Large-Margin Classification in Hyperbolic Space

Hyunghoon Cho[1] **Benjamin DeMeo**[2] **Jian Peng**[3] **Bonnie Berger**[1,*]
[1]Massachusetts Institute of Technology
[2]Harvard University
[3]University of Illinois at Urbana-Champaign
[*]Correspondence: bab@csail.mit.edu

Abstract

Representing data in hyperbolic space can effectively capture latent hierarchical relationships. To enable accurate classification of points in hyperbolic space while respecting their hyperbolic geometry, we introduce hyperbolic SVM, a hyperbolic formulation of support vector machine classifiers, and describe its theoretical connection to the Euclidean counterpart. We also generalize Euclidean kernel SVM to hyperbolic space, allowing nonlinear hyperbolic decision boundaries and providing a geometric interpretation for a certain class of indefinite kernels. Hyperbolic SVM improves classification accuracy in simulation and in real-world problems involving complex networks and word embeddings. Our work enables end-to-end analyses based on the inherent hyperbolic geometry of the data without resorting to ill-fitting tools developed for Euclidean space.

1 Introduction

Learning informative feature representations of symbolic data, such as text documents or graphs, is key to the success of downstream pattern recognition tasks. Recently, embedding data into hyperbolic space—a class of non-Euclidean spaces with constant negative curvature—has received increasing attention due to its effectiveness in capturing latent hierarchical structure (Alanis-Lobato et al., 2016; Muscoloni et al., 2017; Chamberlain et al., 2017; De Sa et al., 2018; Krioukov et al., 2010; Nickel and Kiela, 2017; Papadopoulos et al., 2015). This capability is likely because a key property of hyperbolic space is that the amount of space grows *exponentially* with the distance from a reference point, in contrast to the slower, polynomial growth in Euclidean space. The geometry of tree-structured data, which similarly expands exponentially with distance from the root, can thus be accurately captured in hyperbolic space, but not in Euclidean space (Krioukov et al., 2010).

A number of recent studies have therefore developed effective algorithms for embedding data in hyperbolic space, achieving superior performance on downstream tasks [e.g., answering semantic queries of words (De Sa et al., 2018; Nickel and Kiela, 2017) or link prediction in complex networks (Alanis-Lobato et al., 2016; Muscoloni et al., 2017; Chamberlain et al., 2017; Papadopoulos et al., 2015)] compared to their Euclidean counterparts, consistent with the intuition that better representing the inherent geometry of the data can improve downstream predictions.

However, aside from rudimentary analysis such as calculating the (hyperbolic) distances or angles between pairs of data points, solutions for standard pattern recognition tasks such as classification and clustering are limited to algorithms that are designed for data points in Euclidean spaces. For example, when Chamberlain et al. (2017) set out to classify nodes in a graph after embedding them into hyperbolic space, they resorted to performing logistic regression directly on the embedding coordinates, which relies on decision boundaries that are linear in the Euclidean sense, but are somewhat arbitrary when viewed in the underlying hyperbolic space.

To enable principled, end-to-end analyses that respect the inherent geometry of the data , we generalize linear support vector classifiers, one of the most widely-used classification methods, to data points in hyperbolic space. Despite the complexities of hyperbolic distance calculation, we prove that support vector classification in hyperbolic space can be performed by solving a simple, albeit nonconvex, optimization problem that

Proceedings of the 22[nd] International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89. Copyright 2019 by the author(s).

resembles the Euclidean formulation of SVM, elucidating the close connection between the two. To enable nonlinear classification in hyperbolic space, we derive the kernel version of hyperbolic SVM in a manner similar to the Euclidean case. We provide a technique for turning certain Euclidean kernels (e.g., the polynomial kernel) into a hyperbolic kernel and prove necessary and sufficient conditions for an arbitrary indefinite kernel to be understood as a natural (Minkowski) inner product kernel in hyperbolic space.

Hyperbolic SVM has superior experimental performance over the Euclidean version on two types of simulated datasets (Gaussian point clouds and evolving scale-free networks), real network datasets analyzed by Chamberlain et al. (2017), and semantic classification datasets based on hyperbolic word embeddings. A link to our software and benchmark datasets will be included in the final version of this paper.

The rest of the paper is organized as follows. We review hyperbolic geometry and support vector classification in Sections 2 and 3 and introduce our method, hyperbolic SVM, in Section 4. In Section 5, we extend our methods to hyperbolic kernel SVM, allowing nonlinear decision boundaries. We provide experimental evaluations of hyperbolic SVM in Section 6, and conclude with discussion and future work in Section 7. Our implementation of hyperbolic SVM can be found at `https://github.com/hhcho/hyplinear`.

2 The Hyperboloid Model

While hyperbolic space cannot be isometrically embedded in Euclidean space, there are several useful models of hyperbolic space formulated as a subset of Euclidean space. Our work primarily uses the hyperboloid model, described here. Three other models—Poincaré ball, Klein ball, and Poincaré half-space—are treated fully in Appendix. Figure 1 shows each of these models and corresponding geodesics.

Equip $\mathbb{R}^{n+1}$, with an inner product of the form

$$\mathbf{x} * \mathbf{y} = x_0 y_0 - x_1 y_1 - \cdots - x_n y_n.$$

This is commonly known as Minkowski space. The n-dimensional *hyperboloid model* $\mathbb{L}^n$ sits inside $\mathbb{R}^{n+1}$ as the forward sheet of a hyperboloid:

$$\mathbb{L}^n = \{\mathbf{x} = (x_0, \ldots, x_n) \in \mathbb{R}^{n+1} : x * x = 1, x_0 > 0\}.$$

The distance between two points in $\mathbb{L}^n$ is defined as the length of the geodesic path on the hyperboloid that connects the two points. These geodesic paths are exactly the intersections of $\mathbb{L}^n$ with 2-D planes containing the origin in the ambient Euclidean space $\mathbb{R}^{n+1}$ (Figure 1a).

3 Support Vector Classification Review

Let $\{(\mathbf{x}^{(j)}, y^{(j)})\}_{j=1}^m$ be a set of m training data instances, where the feature vector $\mathbf{x}^{(j)}$ is a point in a metric space $\mathcal{X}$ with distance function d, and $y^{(j)} \in \{1, -1\}$ denotes the true label for all j. Let $h : \mathcal{X} \mapsto \{1, -1\}$ be any decision rule. The *geometric margin* of h with respect to a single data instance $(\mathbf{x}, y)$ is:

$$\gamma_h(\mathbf{x}, y) = yh(\mathbf{x}) \cdot \inf\{d(\mathbf{x}', \mathbf{x}) : \mathbf{x}' \in \mathcal{X}, h(\mathbf{x}') \neq h(\mathbf{x})\}.$$

Increasing the value of γ_h across the training data points is desirable; for correct classifications, we increase our confidence, and for incorrect classifications, we minimize the error.

Maximum margin learning of the optimal decision rule $h^\star$, which provides the foundation for support vector machines, can now be formalized as

$$h^\star = \arg\max_{h \in \mathcal{H}} \min_{j \in [m]} \gamma_h(\mathbf{x}^{(j)}, y^{(j)}), \tag{1}$$

where $\mathcal{H}$ is the set of candidate decision rules.

If we let the data space $\mathcal{X}$ be $\mathbb{R}^n$ and d be the Euclidean distance function and consider only linear classifiers, then it can be shown that the maximum-margin problem given in Eq. 1 is equivalent to the following convex optimization problem:

$$\begin{aligned} &\text{minimize}_{\mathbf{w} \in \mathbb{R}^n} \ \frac{1}{2}\|\mathbf{w}\|^2 \\ &\text{subject to } y^{(j)}(\mathbf{w}^T \mathbf{x}^{(j)}) \geq 1, \forall j \in [m] \end{aligned} \tag{2}$$

The algorithm that solves this problem (via its dual) is known as a support vector machine (SVM). Introducing a relaxation for the separability constraints gives a more commonly used soft-margin variant of SVM:

$$\text{minimize}_{\mathbf{w} \in \mathbb{R}^n} \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{j=1}^m \ell(y^{(j)}(\mathbf{w}^T \mathbf{x}^{(j)})) \tag{3}$$

where $\ell(z) = \max(0, 1 - z)$, and the parameter $C > 0$ determines the trade-off between minimizing misclassification and maximizing the margin. Solving this optimization problem either in its primal form or via its dual has been established as a standard tool for classification in a wide range of domains (Fan et al., 2008).

4 Hyperbolic Support Vector Machines

We newly tackle the problem of solving the max-margin problem in Eq. 1 when the data points lie in

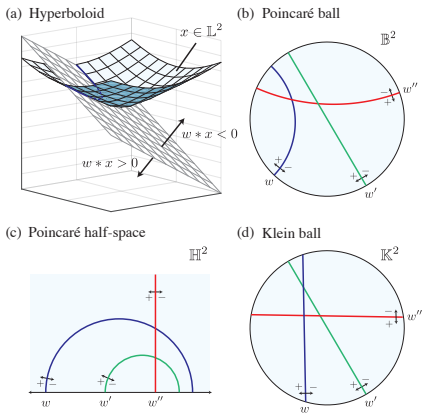

Figure 1: **Linear decision hyperplanes in hyperbolic space models.** Examples where $\mathbf{w}$, $\mathbf{w}'$, and $\mathbf{w}''$ denote different vectors in $\mathbb{R}^3$ that correspond to different decision hyperplanes in hyperbolic space. The correspondence of hyperplanes between different models is meant as an illustration and not drawn to scale.

hyperbolic space. In particular, we will adopt the hyperboloid model to let $\mathcal{X} = \mathbb{L}^n$ and let d be the hyperbolic distance function. The data points need not be initially specified using the hyperboloid model, since coordinates in other models of hyperbolic space can be easily converted to $\mathbb{L}^n$.

Analogous to the Euclidean SVM, we consider a set of decision functions that lead to geodesic decision boundaries *in the hyperbolic space*. It is known that any hyperbolic line (geodesic) in $\mathbb{L}^n$ is an intersection between $\mathbb{L}^n$ and a 2D Euclidean plane in the ambient space $\mathbb{R}^{n+1}$. Thus, a natural way to define decision hyperplanes in $\mathbb{L}^n$ is to use n-dimensional hyperplanes in $\mathbb{R}^{n+1}$ as a proxy. More precisely, we let

$$\mathcal{H} = \{h(\mathbf{x}; \mathbf{w}) : \mathbf{w} \in \mathbb{R}^{n+1}, \mathbf{w} * \mathbf{w} < 0\} \tag{4}$$

where

$$h(x; w) = \begin{cases} 1 & \mathbf{w} * \mathbf{x} > 0, \\ -1 & \text{otherwise}, \end{cases}$$

and $*$ denotes the Minkowski inner product. The corresponding decision boundaries are the n-dimensional hyperplanes in $\mathbb{R}^{n+1}$ given by $\mathbf{w} * \mathbf{x} = 0$.

The condition that $\mathbf{w}$ has negative Minkowski norm squared ($\mathbf{w} * \mathbf{w} < 0$) is needed to ensure we obtain a non-trivial decision function; otherwise, the decision hyperplane does not intersect with $\mathbb{L}^n$ in $\mathbb{R}^{n+1}$ and thus all points in $\mathbb{L}^n$ are classified as the same label.

The following lemma gives a simple closed-form expression for the geometric margin of a given data point to a decision hyperplane in hyperbolic space:

Lemma 4.1. *Given $\mathbf{w} \in \mathbb{R}^{n+1}$ such that $\mathbf{w} * \mathbf{w} < 0$ and a data point $\mathbf{x} \in \mathbb{L}^n$, the minimum hyperbolic distance from $\mathbf{x}$ to the decision boundary associated with $\mathbf{w}$, i.e., $\{\mathbf{z} : \mathbf{w} * \mathbf{z} = 0, \mathbf{z} \in \mathbb{L}^n\}$, is given by*

$$\sinh^{-1}\left(\frac{\mathbf{w} * \mathbf{x}}{\sqrt{-\mathbf{w} * \mathbf{w}}}\right).$$

A proof of Lemma 4.1 is provided in the Appendix.

Given this formula, one can apply a sequence of transformations to the max-margin classification problem in Eq. 1 for the hyperbolic setting to obtain the following result.

Theorem 4.1. *The maximum margin classification problem (Eq. 1), with hyperbolic feature space $\mathcal{X} = L^n$, hyperbolic distance function d, and hyperbolic-linear decision functions $\mathcal{H}$ as defined in Eq. 4, is equivalent to the following optimization problem:*

$$\begin{aligned} &\text{minimize}_{\mathbf{w} \in \mathbb{R}^{n+1}} \quad -\frac{1}{2}\mathbf{w} * \mathbf{w}, \\ &\text{subject to } y^{(j)}(\mathbf{w} * \mathbf{x}^{(j)}) \geq 1, \forall j \in [m], \\ &\mathbf{w} * \mathbf{w} < 0. \end{aligned}$$

The proof of Theorem 4.1 is analogous to the Euclidean version, and is provided in the Appendix.

Despite the apparent complexity of hyperbolic distance calculation, the optimal (linear) maximum margin classifiers in hyperbolic space can be identified via a relatively simple optimization problem that closely resembles the Euclidean version of SVM, where Euclidean inner products are replaced with Minkowski inner products. Unlike Euclidean SVM, however, our optimization problem has a non-convex objective as well as a non-convex constraint. Yet, for non-trivial, finite-sized problems where both classes are present in the data, it is necessary and sufficient to consider only the set of w for which at least one data point lies on either side of the decision boundary, suggesting that the optimal solution lies within a tighter convex region that maps out the convex hull of given data points.

Note that if we restrict $\mathcal{H}$ to decision functions where $w_0 = 0$, then our formulation coincides with with Euclidean SVM. Thus, Euclidean SVM can be viewed as a special case of our formulation where the first coordinate (corresponding to the time axis in Minkowski spacetime) is neglected.

Finally, the soft-margin formulation of hyperbolic SVM can be derived by relaxing the separability constraints as in the Euclidean case. We impose a penalty proportional to the *hyperbolic* distance to the correct classification. Analogous to the Euclidean formulation, we fix the scale of penalty so that the closest

point to decision boundary, $\mathbf{x}_m$, satisfies $\mathbf{w} * \mathbf{x}_m = 1$. Invoking Lemma 4.1, we see that the hyperbolic margin of $\mathbf{x}_m$ is $\sinh^{-1}(1)$. Points closer to the decision boundary are penalized proportional to their hyperbolic distance from the margin. This leads to the optimization problem

$$\begin{aligned}\text{minimize}_{\mathbf{w}\in\mathbb{R}^{n+1}} \ & -\frac{1}{2}\mathbf{w} * \mathbf{w} + C\sum_{j=1}^{m}\ell(y^{(j)}(\mathbf{w} * \mathbf{x}^{(j)})) \\ \text{subject to } & \mathbf{w} * \mathbf{w} < 0, \end{aligned} \tag{5}$$

where $\ell(z) = \max(0, \sinh^{-1}(1) - \sinh^{-1}(z))$.

Because the above formulation is non-convex, we use projected gradient descent to provably find a local optimum (Calamai and Moré, 1987). The initial $\mathbf{w}$ is determined based on the solution $\mathbf{w}'$ of a soft-margin SVM in the ambient Euclidean space of the hyperboloid model, so that $\mathbf{w} * \mathbf{x} = (\mathbf{w}')^T\mathbf{x}$ for all $\mathbf{x}$. This provides a good initialization for the optimization and improves the stability of the algorithm in the presence of potentially many local optima.

5 Nonlinear Classification in Hyperbolic Space

To enable classification with nonlinear decision boundaries, we construct feature mappings $\psi : \mathbb{L}^n \to \mathbb{L}^{\tilde{n}}$ that map the data points to another (typically higher-dimensional) hyperbolic feature space. Linear decision functions in $\mathbb{L}^{\tilde{n}}$ correspond to nonlinear decision functions in $\mathbb{L}^n$. This technique is well-established in the Euclidean setting and is commonly achieved without constructing the feature mapping ϕ, but instead working only with the kernel function $k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \phi(\mathbf{x}^{(i)})^T\phi(\mathbf{x}^{(j)})$. A Euclidean linear SVM on the transformed data points $\phi(\mathbf{x}^{(i)})$ is equivalent to solving

$$\begin{aligned}\text{minimize}_\alpha \ & \sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y^{(i)}y^{(j)}k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) + \sum_{i=1}^{m}\alpha_i \\ \text{subject to } & 0 \le \alpha_i \le C, \forall i \in [m], \end{aligned}$$

which is referred to as *kernel* SVM. Once a solution for α is obtained, the prediction for a new data point $\mathbf{z}$ is the sign of $\sum_{i=1}^{m}\alpha_i k(\mathbf{x}^{(i)}, \mathbf{z})$. In the following, we adapt this framework to hyperbolic space.

5.1 Hyperbolic Kernel SVM

Analogous to the derivation of kernel SVM in the Euclidean setting, analyzing the first order conditions of the Lagrangian of the max margin problem in Theorem 4.1 reveals that any $\mathbf{w}$ that represents a stationary solution can be written in the form $\mathbf{w} = -\sum_{i=1}^{m}\alpha_i\phi(\mathbf{x}^{(i)})$ for $0 \le \alpha_i \le C$. Re-parameterizing, we obtain the following formulation of hyperbolic kernel SVM (full derivation in Appendix):

$$\begin{aligned}\text{minimize}_\alpha \ & \sum_i\sum_j\alpha_i\alpha_j y^{(i)}y^{(j)}k_H(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) + \sum_i\alpha_i \\ \text{subject to } & 0 \le \alpha_i \le C, \forall i, \\ & \sum_i\sum_j\alpha_i\alpha_j y^{(i)}y^{(j)}k_H(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) > 0. \end{aligned}$$

where k_H denotes the Minkowski inner-product kernel (Minkowski kernel for short) defined as $k_H(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = -\psi(\mathbf{x}^{(i)}) * \psi(\mathbf{x}^{(j)})$ with a corresponding feature map $\psi : L^n \mapsto L^{\tilde{n}}$.

Euclidean kernels are usually required to be *positive semidefinite* (PSD), i.e., all eigenvalues of $M = [k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})]$ are non-negative for any set of $\mathbf{x}^{(i)}$. However, as we discuss further in Section 5.3, the Minkowski kernel is always non-PSD, so hyperbolic kernel SVM is distinct from standard Euclidean SVM.

5.2 Bootstrap Construction of Nonlinear Hyperbolic Kernels

The following lemma allows us to take any Euclidean kernel satisfying a certain condition and build a valid hyperbolic counterpart:

Lemma 5.1. *Let k_E be a Euclidean inner product kernel that satisfies $k_E(\mathbf{x}, \mathbf{x}) < 1$ for all $\|\mathbf{x}\| \le 1$. Then*

$$k_H(\mathbf{x}, \mathbf{y}) = \frac{k_E(g(\mathbf{x}), g(\mathbf{y})) - 1}{\sqrt{(1 - k_E(g(\mathbf{x}), g(\mathbf{x})))(1 - k_E(g(\mathbf{y}), g(\mathbf{y})))}}$$

is a valid Minkowski inner product kernel, where g maps coordinates in the hyperboloid model to the Klein ball model.

Notably, Lemma 5.1 immediately gives a generalization of Euclidean polynomial kernels $k_E(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T\mathbf{y})^d$ for any degree d to hyperbolic space. The resulting hyperbolic polynomial kernel in fact naturally corresponds to decision functions that take the shape of polynomial curves or hypersurfaces in hyperbolic space, which we describe in more detail in Appendix, along with the proof of the lemma.

5.3 Hyperbolic Kernel Matrix Properties

In Euclidean kernel SVM, much of the power arises from the ability to construct and optimize over kernels between arbitrary objects (graphs, documents, etc.), as long as the kernels are PSD. In order to allow arbitrary kernels in hyperbolic space, we asked what properties must be satisfied by a kernel $k(\mathbf{x}, \mathbf{y})$ in order for it to be formulated as $-\phi(\mathbf{x}) * \phi(\mathbf{y})$ for some feature-space mapping ϕ. We have the following Theorem:

Theorem 5.1. *Let M be an $n \times n$ real symmetric matrix. Then M can be expressed as $-[\phi(\mathbf{x}_i) * \phi(\mathbf{x}_j)]$ for some mapping ϕ into the hyperboloid model if and only if the following hold:*

1. *The diagonal entries of M are all -1, and the remaining entries are ≤ -1.*
2. *M has exactly one negative eigenvalue.*

These criteria allow the researcher to identify when a certain non-PSD kernel SVM problem can be rephrased as a hyperbolic SVM problem. Note that if M obeys the conditions when -1 is replaced with some $-a < 0$, M can be scaled to fit the criteria without affecting the optimization problem it represents. A full proof of Theorem 5.1 is presented in the Appendix.

Interestingly, a natural extension of Gaussian radial basis function (RBF) kernel with hyperbolic distance function d_H, defined as $k(\mathbf{x}, \mathbf{y}) = \exp\{-\gamma d_H(\mathbf{x}, \mathbf{y})^2\}$ with a parameter $\gamma > 0$, does not satisfy the conditions of Theorem 5.1 in general. Although the Euclidean RBF kernel has an equivalent infinite-dimensional feature map, our observation shows that a similar derivation is not directly possible in the hyperbolic case. Whether there is a modified form of hyperbolic RBF that fits the description of Theorem 5.1 remains an open question. Note that in practice a polynomial kernel with a sufficiently high degree may be a viable alternative to RBF kernels (Cotter et al., 2011).

6 Experimental Results

Below, we compare hyperbolic SVM to the original Euclidean SVM (i.e., L2-regularized hinge-loss optimization) on a range of real and simulated datasets. After describing our evaluation setup (Section 6.1), we present the results for linear classification (Sections 6.2-6.4). Experiments for nonlinear classification in hyperbolic space are provided in Section 6.5.

6.1 Evaluation Setting

To enable multi-class classification, we adopt a one-vs-all (OVA) strategy, where several binary classifiers are independently trained to distinguish each class from the rest. For each method, the resulting prediction scores on the holdout data are transformed into probability outputs via Platt scaling (Platt et al., 1999) across all classes and collectively analyzed to quantify the overall classification accuracy. For hyperbolic SVM we use the Minkowski inner product between the learned weight vector and the data point in the hyperboloid model as the prediction score, which is a monotonic transformation of the geometric margin.

In both hyperbolic and Euclidean SVMs, the tradeoff between minimizing misclassification and maximizing margin is determined by the parameter C (see Eqs. 3 and 5). In all our experiments, we determined the optimal $C \in \{0.1, 1, 10\}$ separately for each run via a nested cross-validation procedure.

Our main performance metric is macro-averaged area under the precision recall curve (AUPR), which is obtained by computing the AUPR of predicting each class against the rest separately, then taking the average across all classes. The results based on other performance metrics, such as the area under the ROC curve and the micro-average variants of both metrics, led to similar conclusions across all our experiments.

6.2 Simulated Gaussian Mixture Datasets

We first generated a collection of 100 toy datasets by sampling data points from a Gaussian mixture model defined in the Poincaré disk model. Note that, analogous to the Euclidean setting, the probability density function of an (isotropic) hyperbolic Gaussian distribution decays exponentially with the squared hyperbolic distance from the centroid, inversely scaled by the variance parameter. For each dataset, we sampled four centroids from a zero-mean hyperbolic Gaussian distribution with variance parameter 1.5. Then, we sampled 100 data points from a unit-variance hyperbolic Gaussian distribution centered at each centroid to form a dataset of 400 points assigned to 4 classes.

The results of two-fold cross validation experiments on each of the 100 datasets are summarized in Figure 2a. We observed a strongly significant improvement of hyperbolic SVM over the Euclidean version in terms of prediction accuracy, with a one-sided paired-sample t-test p-value of 6.17×10^{-28}. Our method also outperformed Euclidean SVM based on the Klein and hyperboloid models of hyperbolic space (Appendix).

We attribute the performance improvement of hyperbolic SVM to the fact that its decision functions better match the geometry of the given data. Example decision boundaries for both methods are shown in Figures 2b and c. Note that the apparent nonlinearity of hyperbolic decision boundaries is due to our use of the Poincaré disk for visualization; in the hyperbolic space, these decision boundaries are in fact linear.

6.3 Semantic Classification of Word Embeddings

A key application of hyperbolic embeddings is learning representations of words that capture their semantic hierarchy (De Sa et al., 2018; Nickel and Kiela, 2017). We next evaluated hyperbolic SVM on a natural lan-

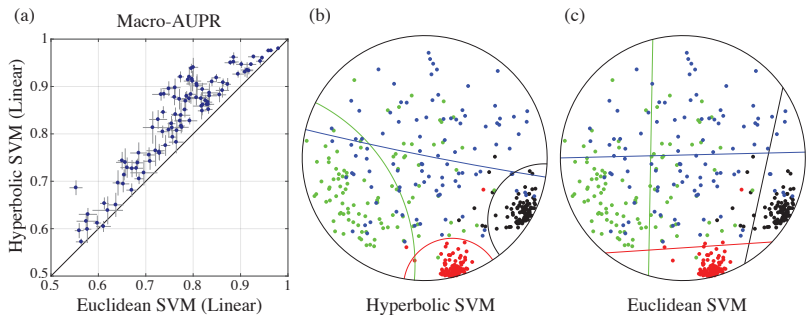

Figure 2: **Multi-class classification of Gaussian mixtures in hyperbolic space**. (a) Two-fold cross validation results for 100 hyperbolic Gaussian mixture datasets. Each dot represents the average performance over 5 trials. Vertical and horizontal lines represent standard deviations. Example decision hyperplanes for hyperbolic and Euclidean SVMs are shown in (b) and (c), respectively, using the Poincaré disk model. Color of each decision boundary denotes which component is being discriminated from the rest.

WordNet Subtree	Classifier (Linear)	
	Hyperbolic SVM	Euclidean SVM
tree.n.01	**0.29** $\pm$ 0.03	0.22 $\pm$ 0.01
worker.n.01	**0.31** $\pm$ 0.01	0.23 $\pm$ 0.01
group.n.01	**0.66** $\pm$ 0.01	0.59 $\pm$ 0.01
solid.n.01	**0.67** $\pm$ 0.13	0.53 $\pm$ 0.03
mammal.n.01	**0.70** $\pm$ 0.11	0.41 $\pm$ 0.07
animal.n.01	**0.89** $\pm$ 0.01	0.87 $\pm$ 0.01

Table 1: **Semantic classification performance on WordNet dataset.** For six different subtrees of the WordNet hierarchy, we performed five-fold cross validation for predicting words in the subtree based on the 2D hyperbolic embedding of Ganea et al. (2018a). Average AUPR summarized over 10 trials is shown, each followed by the standard deviation. Better performance for each dataset is shown in boldface.

guage processing task of classifying whether a given word belongs to a semantic category. Following the work of Ganea et al. (2018b), we embedded the semantic hierarchy of all English nouns in the WordNet dataset* into a 2D hyperbolic space using a recently proposed approach of Ganea et al. (2018a). We then performed cross validation experiments for predicting whether a word belongs to a chosen subtree based on the hyperbolic embeddings. We observed that hyperbolic SVM consistently outperforms Euclidean SVM applied directly on the Poincaré disk coordinates (Table 1). Our performance improvement was most significant for predicting words in the mammal subtree, where hyperbolic SVM increased the average AUPR by 0.29 over the Euclidean approach.

6.4 Node Classification in Complex Networks

Another key application of hyperbolic space embedding is modeling complex, scale-free networks (Alanis-Lobato et al., 2016; Muscoloni et al., 2017; Papadopoulos et al., 2015, 2012). We tested whether hyperbolic SVM can improve node classification performance on the hyperbolic embedding of such networks.

6.4.1 Real-World Static Networks

We evaluated hyperbolic SVM on four real-world network datasets used by Chamberlain et al. (2017). These include: (1) karate (Zachary, 1977): a social network of 34 people divided into two factions, (2) polbooks†: co-purchasing patterns of 105 political books in 2004 divided into 3 affiliations, (3) football (Girvan and Newman, 2002): football matches among 115 colleges in Fall 2000 divided into 12 leagues, and (4) polblogs (Lada and Natalie, 2005): a hyperlink network of 1224 political blogs in 2005 divided into two affiliations. We excluded the adjnoun dataset due to the near-random performance of all methods considered.

For each dataset, we embedded the network into a 2D hyperbolic space using the approach of Chamberlain et al. (2017) based on random walks. Their method closely follows an existing network embedding algorithm called DeepWalk (Perozzi et al., 2014) except Euclidean inner products are replaced with a measure of hyperbolic angle. Given the hyperbolic embedding of each network, we performed two-fold cross validation to compare the node classification accuracy of hyperbolic SVM with Euclidean SVM.

For all four datasets, hyperbolic SVM matched or outperformed the performance of Euclidean SVM (Table 2). Notably, the two datasets where the performance was comparable between the two methods (karate and polblogs) consisted of only two well-separated classes, in which case a Euclidean linear decision boundary is expected to perform well.

*https://wordnet.princeton.edu/

†http://www-personal.umich.edu/~mejn/netdata/

Classifier (Linear)	Embedding (Dimension)	Dataset			
		karate	polbooks	football	polblogs
Hyperbolic SVM	Hyperbolic (2)	**0.86** $\pm$ 0.03	**0.73** $\pm$ 0.04	**0.24** $\pm$ 0.03	**0.93** $\pm$ 0.01
Euclidean SVM	Hyperbolic (2)	**0.86** $\pm$ 0.03	0.66 $\pm$ 0.02	0.21 $\pm$ 0.01	**0.93** $\pm$ 0.01
Euclidean SVM	Euclidean (2)	0.47 $\pm$ 0.07	0.34 $\pm$ 0.03	0.09 $\pm$ 0.01	0.60 $\pm$ 0.09
Euclidean SVM	Euclidean (5)	0.55 $\pm$ 0.08	0.35 $\pm$ 0.03	0.10 $\pm$ 0.01	0.69 $\pm$ 0.04
Euclidean SVM	Euclidean (10)	0.50 $\pm$ 0.08	0.36 $\pm$ 0.03	0.10 $\pm$ 0.01	0.72 $\pm$ 0.04
Euclidean SVM	Euclidean (25)	0.50 $\pm$ 0.09	0.37 $\pm$ 0.04	0.11 $\pm$ 0.02	0.80 $\pm$ 0.03

Table 2: **Node classification performance on four real-world network datasets.** We performed two-fold cross validation experiments on four real-world network datasets with labeled nodes. Average macro-AUPR over a total of 20 cross-validation trials based on 5 different embeddings of each network is shown, each followed by the standard deviation. Numbers corresponding to the best performance on each dataset are shown in boldface.

In addition, we tested Euclidean SVM based on the *Euclidean* embeddings obtained by DeepWalk with dimensions 2, 5, 10, and 25. Even with as many as 25 dimensions, Euclidean SVM was not able to achieve competitive prediction accuracy based on the Euclidean embeddings across all datasets (Table 2). This supports the conclusion that hyperbolic geometry likely underlies these networks and that increasing the number of dimensions for the Euclidean embedding does not necessarily lead to representations that are as informative as the hyperbolic embedding.

6.4.2 Simulated Dynamic Networks

We next considered node classification tasks on *time-varying* networks, a commonly studied subject in the context of hyperbolic geometry. To this end, we generated random scale-free networks using the popularity-vs-similarity (PS) model (Figure 3a), which was shown to capture the properties of many real-world networks (Papadopoulos et al., 2012). We embedded each simulated network into hyperbolic space using LaBNE (Alanis-Lobato et al., 2016), a network embedding method based on the PS model (Figure 3b).

Inspired by the gene function prediction task in network biology (Cho et al., 2016), we then generated a multi-class, multi-label dataset for each simulated network. For each new label, we randomly choose a node in the network to be the first node to be annotated with the label. Then, we replay the evolution of the network, and each time a new node is connected to an existing node with a given label, the label propagates to the new node with a set probability (0.8 in our experiments), which simulates the stochastic inheritance of node properties in evolving networks.

Given a target range for the label size (number of nodes having the label), we created 10 labels to obtain a multi-label classification dataset with 10 classes. This process was repeated 5 times for 10 different simulated networks to generate a total of 150 datasets with varying label sizes (20–50, 50–100, and 100–200). Across all label size ranges and networks, hyperbolic SVM matched or outperformed Euclidean SVM (Figure 3c). The overall improvement of hyperbolic SVM was statistically significant, with a one-sided paired-sample t-test p-value of 3.99×10^{-21}.

6.5 Nonlinear Hyperbolic Classification

Here we demonstrate the performance of hyperbolic SVM with *nonlinear* decision boundaries in hyperbolic space. Given the relatively simple geometry of our datasets, we restrict our attention to the quadratic versions of hyperbolic versus Euclidean SVMs in our experiments. Implementation details of quadratic hyperbolic SVM is provided in the Appendix.

We tested both methods on more challenging Gaussian mixture datasets, where each component was an elliptical Gaussian distribution (with a random shape) rather than an isotropic distribution (Section 6.2). Quadratic hyperbolic SVM significantly outperforms quadratic Euclidean SVM on these datasets (Figure 4a) with a one-sided paired t-test p-value of 5.00×10^{-18}. Similar improvement was observed on the evolving network node classification datasets (Figure 4b) with a p-value of 9.09×10^{-23}. Our improvement was most pronounced for datasets of smaller labels (20–50 nodes), likely because larger labels more evenly partition the space and are less sensitive to the particular choice of decision boundaries. Quadratic hyperbolic SVM outperformed Euclidean SVM on the WordNet benchmark data (Table 3), in addition to improving on the linear classification results in Table 1.

7 Discussion and Future Work

We developed support vector classification in hyperbolic space, demonstrated its improved performance on a wide range of datasets, and developed the hyperbolic analog of kernel SVM, lending geometric intuition and algorithmic tractability to large-margin learning with a range of non-PSD kernels (Theorem 5.1). Alternative non-convex optimization methods,

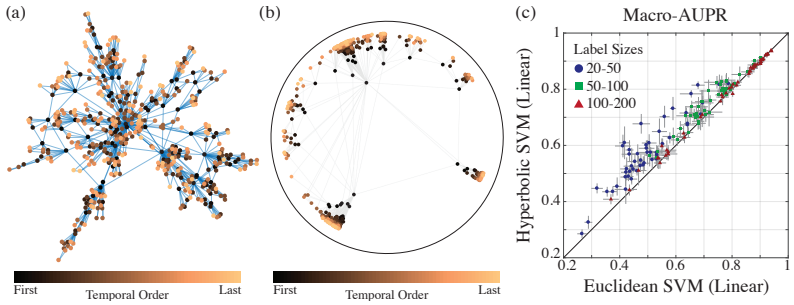

Figure 3: **Multi-class multi-label classification of nodes in simulated evolving networks.** (a) One of the simulated networks used to generate our benchmark datasets based on the PS model (Papadopoulos et al., 2012). We set the number of nodes to 500, average degree to 4, scaling exponent to 2.25, and temperature to 0, in order to achieve a modest level of clustering. (b) Embedding of the same network in two-dimensional hyperbolic space as visualized in the Poincaré disk model. (c) Two-fold cross validation results for predicting 10 labels per dataset, where each label is assigned to a random node and stochastically propagated to its descendants. We repeated the experiment for different size ranges for the labels, denoted by marker type/color. Each marker represents the mean performance over 5 trials. Vertical and horizontal lines represent standard deviations.

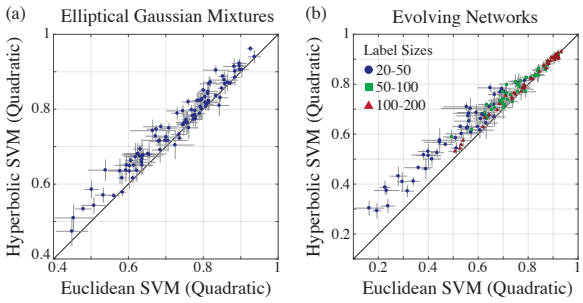

Figure 4: **Comparison of quadratic hyperbolic and Euclidean SVMs on our simulated datasets.** Panels (a) and (b) correspond to the experiments depicted in Figure 2a and Figure 3c, respectively, but with quadratic classifiers. In addition, for (a), we used a mixture of elliptical Gaussian distributions instead of isotrophic ones to better motivate the use of quadratic decision functions. On both benchmark datasets, quadratic hyperbolic SVM significantly outperforms the Euclidean counterpart.

WordNet Subtree	Classifier (Quadratic)	
	Hyperbolic SVM	Euclidean SVM
tree.n.01	**0.46** $\pm$ 0.07	0.28 $\pm$ 0.22
worker.n.01	**0.48** $\pm$ 0.12	0.16 $\pm$ 0.10
group.n.01	**0.65** $\pm$ 0.03	0.61 $\pm$ 0.01
solid.n.01	**0.72** $\pm$ 0.02	0.60 $\pm$ 0.09
mammal.n.01	**0.91** $\pm$ 0.04	**0.89** $\pm$ 0.05
animal.n.01	**0.90** $\pm$ 0.01	**0.90** $\pm$ 0.01

Table 3: **Semantic classification with quadratic hyperbolic and Euclidean SVMs.** We replicated the experiment in Table 1 for quadratic classifiers. Mean AUPR summarized over 10 trials is shown, each followed by the standard deviation. Numbers within a standard deviation from the best result are in boldface.

such as Krein methods (Loosli et al., 2016) or programming with differences of convex functions (Xu et al., 2017), may further improve the performance of hyperbolic SVM.

Our work belongs to a growing body of algorithms that learn directly over a Riemannian manifold (Porikli, 2010; Tuzel et al., 2008). Linear hyperplane-based classifiers and clustering algorithms have previously been formulated for spherical spaces (Dhillon and Modha, 2001; Lebanon and Lafferty, 2004; Wilson and Hancock, 2010). To the best of our knowledge, our work is the first to develop and experimentally demonstrate support vector classification in hyperbolic geometry. We envision further development of hyperbolic space-equivalents of other standard machine learning tools in the near future. For example, a concurrent work of Ganea et al. (2018b) introduces a hyperbolic formulation of neural networks, a potential alternative for hyperbolic space classification.

The kernel representation of hyperbolic SVM allows us to learn in hyperbolic space without constructing an embedding. Instead, we can look for kernels that satisfy the conditions enumerated in Theorem 5.1. Although many convenient kernels are not PSD, some of them may be equivalent to a Minkowski kernel after some modifications, permitting a convenient formulation in hyperbolic space. Thus, our work represents a first step towards a geometric understanding of indefinite kernel classification.

References

Alanis-Lobato, G., Mier, P., and Andrade-Navarro, M. A. (2016). Efficient embedding of complex networks to hyperbolic space via their Laplacian. *Scientific reports*, 6:30108.

Calamai, P. H. and Moré, J. J. (1987). Projected gradient methods for linearly constrained problems. *Mathematical Programming*, 39(1):93–116.

Chamberlain, B. P., Clough, J., and Deisenroth, M. P. (2017). Neural Embeddings of Graphs in Hyperbolic Space. *CoRR*, stat.ML.

Cho, H., Berger, B., and Peng, J. (2016). Compact Integration of Multi-Network Topology for Functional Analysis of Genes. *Cell Systems*, 3(6):540–548.e5.

Cotter, A., Keshet, J., and Srebro, N. (2011). Explicit approximations of the gaussian kernel. *CoRR*, abs/1109.4603.

De Sa, C., Gu, A., Ré, C., and Sala, F. (2018). Representation Tradeoffs for Hyperbolic Embeddings. *CoRR*, abs/1804.03329.

Dhillon, I. S. and Modha, D. S. (2001). Concept Decompositions for Large Sparse Text Data Using Clustering. *Machine Learning*, 42(1):143–175.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9(Aug):1871–1874.

Ganea, O., Bécigneul, G., and Hofmann, T. (2018a). Hyperbolic entailment cones for learning hierarchical embeddings. *CoRR*, abs/1804.01882.

Ganea, O., Bécigneul, G., and Hofmann, T. (2018b). Hyperbolic neural networks. *CoRR*, abs/1805.09112.

Girvan, M. and Newman, M. E. (2002). Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826.

Krioukov, D., Papadopoulos, F., Kitsak, M., Vahdat, A., and Bogu n 'a, M. a. (2010). Hyperbolic geometry of complex networks. *Phys. Rev. E*, 82:036106.

Lada, A. and Natalie, G. (2005). The political blogosphere and the 2004 us election. In *Proceedings of the 3rd international workshop on Link discovery*, volume 1, pages 36–43.

Lebanon, G. and Lafferty, J. (2004). Hyperplane Margin Classifiers on the Multinomial Manifold. In *Proceedings of the Twenty-first International Conference on Machine Learning*, pages 66–66, New York, NY, USA. ACM.

Loosli, G., Canu, S., and Ong, C. S. (2016). Learning svm in kren spaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(6):1204–1216.

Muscoloni, A., Thomas, J. M., Ciucci, S., Bianconi, G., and Cannistraci, C. V. (2017). Machine learning meets complex networks via coalescent embedding in the hyperbolic space. *Nature communications*, 8(1):1615.

Nickel, M. and Kiela, D. (2017). Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems*, pages 6341–6350.

Papadopoulos, F., Aldecoa, R., and Krioukov, D. (2015). Network geometry inference using common neighbors. *Physical Review E*, 92(2):022807.

Papadopoulos, F., Kitsak, M., Serrano, M. Á., Boguñá, M., and Krioukov, D. (2012). Popularity versus similarity in growing networks. *Nature*, 489(7417):537–540.

Perozzi, B., Al-Rfou, R., and Skiena, S. (2014). DeepWalk: Online Learning of Social Representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 701–710, New York, NY, USA. ACM.

Platt, J. et al. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74.

Porikli, F. (2010). Learning on Manifolds. In Hancock, E. R., Wilson, R. C., Windeatt, T., Ulusoy, I., and Escolano, F., editors, *Structural, Syntactic, and Statistical Pattern Recognition*, pages 20–39, Berlin, Heidelberg. Springer Berlin Heidelberg.

Tuzel, O., Porikli, F., and Meer, P. (2008). Pedestrian Detection via Classification on Riemannian Manifolds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(10):1713–1727.

Wilson, R. C. and Hancock, E. R. (2010). Spherical Embedding and Classification. In Hancock, E. R., Wilson, R. C., Windeatt, T., Ulusoy, I., and Escolano, F., editors, *Structural, Syntactic, and Statistical Pattern Recognition*, pages 589–599, Berlin, Heidelberg. Springer Berlin Heidelberg.

Xu, H.-M., Xue, H., Chen, X., and Wang, Y. (2017). Solving indefinite kernel support vector machine with difference of convex functions programming. In *AAAI*, pages 2782–2788.

Zachary, W. W. (1977). An information flow model for conflict and fission in small groups. *Journal of anthropological research*, 33(4):452–473.