THE EMERGENCE OF PROTOTYPICALITY: UNSUPERVISED FEATURE LEARNING IN HYPERBOLIC SPACE

Anonymous authors
Paper under double-blind review

ABSTRACT

Prototypicality is extensively studied in machine learning and computer vision. However, there is still no widely accepted definition of prototypicality. In this paper, we first propose to define prototypicality based on the concept of congealing. Then, we develop a novel method called HACK to automatically discover prototypical examples from the dataset. HACK conducts unsupervised prototypicality learning in Hyperbolic space with sphere pACKing. HACK first generates uniformly packed particles in the Poincaré ball of hyperbolic space and then assigns the image uniquely to each particle. Due to the geometrical property of hyperbolic space, prototypical examples naturally emerge and tend to locate in the center of the Poincaré ball. HACK naturally leverages hyperbolic space to discover prototypical examples in a data-driven fashion. We verify the effectiveness of the method with synthetic dataset and natural image datasets. Extensive experiments show that HACK can naturally discover the prototypical examples without supervision. The discovered prototypical examples and atypical examples can be used to reduce sample complexity and increase model robustness.

1 INTRODUCTION

Not all instances are created equal. Some instances are more representative of the class and some instances are outliers or anomalies. Representative examples can be viewed as prototypes and used for interpretable machine learning (Bien & Tibshirani, 2011), curriculum learning (Bengio et al., 2009) and learning better decision boundaries (Carlini et al., 2018). With prototypical examples, we can also conduct classification with few or even one example (Miller et al., 2000). Given an image dataset, thus it is desirable to organize the examples based on prototypicality.

If the features of the images are given, it is relatively easy to find the prototypes by examining the density peaks of the feature distribution. If the features are not given, to discover prototypical examples without supervision is difficult: there is no universal definition or simple metric to assess the prototypicality of the examples. A naive method to address this problem is to examine the gradient magnitude (Carlini et al., 2018). However, this approach is shown to have a high variance which is resulted from different training setups (Carlini et al., 2018). Some methods address this problem from the perspective of adversarial robustness (Stock & Cisse, 2018; Carlini et al., 2018): prototypical examples should be more adversarially robust. However, the selection of the prototypical examples highly depends on the adversarial method and the metric used in adversarial attack. Several other methods exist for this problem but they are either based on heuristics or lack a proper justification (Carlini et al., 2018).

In this paper, we first introduce a way of obtaining prototypical examples from image congealing (Miller et al., 2000). Congealing is the process of jointly aligning a set of images. The congealed images are transformed to better align with the average image and thus more typical. We further propose a novel method, called HACK, by leveraging the geometry of *hyperbolic space* for unsupervised learning. Hyperbolic space is non-Euclidean space with constant non-negative curvature Anderson (2006). Different from Euclidean space, hyperbolic space can represent hierarchical relation with low distortion. Poincaré ball model is one of the most commonly used models for hyperbolic space (Nickel & Kiela, 2017b). One notable property of Poincaré ball model is that the distance to the origin grows exponentially as we move towards the boundary. Thus, the points located in the center of the ball are close to all the other points while the points located close to the boundary are infinitely

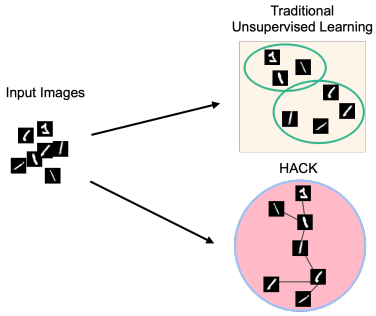


Figure 1: **Different from the existing unsupervised learning methods which aim to group examples via semantic similarity, HACK organizes images in hyperbolic space in a hierarchical manner.** The typical images are at the center of the Poincaré ball and the atypical images are close to the boundary of the Poincaré ball.

far away from other points. With unsupervised learning in hyperbolic space, HACK can learn features which capture both visual similarity and prototypicality(Figure 1).

HACK optimizes the organization of the dataset by assigning the images to a set of uniformly distributed particles in hyperbolic space. The assignment is done by minimizing the total hyperbolic distance between the image features and the particles via Hungarian algorithm. The prototypicality arises naturally based on the distance of the example to other examples. Prototypical examples tend to locate in the center of the Poincaré ball and atypical examples tend to locate close to the boundary. Hyperbolic space readily facilitates such an organization due to property of the hyperbolic distance.

In summary, the **contributions** of the papers are,

- We propose the first unsupervised feature learning method to learn features which capture both visual similarity and prototypicality. The positions of the features reflect prototypicality of the examples.
- The proposed method HACK assigns images to particles that are uniformly packed in hyperbolic space. HACK fully exploits the property of hyperbolic space and prototypicality arises naturally.
- We ground the concept of prototypicality based on congealing which conforms to human visual perception. The congealed examples can be used to replace the original examples for constructing datasets with known prototypicality. We validate the effectiveness of the method by using a synthetic data with natural and congealed images. We further apply the proposed method to commonly used image datasets to reveal prototypicality.
- The discovered prototypical and atypical examples are shown to reduce sample complexity and increase robustness of the model.

2 Related Work

Prototypicality. The study of prototypical examples in machine learning has a long history. In Zhang (1992), the authors select typical instances based on the fact that typical instances should be representative of the cluster. In Kim et al. (2016), prototypical examples are defined as the examples that have minimum maximum mean discrepancy within the data. Li et al. (Li et al., 2018) propose to discover prototypical examples by architectural modifications: the dataset is first projected onto a low-dimensional manifold and a prototype layer is used to minimize the distance between inputs and the prototypes on the manifold. The robustness to adversarial attacks are also used as a criteria for prototypicality (Stock & Cisse, 2018). In Carlini et al. (2018), the authors propose multiple metrics for prototypicality discovery. For example, the features of prototypical examples should be consistent across different training setups. However, these metrics usually depend heavily on the training setups and hyperparameters used for training. The idea of prototypicality is also extensively studied in meta-learning for one-shot or few-shot classification (Snell et al., 2017). No existing works address the prototypicality discovery problem in a data-driven fashion. Our proposed HACK naturally exploits hyperbolic space to organize the images based on prototypicality.

Unsupervised Learning in Hyperbolic Space. Learning features in hyperbolic space has shown to be useful for many machine learning problems (Nickel & Kiela, 2017a; Ganea et al., 2018). One

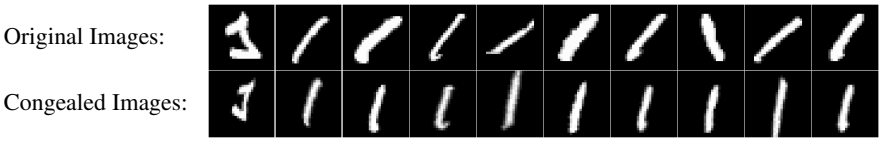


Figure 2: **Congealed images are more typical than the original images.** First row: sampled original images. Second row: the corresponding congealed images.

useful property is that hierarchical relations can be embedded in hyperbolic space with low distortion (Nickel & Kiela, 2017a). A generalized version of the normal distribution called wrapped normal distribution is proposed for modeling distribution of points in hyperbolic space (Nagano et al., 2019). The proposed wrapped normal distribution is used as the latent space for constructing hyperbolic variational autoencoders (VAEs) (Kingma & Welling, 2013). Poincaré VAEs is constructed in Mathieu et al. (2019) with a similar idea to Nagano et al. (2019) by replacing the standard normal distribution with hyperbolic normal distribution. Unsupervised 3D segmentation (Hsu et al., 2020) and instance segmentation (Weng et al., 2021) are conducted in hyperbolic space via hierarchical hyperbolic triplet loss. CO-SNE (Guo et al., 2021a) is recently proposed to visualize high-dimensional hyperbolic features in a two-dimensional hyperbolic space. Although hyperbolic distance facilitates the learning of hierarchical structure, how to leverage hyperbolic space for unsupervised prototypicality discovery is not explored in the current literature.

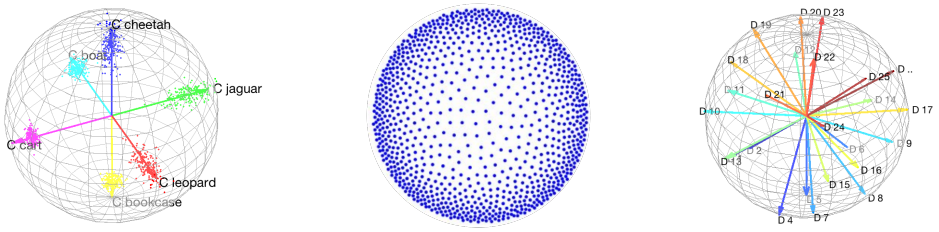
Sphere Packing. The problem of sphere packing is to pack a set of particles as densely as possible in a space (Conway & Sloane, 2013). Sphere packing can be served as a toy model for granular materials and has applications in information theory (Shannon, 2001) to find error-correcting codes (Cohn, 2016). Sphere packing is difficult due to multiple local minima, the curse of high-dimensionality and complicated geometrical configurations. Packing in hyperbolic space is also studied in the literature. It is given in Böröczky (1978) a universal upper bound for the density of sphere packing in an n-dimensional hyperbolic space when $n \geq 2$. We are interested in generating uniform packing in a two-dimensional hyperbolic space. Uniformity has been shown to be a useful criterion for learning good features on the hypersphere (Wang & Isola, 2020). We opt to find the configuration with an optimization procedure which is easily applicable even with thousands of particles.

3 Overview

Given existing features $\{f(v_i)\}$ which are obtained by applying a feature extractor for each instance v_i, we can find the prototypical examples by examining the density peaks via techniques from density estimation. For example, the K-nearest neighbor density (K-NN) estimation (Fix & Hodges, 1989) is defined as,

$$p_{knn}(v_i, k) = \frac{k}{n} \frac{1}{A_d \cdot D^d(v_i, v_{k(i)})} \tag{1}$$

where d is the feature dimension, $A_d = \pi^{d/2}/\Gamma(d/2+1)$, $\Gamma(x)$ is the Gamma function and $k(i)$ is the kth nearest neighbor of example v_i. The nearest neighbors can be found by computing the distance between the features. However, different training setups can induce different feature spaces, which in turn lead to different conclusions of prototypicality. Our goal is to learn features that naturally reflect prototypicality of the examples. We ground our concept of prototypicality based on congealing (Miller et al., 2000). In particular, we define prototypical examples in the *pixel space* by examining the distance of the images to the average image in the corresponding class. Our idea is based on a traditional computer vision technique called image alignment (Szeliski et al., 2007) which aims to find correspondences across images. During congealing (Miller et al., 2000), a set of images are transformed to be jointly aligned by minimizing the joint pixelwise entropies. The congealed images are more prototypical: they are better aligned with the average image. Thus, we have a simple way to transform an atypical example to a typical example (see Figure 2). This is useful since given an unlabeled image dataset the typicality of the examples are unknown, congealing examples can be naturally served as examples with known typicality and be used as a validation for the effectiveness of our method.



a) Supervised classification with fixed known targets **b)** Our unsupervised feature learning with fixed but *unknown* targets **c)** Metric feature learning with unknown targets

Figure 3: **The proposed HACK has a predefined geometrical arrangement and allows the images to be freely assigned to any particle.** a) Standard supervised learning has predefined targets. The image is only allowed to be assigned to the corresponding target. b) HACK packs particles uniformly in hyperbolic space to create initial seeds for organization. The images are assigned to the particles based on their prototypicality and semantic similarities. c) Standard unsupervised learning has no predefined targets and images are clustered based on their semantic similarities.

4 Unsupervised Feature Representation in Hyperbolic Space

We aim to develop a method which can automatically discover prototypical examples unsupervisedly. In particular, we conduct unsupervised learning in hyperbolic space with sphere packing (Figure 5). We specify where the targets should be located ahead of training with uniform packing in hyperbolic space, which by design are maximally evenly spread out in hyperbolic space. The uniformly distributed particles guide feature learning to achieve maximum instance discrimination (Wu et al., 2018).

HACK figures out which instance should be mapped to which target through bipartite graph matching as a global optimization procedure. During training HACK minimizes the total hyperbolic distances between the mapped image point (in the feature space) and the target, those that are more typical naturally emerge closer to the origin of Poincaré ball. Prototypicality comes for free as a result of self-organization. HACK differs from the existing learning methods in several aspects (Figure 3). Different from supervised learning, HACK allows the image to be assigned to *any* target (particle). This enables exploration of natural organizations of the data. Different from existing unsupervised learning learning method, HACK specifies a predefinted geometrical organization which encourages the corresponding structure to be emerged from the dataset. Existing methods are not applicable for prototypicality discovery without supervision due to their aforementioned limitations.

Section 4.1 gives the background on hyperbolic space. Section 4.2 describes the steps for generating uniformly distributed particles in hyperbolic space. Section 4.3 delineates the details of hyperbolic instance assignment via Hungarian algorithm.

4.1 Poincaré Ball Model for Hyperbolic Space

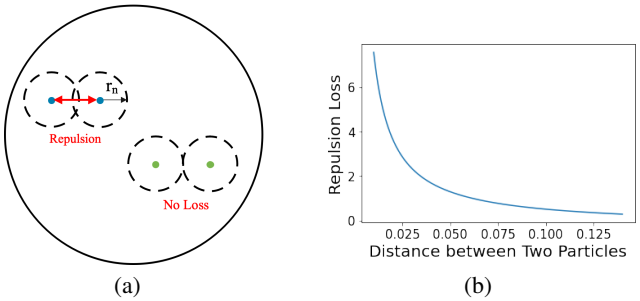
Hyperbolic space. Euclidean space has a curvature of zero and a hyperbolic space is a Riemannian manifold with a constant negative curvature.

Poincaré Ball Model for Hyperbolic Space. There are several isometrically equivalent models for visualizing hyperbolic space with Euclidean representation. The Poincaré ball model is the commonly used one in hyperbolic representation learning (Nickel & Kiela, 2017b). The n-dimensional Poincaré ball model is defined as $(\mathbb{B}^n, \mathfrak{g}_{\mathbf{x}})$, where $\mathbb{B}^n = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}\| < 1\}$ and $\mathfrak{g}_{\mathbf{x}} = (\gamma_{\mathbf{x}})^2 I_n$ is the Riemannian metric tensor. $\gamma_{\mathbf{x}} = \frac{2}{1-\|\mathbf{x}\|^2}$ is the conformal factor and I_n is the Euclidean metric tensor.

Hyperbolic Distance. Given two points $\boldsymbol{u} \in \mathbb{B}^n$ and $\boldsymbol{v} \in \mathbb{B}^n$, the hyperbolic distance is defined as,

$$d_{\mathbb{B}^n}(\boldsymbol{u}, \boldsymbol{v}) = \operatorname{arcosh}\left(1 + 2\frac{\|\boldsymbol{u} - \boldsymbol{v}\|^2}{(1 - \|\boldsymbol{u}\|^2)(1 - \|\boldsymbol{v}\|^2)}\right) \tag{2}$$

where arcosh is the inverse hyperbolic cosine function and $\|\cdot\|$ is the usual Euclidean norm.



(a) (b)

Figure 4: **The proposed repulsion loss is used to generate uniformly packed particles in hyperbolic space.** (a) If the distance between two particles are within $r_{n,r}$, minimizing the repulsion loss would push the two particles away. (b) The repulsion loss is larger when the two particles become closer.

Hyperbolic distance has the unique property that it grows exponentially as we move towards the boundary of the Poincaré ball. In particular, the points on the circle represents points in the infinity. Hyperbolic space is naturally suitable for embedding hierarchical structure (Sarkar, 2011; Nickel & Kiela, 2017b) and can be regarded as a continuous representation of trees (Chami et al., 2020). The hyperbolic distance between samples implicitly reflects their hierarchical relation. Thus, by embedding images in hyperbolic space we can naturally organize images based on their semantic similarity and prototypicality.

4.2 Sphere Packing in Hyperbolic Space

Given n particles, our goal is to pack the particles into a two-dimensional hyperbolic space as densely as possible. We derive a simple repulsion loss function to encourage the particles to be equally distant from each other. The loss is derived via the following steps. First, we need to determine the radius of the Poincaré ball used for packing. We use a curvature of 1.0 so the radius of the Poincaré ball is 1.0. The whole Poincaré ball cannot be used for packing since the volume is infinite. We use $r < 1$ to denote the actual radius used for packing. Thus, our goal is to pack n particles in a compact subspace of Poincaré ball. Then, the Euclidean radius r is further converted into hyperbolic radius $r_{\mathbb{B}}$. Let $s = \frac{1}{\sqrt{c}}$, where c is the curvature. The relation between r and $r_{\mathbb{B}}$ is $r_{\mathbb{B}} = s \log \frac{s+r}{s-r}$. Next, the total hyperbolic area $A_{\mathbb{B}}$ of a Poincaré ball of radius $r_{\mathbb{B}}$ can be computed as $A_{\mathbb{B}} = 4\pi s^2 \sinh^2(\frac{r_{\mathbb{B}}}{2s})$, where $\sinh$ is the hyperbolic sine function. Finally, the area per point A_n can be easily computed as $\frac{A_{\mathbb{B}}}{n}$, where n is the total number of particles. Given A_n, the radius per point can be computed as $r_n = 2s \sinh^{-1}(\sqrt{\frac{A_n}{4\pi s^2}})$. We use the following loss to generate uniform packing in hyperbolic space. Given two particles i and j, the repulsion loss V is defined as,

$$V(i,j;k,n,r) = \{\frac{1}{[2r_n - \max(0, 2r_n - d_{\mathbb{B}}(i,j))]^k} - \frac{1}{(2r_n)^k}\} \cdot C(k) \tag{3}$$

where $C(k) = \frac{(2r_n)^{k+1}}{k}$ and k is a hyperparameter. Intuitively, if the particle i and the particle j are within $2r_n$, the repulsion loss is positive. Minimizing the repulsion loss would push the particle i and j away. If the repulsion is zero, this indicates all the particles are equally distant (Figure 4 a). Figure 4 b) shows that the repulsion loss grows significantly when the two particles become close.

We also adopt the following boundary loss to prevent the particles from escaping the ball,

$$B(i;r) = \max(0, \text{norm}_i - r + \text{margin}) \tag{4}$$

where norm_i is the ℓ_2 norm of the representation of the particle i. Figure 3 b) shows an example of the generated particles that are uniformly packed in hyperbolic space.

4.3 Hyperbolic Instance Assignment

HACK learns the features by optimizing the assignments of the images to the particles (Figure 5). Once we generate a fixed set of uniformly packed particles in a two-dimensional hyperbolic space, our next goal is to assign each image to the corresponding particle. The assignment should be one-to-one, that is, each image should be assigned to one particle and each particle is allowed to be associated with only one image. We cast the instance assignment problem as a bipartite matching problem (Gibbons, 1985) and solve it Hungarian algorithm (Munkres, 1957).

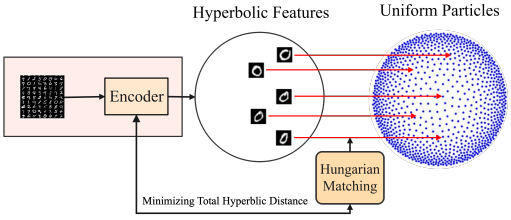


Figure 5: **HACK conducts unsupervised learning in hyperbolic space with sphere packing.** The images are mapped to particles by minimizing the total hyperbolic distance. HACK learns features that can capture both visual similarities and prototypicality.

Algorithm 1 HACK: Unsupervised Learning in Hyperbolic Space.

Require: # of images: $n \geq 0$. Radius for packing: $r < 1$. An encoder with parameters θ: f_θ

1: Generate uniformly distributed particles in hyperbolic space by minimizing the repulsion loss in Equation 3
2: Given $\{(\mathbf{x}_1, s_1), (\mathbf{x}_2, s_2), ..., (\mathbf{x}_b, s_b)\}$, optimize f_θ by minimizing the total hyperbolic distance via Hungarian algorithm.

Initially, we randomly assign the particles to the images, thus there is a random one-to-one correspondence between the images to the particles (not optimized). Given a batch of samples $\{(\mathbf{x}_1, s_1), (\mathbf{x}_2, s_2), ..., (\mathbf{x}_b, s_b)\}$, where $\mathbf{x}_i$ is an image and s_i is the corresponding particle, and an encoder f_θ, we generate the hyperbolic feature for each image $\mathbf{x}_i$ as $f_\theta(\mathbf{x}_i) \in \mathbb{B}^2$, where $\mathbb{B}^2$ is a two-dimensional Poincaré ball. We aim to find the minimum cost bipartite matching of the images to the particles within this batch. It is worth noting that no labels are needed and the assignment is done without supervision.

In the bipartite matching, the cost is the hyperbolic distance of each image to the particle. Thus, the criterion is to minimize the total hyperbolic distances of the assignment. We achieve this goal with Hungarian algorithm Munkres (1957) which has a complexity of $\mathcal{O}(b^3)$, where b is the batch size. It is worth noting that the assignment is only limited to the samples in the particular batch, thus the time and memory complexity is tolerable. The one-to-one correspondence between the images and particles are always maintained during training. The details of HACK is shown in Algorithm 1.

Due to the property of hyperbolic distance, the images that are more typical tend to be assigned to the particles located in the center of the Poincaré ball. Thus, HACK implicitly defines prototypicality as the distance of the sample to all the other samples. The prototypicality of the images can be easily reflected by the location of the assigned particles. Moreover, similar images tend to cluster together due to semantic similarity. In summary, with hyperbolic instance assignment, HACK automatically organizes images based on prototypicality by exploiting hyperbolicity of the space.

Why Does HACK Work? Hyperbolic space can embed tree structure with no distortion. In particular, the root of the tree can be embedded in the center of of the Poincaré ball and the leaves are embedded close to the boundary. Thus, the root is close to all the other nodes. This agrees with our intuition that typical examples should be close to all other examples. By minimizing the total assignment loss of the images to the particles, we seek to organize the images implicitly in a tree-structure manner. Consider three images A, B, C for an example. Assume image A is the most typical image. Thus the feature of A is close to both the features of B and C. The bipartite matching tends to assign image A to the particle in the center since this naturally reflects the feature distances between the three images.

Connection to Existing Methods. Existing works address the problem of prototypicality discovery with ad-hoc defined metrics (Carlini et al., 2018). These metrics usually have high-variances due to different training setups or hyperparameters. In this paper, we take a different perspective by exploiting the natural organization of the data by optimizing hyperbolic instance assignment. The property of hyperbolic space facilitates discovery of prototypicality. Also, popular contrastive learning based unsupervised learning methods such as SimCLR (Chen et al., 2020) and MoCo (He et al., 2020) cannot achieve this goal since the predefined structure is not specified.

5 Experiments

We design several experiments to show the effectiveness of HACK for semantic and prototypical organization. First, we first construct a dataset with known prototypicality using the congealing algorithm (Miller et al., 2000). Then, we apply HACK to datasets with unknown prototypicality to

organize the samples based on the semantic and prototypical structure. Finally, we show that the prototypical structure can be used to reduce sample complexity and increase model robustness.

5.1 DATASETS

We first construct a dataset called *Congealed MNIST*. To verify the efficacy of HACK for unsupervised prototypicality discovery, we need a benchmark with known prototypical examples. However, currently there is no standard benchmark for this purpose. To construct the benchmark, we use the congealing algorithm from Miller et al. (2000) to align the images in each class of MNIST (LeCun, 1998). The congealing algorithm is initially used for one-shot classification. During congealing, the images are brought into correspondence with each other jointly. The congealed images are more prototypical: they are better aligned with the average image. In Figure 2, we show the original images and the images after congealing. The original images are transformed via affine transformation to better align with each other. The synthetic data is generated by replacing 500 original images with the corresponding congealed images. In Section E of the Appendix, we show the results of changing the number of replaced original images. We expect HACK to discover the congealed images and place them in the center of the Poincaré ball. We also aim to discover the prototypical examples from each class of the standard MNIST dataset (LeCun, 1998) and CIFAR10 (Krizhevsky et al., 2009). CIFAR10 consists of 60000 from 10 object categories ranging from airplane to truck. CIFAR10 is more challenging than MNIST since it has larger intra-class variations.

5.2 BASELINES

We consider several existing metrics proposed in Carlini et al. (2018) for prototypicality discovery, the details can be found in Section C of the Appendix.

Holdout Retraining: We consider the Holdout Retraining proposed in Carlini et al. (2018). The idea is that the distance of features of prototypical example obtained from models trained on different datasets should be close.

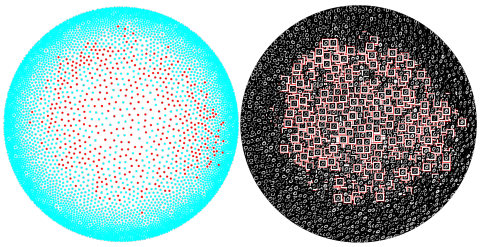
Model Confidence: Intuitively, the model should be confident on prototypical examples. Thus, it is natural to use the confidence of the model prediction as the criterion for prototypicality.

5.3 IMPLEMENTATION DETAILS

We implement HACK in Pytorch and the code will be made public. To generate the uniform particles, we first randomly initialize the particles. We run the training for 1000 epochs to minimize the repulsion loss and boundary loss. The learning rate is 0.01. The curvature of the Poincaré ball is 1.0 and the r is 0.76 which is used to alleviate the numerical issues (Guo et al., 2021b). The hyperparameter k is 1.55 which is shown to generate uniform particles well. For the assignment, we use a LeNet (LeCun et al., 1998) for MNIST and a ResNet20 (He et al., 2016) for CIFAR10 as the encoder. We apply HACK to each class separately. We attach a fully connected layer to project the feature into a two-dimensional Euclidean space. The image features are further projected onto hyperbolic space via an exponential map. We run the training for 200 epochs and the initial learning rate is 0.1. We use a cosine learning rate scheduler (Loshchilov & Hutter, 2016). We optimize the assignment *every other* epoch. All the experiments are run on a NVIDIA TITAN RTX GPU.

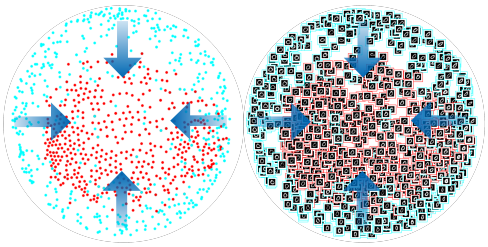
5.4 PROTOTYPICALITY DISCOVERY ON CONGEALED MNIST

Figure 6 shows that HACK can discover the congealed images from all the images. In Figure 6 a), the red particles denote the congealed images and cyan particles denote the original images. We can observe that the congealed images are assigned to the particles that locate in the center of the Poincaré ball. This verifies that HACK can *indeed* discover prototypical examples from the original dataset. Section G.1 in the Appendix shows that during training the features of atypical examples gradually move to the boundary of the Poincaré ball. In Figure 6 b), we show the actual images that are embedded in the two-dimensional hyperbolic space. We can observe that the images in the center of Poincaré ball are more prototypical and images close to the boundary are more atypical. Also, the images are naturally organized by their semantic similarity. Figure 7 shows that the features of the original images become closer to the center of Poincaré ball after congealing. In summary, HACK



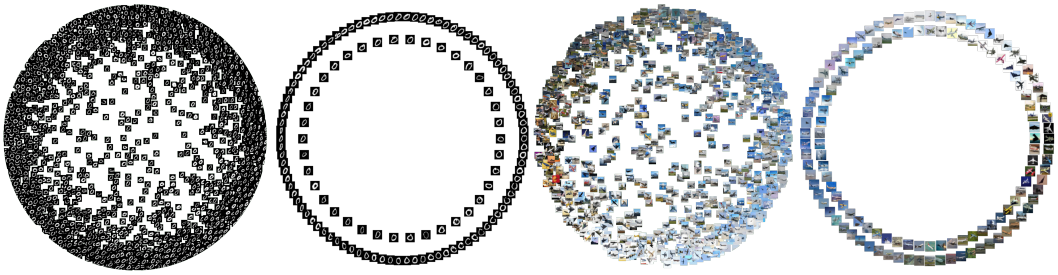
a) b)

Figure 6: **Congealed images are located in the center of the Poincaré ball.** a) Red dots denote congealed images and cyan dots denote original images. b) Typical images are in the center and atypical images are close to the boundary. Images are also clustered together based on visual similarity. Congealed images are shown in red boxs.



a) b)

Figure 7: **Original images are pushed to the center of the ball after congealing.** We train the first model with original images. Then we train the second model by replacing a subset of original images (marked with cyan) with the corresponding congealed images. The features of the congealed images (marked with red) become closer to the center of the ball.



a) b) c) d)

Figure 8: **Our unsupervised learning methods conforms to our visual perception HACK TODO** a) Samples of 2000 images from MNIST. b) Images of MNIST arranged angularly. c) Samples of 2000 images from CIFAR10. d) Images of CIFAR10 arranged angularly. Images are organized based on prototypicality and visual similarity.

can discover prototypicality and also organizes the images based on their semantics. To the best of our knowledge, this is the first unsupervised learning method that can be used to discover prototypical examples in a data-driven fashion.

5.5 Results on Standard Benchmarks

Figure 8 shows the embedding of class 0 from MNIST and class "airplane" from CIFAR10 in the hyperbolic space. We sample 2000 images from MNIST and CIFAR10 for better visualization. We also show the arrangement of the images angularly with different angles. Radially, we can observe that images are arranged based on prototypicality. The prototypical images tend to locate in the center of the Poincaré ball. Especially for CIFAR10, the images become blurry and even unrecognizable as we move towards the boundary of the ball. Angularly, the images are arranged based on visual similarity. The visual similarity of images has a smooth transition as we move around angularly. Please see Section D for more results.

Comparison with Baselines Figure 11 shows the comparison of the baselines with HACK. We can observe that both HACK and Model Confidence (MC) can discover typical and atypical images. Compared with MC, HACK defines prototypicality as the distance of the sample to other samples which is more aligned with human intuition. Moreover, in addition to prototypicality, HACK can also be used to organize examples by semantic similarities. Holdout Retraining (HR) is not effective for prototypicality discovery due to the randomness of model training.

Figure 11: **HACK can discover both typical and atypical examples.** First row: typical images discovered by different methods. Second row: atypical images discovered by different methods.

5.6 Application of Prototypicality

Reducing Sample Complexity. The proposed HACK can discover prototypical images as well as atypical images. We show that with *atypical* images we can reduce the sample complexity for training the model. Prototypical images are representative of the dataset but lack variations. Atypical examples contain more variations and it is intuitive that models trained on atypical examples should generalize better to the test samples. To verify this hypothesis, we select a subset of samples based on the norm of the features which indicates prototypicality of the examples. We consider using both the most typical and atypical examples for training the model. We train a LeNet on MNIST for 10 epochs with a learning rate of 0.1. Figure 9 a) shows that training with atypical images can achieve much higher accuracy than training with typical images. In particular, training with the most atypical 10% of the images achieves 16.54% higher accuracy than with the most typical 10% of the images. Thus, HACK provides an easy solution to reduce sample complexity. The results further verify that HACK can distinguish between prototypical and atypical examples.

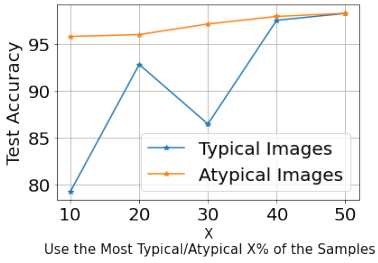


Figure 9: **Training with atypical examples achieves higher accuracy than training with typical examples.**

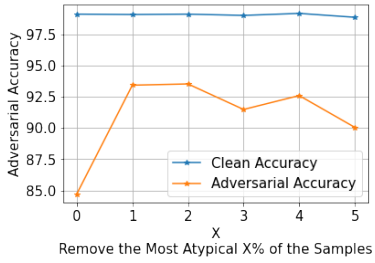


Figure 10: **The adversarial accuracy greatly improves after removing the X% of most atypical examples.**

Increasing Model Robustness. Training models with atypical examples can lead to vulnerable model to adversarial attacks (Liu et al., 2018; Carlini et al., 2018). Intuitively, atypical examples lead to less smooth decision boundary and a small perturbation to the example is likely to change the prediction. With HACK, we can easily identify atypical samples to improve the robustness of the model. We use MNIST as the benchmark and use FGSM (Goodfellow et al., 2014) to attack the model with an $\epsilon = 0.07$. We identify the atypical examples with HACK and remove the most atypical X% of the examples. Figure 9 b) shows that discarding atypically examples greatly improve the robustness of the model: the adversarial accuracy is improved from 84.72% to 93.42% by discarding the most atypical 1% of the examples. It is worth noting that the clean accuracy remains the same after removing a small number of atypical examples.

6 Summary

We propose an unsupervised learning method, called HACK, for organizing images with sphere packing in hyperbolic space. HACK optimizes the assignments of the images to a fixed set of uniformly distributed particles. Prototypical and semantic structures emerge naturally due to the property of hyperbolic distance. We apply HACK to synthetic data with known prototypicality and standard image datasets. The discovered prototypicality and atypical examples can be used to reduce sample complexity and increase model robustness.

REFERENCES

James W Anderson. *Hyperbolic geometry*. Springer Science & Business Media, 2006.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pp. 41–48, 2009.

Jacob Bien and Robert Tibshirani. Prototype selection for interpretable classification. *The Annals of Applied Statistics*, 5(4):2403–2424, 2011.

Károly Böröczky. Packing of spheres in spaces of constant curvature. *Acta Mathematica Hungarica*, 32(3-4):243–261, 1978.

Nicholas Carlini, Ulfar Erlingsson, and Nicolas Papernot. Prototypical examples in deep learning: Metrics, characteristics, and utility. 2018.

Ines Chami, Albert Gu, Vaggos Chatziafratis, and Christopher Ré. From trees to continuous embeddings and back: Hyperbolic hierarchical clustering. *Advances in Neural Information Processing Systems*, 33:15065–15076, 2020.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Henry Cohn. A conceptual breakthrough in sphere packing. *arXiv preprint arXiv:1611.01685*, 2016.

John Horton Conway and Neil James Alexander Sloane. *Sphere packings, lattices and groups*, volume 290. Springer Science & Business Media, 2013.

Evelyn Fix and Joseph Lawson Hodges. Discriminatory analysis. nonparametric discrimination: Consistency properties. *International Statistical Review/Revue Internationale de Statistique*, 57(3): 238–247, 1989.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. *arXiv preprint arXiv:1805.09112*, 2018.

Alan Gibbons. *Algorithmic graph theory*. Cambridge university press, 1985.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Yunhui Guo, Haoran Guo, and Stella Yu. Co-sne: Dimensionality reduction and visualization for hyperbolic data. *arXiv preprint arXiv:2111.15037*, 2021a.

Yunhui Guo, Xudong Wang, Yubei Chen, and Stella X Yu. Free hyperbolic neural networks with limited radii. *arXiv preprint arXiv:2107.11472*, 2021b.

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*, pp. 770–778, 2016.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738, 2020.

Joy Hsu, Jeffrey Gu, Gong-Her Wu, Wah Chiu, and Serena Yeung. Learning hyperbolic representations for unsupervised 3d segmentation. *arXiv preprint arXiv:2012.01644*, 2020.

Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. *Advances in neural information processing systems*, 29, 2016.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Yongshuai Liu, Jiyu Chen, and Hao Chen. Less is more: Culling the training set to improve robustness of deep neural networks. In *International Conference on Decision and Game Theory for Security*, pp. 102–114. Springer, 2018.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Emile Mathieu, Charline Le Lan, Chris J Maddison, Ryota Tomioka, and Yee Whye Teh. Continuous hierarchical representations with poincar\'e variational auto-encoders. *arXiv preprint arXiv:1901.06033*, 2019.

Erik G Miller, Nicholas E Matsakis, and Paul A Viola. Learning from one example through shared densities on transforms. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, volume 1, pp. 464–471. IEEE, 2000.

James Munkres. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38, 1957.

Yoshihiro Nagano, Shoichiro Yamaguchi, Yasuhiro Fujita, and Masanori Koyama. A wrapped normal distribution on hyperbolic space for gradient-based learning. In *International Conference on Machine Learning*, pp. 4693–4702. PMLR, 2019.

Maximilian Nickel and Douwe Kiela. Poincar\'e embeddings for learning hierarchical representations. *arXiv preprint arXiv:1705.08039*, 2017a.

Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. *Advances in neural information processing systems*, 30, 2017b.

Rik Sarkar. Low distortion delaunay embedding of trees in hyperbolic plane. In *International Symposium on Graph Drawing*, pp. 355–366. Springer, 2011.

Claude Elwood Shannon. A mathematical theory of communication. *ACM SIGMOBILE mobile computing and communications review*, 5(1):3–55, 2001.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

Pierre Stock and Moustapha Cisse. Convnets and imagenet beyond accuracy: Understanding mistakes and uncovering biases. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 498–512, 2018.

Richard Szeliski et al. Image alignment and stitching: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 2(1):1–104, 2007.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pp. 9929–9939. PMLR, 2020.

Zhenzhen Weng, Mehmet Giray Ogut, Shai Limonchik, and Serena Yeung. Unsupervised discovery of the long-tail in instance segmentation using hierarchical self-supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2603–2612, 2021.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3733–3742, 2018.

Jianping Zhang. Selecting typical instances in instance-based learning. In *Machine learning proceedings 1992*, pp. 470–479. Elsevier, 1992.

A APPENDIX

B MORE DETAILS ON HYPERBOLIC INSTANCE ASSIGNMENT

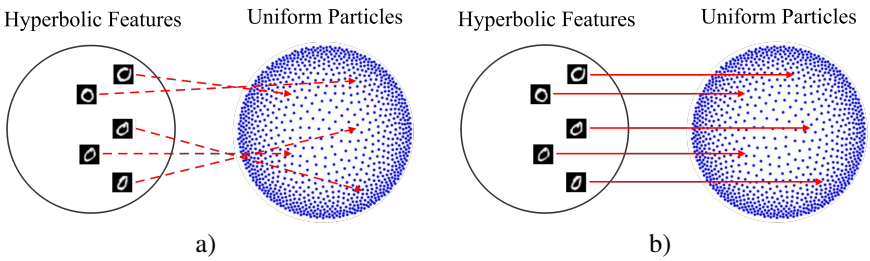


Figure 12: **Hyperbolic Instance Assignment minimizes the total hyperbolic distances between the image features and the particles.** a) Initial assignment. b) Optimized assignment.

A more detailed description of the hyperbolic instance assignment is given.

Initially, we randomly assign the particles to the images. Given a batch of samples $\{(\mathbf{x}_1, s_1), (\mathbf{x}_2, s_2), ..., (\mathbf{x}_b, s_b)\}$, where $\mathbf{x}_i$ is an image and s_i is the corresponding particle. Given an encoder f_θ, we generate the hyperbolic feature for each image $\mathbf{x}_i$ as $f_\theta(\mathbf{x}_i) \in \mathbb{B}^2$, where $\mathbb{B}^2$ is a two-dimensional Poincaré ball.

we aim to find the minimum cost bipartite matching of the images to the particles. The cost to minimize is the total hyperbolic distance of the hyperbolic features to the particles. We first compute all the pairwise distances between the hyperbolic features and the particles. This is the cost matrix of the bipartite graph. Then we use Hungarian algorithm to optimize the assignment (Figure 12).

Suppose we train the encoder f_θ for T epochs. We run the hyperbolic instance assignment every other epoch to avoid instability during training. **We optimize the encoder f_θ to minimize the hyperbolic distance of the hyperbolic feature to the assigned particle in each batch**.

C DETAILS OF BASELINES

Holdout Retraining: We consider the Holdout Retraining proposed in Carlini et al. (2018). The idea is that the distance of features of prototypical example obtained from models trained on different datasets should be close. In Holdout Retraining, multiple models are trained on the same dataset. The distances of the features of the images obtained from different models are computed and ranked. The prototypical examples are those examples with closest feature distance.

Model Confidence: Intuitively, the model should be confident on prototypical examples. Thus, it is natural to use the confidence of the model prediction as the criterion for prototypicality. Once we train a model on the dataset, we use the confidence of the model to rank the examples. The prototypical examples are those examples that the model is most

D MORE RESULTS ON PROTOTYPICALITY DISCOVERY

We show the visualization of all the images in Figure 17 and Figure 18. The images are organized naturally based their prototypicality and semantic similarity. We further conduct retrieval based on the norm of the hyperbolic features to extract the most typical and atypical images on CIAFR10 in Figure 19. The hyperbolic features with large norms correspond to atypical images and the hyperbolic features with small norms correspond to typical images. It can be observed that the object in the atypical images are not visible.

E GRADUALLY ADDING MORE CONGEALED IMAGES

We gradually increase the number of original images replaced by congealed images from 100 to 500. Still, as shown in Figure 13, HACK can learn representation that capture the concept of prototypicality regardless of the number of congealed images. This again confirms that the effectiveness of HACK for discovering prototypicality.

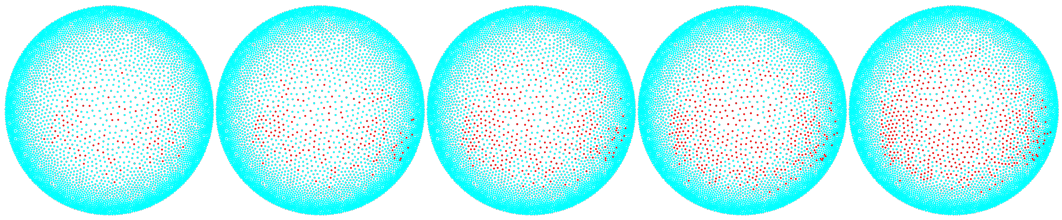


Figure 13: **HACK consistently places congealed images in the center of the Poincaré ball**. We gradually increase the number of original images replaced by congealed images from 100 to 500. The congealed images are marked with red dots and the original images are marked with cyan dots.

F DIFFERENT RANDOM SEEDS

We further run the assignment for 5 times with different random seeds. The results are shown in Figure 14. We observe that the algorithm does not suffer from high variance and the congealed images are always assigned to the particles in the center of the Poincaré ball. This further confirms the efficacy of the proposed method for discovering prototypicality.

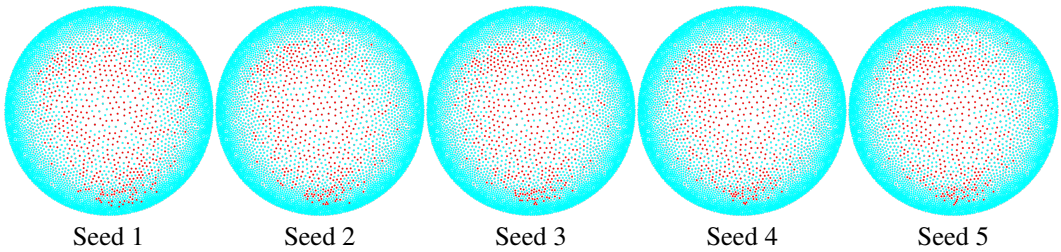


Figure 14: **HACK consistently places congealed images in the center of the Poincaré ball in multiple runs with different random seeds.**. The congealed images are marked with red dots and the original images are marked with cyan dots.

G EMERGENCE OF PROTOTYPICALITY IN THE FEATURE SPACE

Existing unsupervised learning methods mainly focus on learning features for differentiating different classes or samples Wu et al. (2018); He et al. (2020); Chen et al. (2020). The learned representations are transferred to various downstream tasks such as segmentation and detection. In contrast, the features learned by HACK aim at capturing prototypicality within a single class.

To investigate the effectiveness of HACK for revealing prototypicality, we can include or exclude congealed images in the training process. When the congealed images are included in the training process, we expect the congealed images to be located in the center of the Poincaré ball while the original images to be located near the boundary of the Poincaré ball. When the congealed images are excluded from the training process, we expect the features of congealed images produced via the trained network are located in the center of the Poincaré ball.

G.1 TRAINING WITH CONGEALED IMAGES AND ORIGINAL IMAGES

We follow the same setups as in the Section 4.3.1 of the main text. Figure 15 shows the hyperbolic features of the congealed images and original images in different training epochs. The features of the congealed images stay in the center of the Poincaré ball while the features of the original images gradually expand to the boundary.

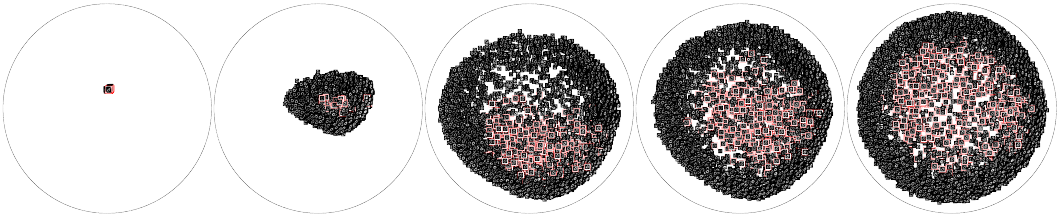


Figure 15: **Atypical images gradually move to the boundary of the Poincaré ball**. This shows that the representations learned by HACK captures prototypicality. Congealed images are in red boxes which are more typical. The network is trained with *both* the congealed images and original images.

G.2 TRAINING ONLY WITH ORIGINAL IMAGES

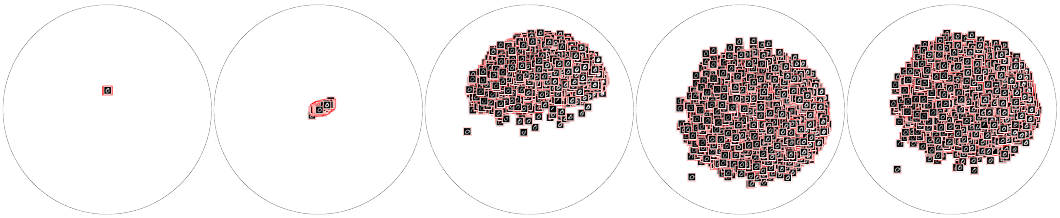


Figure 16: **The representations learned by HACK gradually capture prototypicality during the training process.** Congealed images are in red boxes which are more typical. We produce the features of the congealed images with the trained network in different epochs. The network is *only* trained with original images.

Figure 16 shows the hyperbolic features of the congealed images **when the model is trained only with original images**. As we have shown before, congealed images are naturally more typical than their corresponding original images since they are aligned with the average image. The features of congealed images are all located close to the center of the Poincaré ball. This demonstrate that prototypicality naturally emerge in the feature space.

Without using congealed images during training, we exclude any artifacts and further confirm the effectiveness of HACK for discovering prototypicality. We also observe that the features produced by HACK also capture the fine-grained similarities among the congealing images despite the fact that all the images are aligned with the average image.

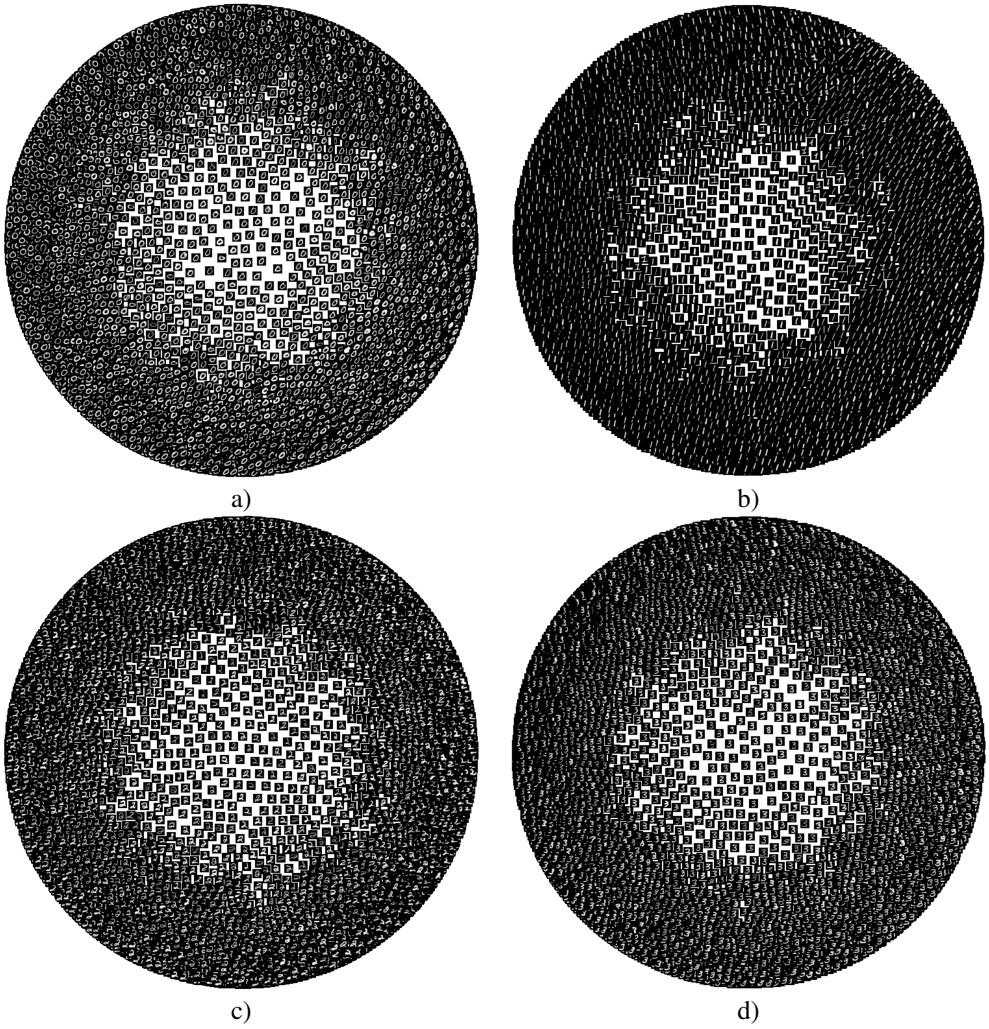


Figure 17: **HACK captures prototypicality and semantic similarity on MNIST.** a) Class 0. b) Class 1. c) Class 2. d) Class 3.

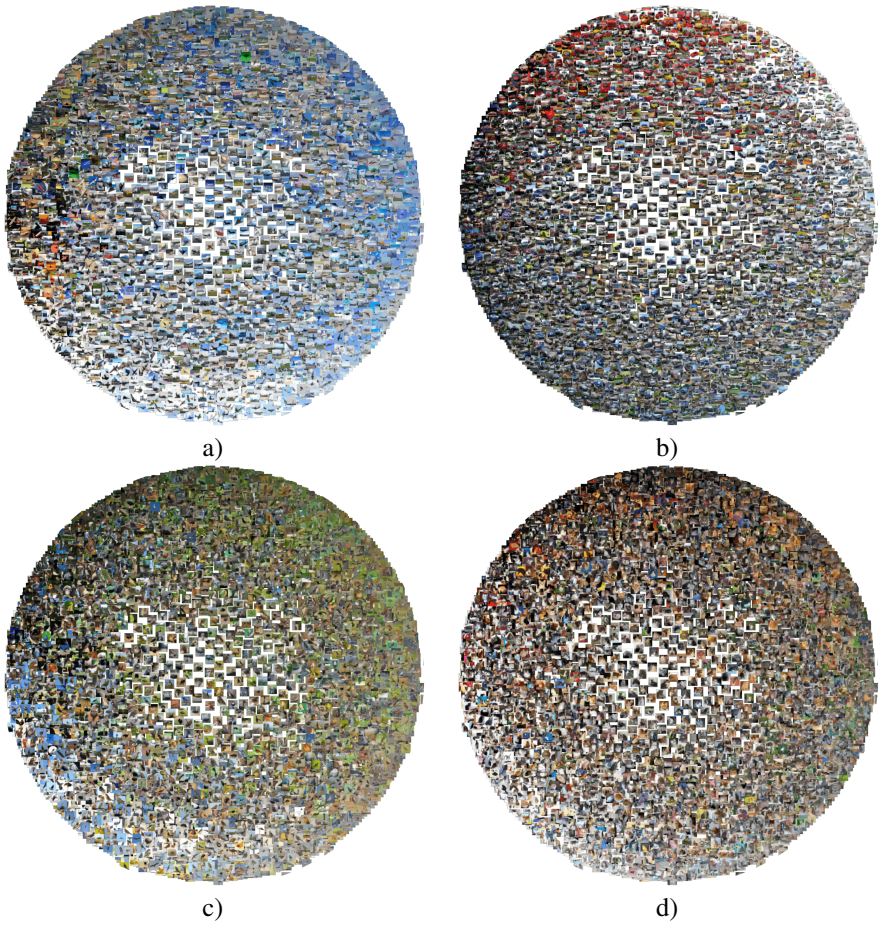


Figure 18: **HACK captures prototypicality and semantic similarity on CIFAR10.** a) Class "airplane". b) Class "automobile". c) Class "bird". d) Class "cat".

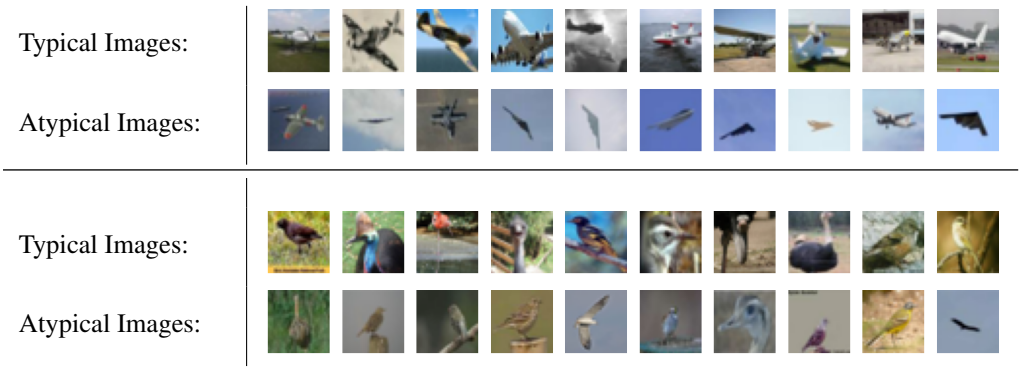


Figure 19: **Most typical and atypical images extracted by HACK from CIFAR10.**

H Discussions on Societal Impact and Limitations.

We address the problem of unsupervised learning in hyperbolic space. We believe the proposed HACK should not raise any ethical considerations. We discuss current limitations below,

Applying to the Whole Dataset Currently, HACK is applied to each class separately. Thus, it would be interesting to apply HACK to all the classes at once without supervision. This is much more

challenging since we need to differentiate between examples from different classes as well as the prototypical and semantic structure.

Exploring other Geometrical Structures We consider uniform packing in hyperbolic space to organize the images. It is also possible to extend HACK by specifying other geometrical structures to encourage the corresponding organization to emerge from the dataset.