in the supplementary materials.

4.3. Quantitative Results

Our method consistently delivers strong quantitative performance across prototypical architectures, often surpassing or matching state-of-the-art baselines. This highlights the effectiveness of our hierarchical approach in enhancing feature representation.

As shown in Table 1, HAPPI significantly improves performance over simpler prototypical architectures like ProtoPNet. On CUB-200-2011, ProtoPNet achieves 45.24% accuracy, whereas integrating HAPPI boosts it to 61.44%, a notable 16.2% improvement. Similarly, on Oxford-IIIT Pets, hyperbolic ProtoPNet achieves 88.25%, up from the baseline's 54.65%, an increase of over 33.6%. These results demonstrate HAPPI's ability to mitigate the limitations of simpler methods by capturing complex hierarchical relationships.

For more advanced architectures such as XProtoNet and ST-ProtoPNet, which already incorporate sophisticated feature representations, performance gains are smaller. On CUB-200-2011, XProtoNet improves from 73.82% to 75.46% with HAPPI, while ST-ProtoPNet sees a marginal increase from 86.54% to 87.21%. A similar trend is observed for transformer-based models like ProtoPFormer with DeiT-Ti and DeiT-S backbones, where improvements range between 0.5% and 4.0%. This suggests that while HAPPI enhances all architectures, its impact is most pronounced in simpler baselines, where hierarchical modeling in hyperbolic space provides the greatest benefit. This behavior likely reflects a saturation of feature representation in advanced architectures, where additional hierarchical structuring yields diminishing returns in accuracy. However, even for these high-performing models, HAPPI still enhances interpretability by structuring feature representations hierarchically, ensuring that both generic and specific prototypes contribute meaningfully to decision-making.

Despite its consistent advantages, HAPPI does not always yield improvements. In some configurations, such as ProtoPFormer with a DeiT-Ti backbone on cropped CUB-200-2011, accuracy remains unchanged at 79.65%. A likely explanation is that DeiT-Ti, being the smallest model tested, lacks the capacity to fully leverage hyperbolic representations, limiting performance gains. This suggests that HAPPI's effectiveness depends on model complexity and dataset-specific optimization strategies. Future work could explore adaptive hyperbolic scaling for smaller backbones

Backbone	Methods	CUB		CUB Cropped		Pets	
		Baseline (Euclidean)	HAPPI (Hyperbolic)	Baseline (Euclidean)	HAPPI (Hyperbolic)	Baseline (Euclidean)	HAPPI (Hyperbolic)
ResNet50 [9]	Black-box	75.16	-	76.34	-	90.49	-
	PIP-Net [22]	73.52	-	82.00	-	88.50	-
	MCPNet [33]	74.28	-	70.78	-	90.32	-
	ProtoPNet [1]	45.24	**61.44**	62.30	**80.51**	54.65	**88.25**
	ProtoPNet - E2E	67.04	**76.44**	71.53	**82.39**	77.35	**88.34**
	XProtoNet [13]	73.82	**75.46**	80.84	**81.03**	89.48	89.40
	XProtoNet - E2E	75.87	**77.91**	82.09	**82.90**	90.00	**91.50**
	ST-ProtoPNet [34]	86.54	**87.21**	86.94	86.92	77.43	**81.93**
DeiT-Ti [29]	Black-box	75.27	-	78.13	-	88.85	-
	ProtoPFormer [37]	77.30	76.18	79.65	79.65	86.89	86.89
DeiT-S [29]	Black-box	79.86	-	82.53	-	92.23	-
	ProtoPFormer [37]	77.68	**78.67**	80.60	**84.59**	88.72	**91.25**
CaiT-XXS-24 [30]	Black-box	80.76	-	82.56	-	92.86	-
	ProtoPFormer [37]	80.86	80.81	82.24	**83.91**	91.47	**91.88**

Table 1. Classification accuracy (%) on full / cropped CUB-200-2011 and Oxford-IIIT Pets using various backbones and methods.

or refined tuning strategies to address these limitations.

Compared to black-box baselines, HAPPI achieves competitive accuracy while significantly improving interpretability. On CUB-200-2011, the ResNet-50 black-box model attains 75.16% accuracy, whereas HAPPI-enhanced ProtoPNet reaches 76.44%, a modest improvement. However, this gain comes with the added benefit of interpretability, as HAPPI's generic and specific prototypes provide insight into the model's decision-making process. Similarly, on Oxford-IIIT Pets, hyperbolic XProtoNet achieves 91.50% accuracy, surpassing the ResNet-50 black-box model's 90.49%. These results highlight HAPPI's ability to balance high performance with transparency, offering a compelling alternative to opaque black-box models.

4.4. Qualitative Results

To illustrate the qualitative effectiveness of HAPPI, we provide visualizations of generic and specific prototypes in Figure 3, learned using the XProtoNet-E2E architecture on the Oxford-IIIT Pets [25] dataset. The figure includes four classes: Abyssinian (top left), Bengal (bottom left), American Bulldog (top right), and Basset Hound (bottom right). Each group has two rows—the bottom row shows a heatmap overlay highlighting attended regions, while the top row masks less relevant areas by shadowing them.

Generic prototypes capture consistent features that broadly distinguish classes. Across all classes, they focus on the nose and mouth area, a key trait distinguishing not only between cats and dogs but also among different breeds within each group. The American Bulldog's generic prototypes highlight facial wrinkles and white fur, while the Basset Hound's emphasize long ears and a brown head. The Abyssinian cat's prototypes focus on fur texture, nose, and eyes, while the Bengal cat's emphasize its distinct tiger-like fur pattern.

In contrast, specific prototypes capture variations in body shapes, poses, and colors. The American Bulldog's specific prototypes highlight a different color combination—white and brown—unlike its generic prototypes, which focus only on white fur. In the last specific prototypes of both the American Bulldog and Bengal cat, a diffused heatmap or less pronounced shadowing suggests a broader focus, capturing some background details.

Overall, generic prototypes identify prominent class-defining features, while specific prototypes capture fine-grained variations, ensuring a hierarchical, multi-level understanding of class distinctions. This layered approach enhances the model's capacity to differentiate classes across various levels of abstraction.

Specific prototypes, in contrast, provide complementary information that captures specific body shapes, poses, and angles, adding variable, fine-grained detail that further supports classification. This layered approach, where generic prototypes identify defining distinctions and specific prototypes capture detailed variations, enhances the model's capacity to differentiate classes across multiple levels of abstraction.

4.5. Ablation Study

We performed an ablation study using XProtoNet-E2E to assess the impact of hyperbolic space and hierarchical structuring. As shown in Table 2, transitioning to hyperbolic space and introducing generic and specific prototypes improved performance over the Euclidean base-

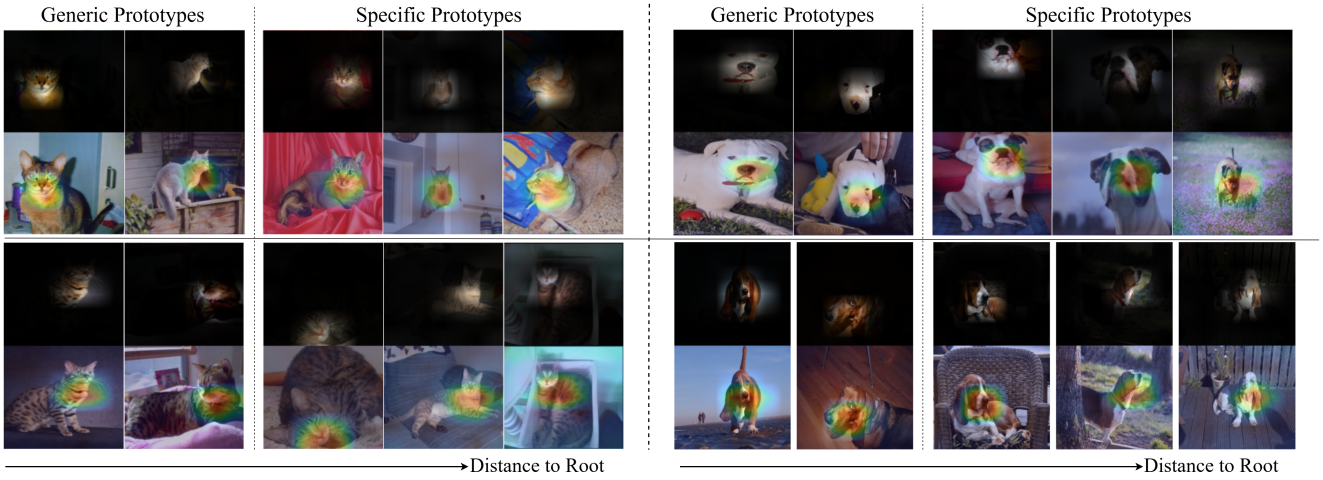

Figure 3. Qualitative visualization of learned prototypes in the XProtoNet-E2E architecture for the Oxford-IIIT Pets dataset [25]. Generic prototypes capture defining class features, such as a 'white American Bulldog' (top right), while specific prototypes capture intra-class variations like color differences. As prototype distance from the hyperboloid root increases, it covers broader areas or more variations.

line, even without entailment loss ($\lambda_{entail} = 0$). Increasing λ_{entail} slightly reduced accuracy, highlighting a tradeoff between classification performance and hierarchical structuring, consistent with observations in [2]. Larger λ_{entail} values enhanced structure but posed optimization challenges. With $\lambda_{entail} = 0.1$, the model maintained high accuracy while preserving a meaningful hierarchy, making it the preferred setting.

To further analyze the effect of prototype allocation, we varied the number of generic (K_g) and specific (K_s) prototypes per class, as shown in Table 3. Increasing the number of prototypes generally improved performance, but gains plateaued around $K_g = 2$ and $K_s = 5$. Notably, models with more specific prototypes performed better than those with more generic ones, reinforcing the importance of capturing intra-class variability. The setup with $K_g = 5$ and $K_s = 1$ performed worse than $K_g = 1$ and $K_s = 5$, indicating that generic prototypes alone are not sufficient for fine-grained classification. When specific prototypes were sufficient, reducing the number of generic prototypes from 5 to 2 further improved optimization stability. These results suggest that the ideal prototype configuration depends on the complexity of the dataset. A careful balance between generic and specific prototypes is crucial, as different datasets may require different levels of hierarchical structuring to achieve optimal performance.

Method	λ_{entail} 0	0.1	0.2	0.5
Euclidean	90.00	-	-	-
Hyperbolic	91.50	91.50	91.47	91.20

Table 2. Ablation study, using XProtoNet-E2E, on the use of hyperbolic space and entailment in terms of Accuracy (%) on the Pets dataset. The first row refers to the Euclidean model which has 10 prototypes per class, while the other rows are hyperbolic versions with 2 generic and 10 specific prototypes.

		K_s 1	2	5	10
K_g	1	88.83	90.27	91.06	91.06
	2	90.19	90.90	91.91	91.50
	5	90.79	88.53	91.77	91.52

Table 3. Accuracy (%) by number of generic and specific prototypes on the Pets dataset.

5. Conclusion

In this paper, we introduced HAPPI, a model-agnostic approach for adapting prototypical networks to hyperbolic space to effectively learn hierarchical image representations. Our method showed quantitative improvements over single-scale Euclidean baselines across diverse datasets, highlighting its ability to capture complex, structured relationships within the data. Furthermore, qualitative analyses reveal that HAPPI provides an interpretable understanding of image hierarchies, allowing insights into the features the model attends to when making predictions. Our findings suggest that hyperbolic prototypical networks, as exemplified by HAPPI, hold significant potential for applications requiring both hierarchical understanding and interpretability in visual tasks.

References

[1] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: deep learn-

ing for interpretable image recognition. *Advances in neural information processing systems*, 32, 2019. 1, 2, 4, 5, 7

[2] Karan Desai, Maximilian Nickel, Tanmay Rajpurohit, Justin Johnson, and Ramakrishna Vedantam. Hyperbolic image-text representations. *arXiv (Cornell University)*, 2023. 3, 4, 5, 8

[3] Jon Donnelly, Alina Jade Barnett, and Chaofan Chen. Deformable protopnet: An interpretable image classifier using deformable prototypes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10265–10275, 2022. 1

[4] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2

[5] Ashkan Farhangi, Ning Sui, Nan Hua, Haiyan Bai, Arthur Huang, and Zhishan Guo. Protoformer: Embedding prototypes for transformers. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 447–458. Springer, 2022. 2

[6] Yuting Gao, Jinfeng Liu, Zihan Xu, Jun Zhang, Ke Li, Rongrong Ji, and Chunhua Shen. Pyramidclip: Hierarchical feature alignment for vision-language model pretraining. *arXiv (Cornell University)*, 2022. 3

[7] Mina Ghadimi Atigh et al. Hyperbolic busemann learning with ideal prototypes. *Advances in neural information processing systems*, 34:103–115, 2021. 3

[8] Yunhui Guo, Xudong Wang, Yubei Chen, and Stella X Yu. Clipped hyperbolic classifiers are super-hyperbolic classifiers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11–20, 2022. 3

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6, 7

[10] Chengyang Hu, Ke-Yue Zhang, Taiping Yao, Shouhong Ding, and Lizhuang Ma. Rethinking generalizable face anti-spoofing via hierarchical prototype-guided distribution refinement in hyperbolic space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1032–1041, 2024. 2

[11] Chengyang Hu, Ke-Yue Zhang, Taiping Yao, Shouhong Ding, and Lizhuang Ma. Rethinking generalizable face anti-spoofing via hierarchical prototype-guided distribution refinement in hyperbolic space. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1032–1041. IEEE, 2024. 3

[12] Valentin Khrulkov, Leyla Mirvakhabova, Evgeniya Ustinova, Ivan Oseledets, and Victor Lempitsky. Hyperbolic image embeddings. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6418–6428, 2020. 3

[13] Eunji Kim, Siwon Kim, Minji Seo, and Sungroh Yoon. Xprotonet: diagnosis in chest radiography with global and local explanations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15719–15728, 2021. 2, 5, 7

[14] Sungyeon Kim, Boseung Jeong, and Suha Kwak. Hier: Metric learning beyond class labels via hierarchical regularization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19903–19912, 2023. 3

[15] Marc Law, Renjie Liao, Jake Snell, and Richard Zemel. Lorentzian distance learning for hyperbolic representations. In *Proceedings of the 36th International Conference on Machine Learning*, pages 3672–3681. PMLR, 2019. 3

[16] Chiyu Ma, Jon Donnelly, Wenjun Liu, Soroush Vosoughi, Cynthia Rudin, and Chaofan Chen. Interpretable image classification with adaptive prototype-based vision transformers. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 2

[17] Paolo Mandica, Luca Franco, Konstantinos Kallidromitis, Suzanne Petryk, and Fabio Galasso. Hyperbolic learning with multimodal large language models. *arXiv preprint arXiv:2408.05097*, 2024. 3

[18] Pascal Mettes, Mina Ghadimi Atigh, Martin Keller-Ressel, Jeffrey Gu, and Serena Yeung. Hyperbolic deep learning in computer vision: A survey. *International Journal of Computer Vision*, 132(9):3484–3508, 2024. 3

[19] Hermann Minkowski. Raum und Zeit. *Physikalische Zeitschrift*, 1908. 4

[20] Gal Mishne, Zhengchao Wan, Yusu Wang, and Sheng Yang. The numerical stability of hyperbolic representation learning. *arXiv (Cornell University)*, 2022. 3

[21] Meike Nauta, Ron Van Bree, and Christin Seifert. Neural prototype trees for interpretable fine-grained image recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14933–14943, 2021. 2

[22] Meike Nauta, Jörg Schlötterer, Maurice Van Keulen, and Christin Seifert. Pip-net: Patch-based intuitive prototypes for interpretable image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2744–2753, 2023. 1, 2, 6, 7

[23] Maximilian Nickel and Douwe Kiela. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. *arXiv (Cornell University)*, 2018. 3

[24] Avik Pal, Max van Spengler, Guido Maria D'Amely di Melendugno, Alessandro Flaborea, Fabio Galasso, and Pascal Mettes. Compositional entailment learning for hyperbolic vision-language models. *arXiv preprint arXiv:2410.06912*, 2024. 3

[25] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE, 2012. 6, 7, 8

[26] Sameera Ramasinghe, Violetta Shevchenko, Gil Avraham, and Ajanthan Thalaiyasingam. Accept the modality gap: An exploration in the hyperbolic space. pages 27253–27262. IEEE, 2024. 3

[27] Dawid Rymarczyk, Łukasz Struski, Jacek Tabor, and Bartosz Zieliński. Protopshare: Prototypical parts sharing for similarity discovery in interpretable image classification. In *Proceedings of the 27th ACM SIGKDD Conference on Knowl-*

edge Discovery & Data Mining, pages 1420–1430, 2021. 1, 2

[28] Dawid Rymarczyk, Łukasz Struski, Michał Górszczak, Koryna Lewandowska, Jacek Tabor, and Bartosz Zieliński. Interpretable image classification with differentiable prototypes assignment. In *European Conference on Computer Vision*, pages 351–368. Springer, 2022. 2

[29] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021. 7

[30] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 32–42, 2021. 7

[31] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 5

[32] Qiyang Wan, Ruiping Wang, and Xilin Chen. Interpretable object recognition by semantic prototype analysis. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 800–809, 2024. 2

[33] Bor-Shiun Wang, Chien-Yi Wang, and Wei-Chen Chiu. Mcpnet: An interpretable classifier via multi-level concept prototypes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10885–10894, 2024. 2, 6, 7

[34] Chong Wang, Yuyuan Liu, Yuanhong Chen, Fengbei Liu, Yu Tian, Davis McCarthy, Helen Frazer, and Gustavo Carneiro. Learning support and trivial prototypes for interpretable image classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2062–2072, 2023. 2, 7

[35] Jiaqi Wang, Huafeng Liu, Xinyue Wang, and Liping Jing. Interpretable image recognition by constructing transparent embedding space. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 895–904, 2021. 2

[36] Frank Willard, Luke Moffett, Emmanuel Mokel, Jon Donnelly, Stark Guo, Julia Yang, Giyoung Kim, Alina Jade Barnett, and Cynthia Rudin. This looks better than that: Better interpretable models with protopnext. *arXiv preprint arXiv:2406.14675*, 2024. 2

[37] Mengqi Xue, Qihan Huang, Haofei Zhang, Lechao Cheng, Jie Song, Minghui Wu, and Mingli Song. Protopformer: Concentrating on prototypical parts in vision transformers for interpretable image recognition. *arXiv preprint arXiv:2208.10431*, 2022. 2, 5, 7