Journal of Machine Learning Research 9 (2008) 105-130 Submitted 1/07; Revised 12/07; Published 1/08

Active Learning by Spherical Subdivision

Falk-Florian Henrich HENRICH@MATH.TU-BERLIN.DE
Technische Universität Berlin
Institut für Mathematik
Sekr. MA 8–3
Straße des 17. Juni 136
D-10623 Berlin, Germany

Klaus Obermayer OBY@CS.TU-BERLIN.DE
Technische Universität Berlin
Institut für Informatik
FR 2–1, NI
Franklinstraße 28/29
D-10587 Berlin, Germany

Editor: David Cohn

Abstract

We introduce a computationally feasible, "constructive" active learning method for binary classification. The learning algorithm is initially formulated for separable classification problems, for a hyperspherical data space with constant data density, and for great spheres as classifiers. In order to reduce computational complexity the version space is restricted to spherical simplices and learning procedes by subdividing the edges of maximal length. We show that this procedure optimally reduces a tight upper bound on the generalization error. The method is then extended to other separable classification problems using products of spheres as data spaces and isometries induced by charts of the sphere. An upper bound is provided for the probability of disagreement between classifiers (hence the generalization error) for non-constant data densities on the sphere. The emphasis of this work lies on providing mathematically exact performance estimates for active learning strategies.

Keywords: active learning, spherical subdivision, error bounds, simplex halving

1. Introduction

Active learning methods seek a solution to inductive learning problems by incorporating the selection of training data into the learning process. In these schemes, the labeling of a data point occurs only after the algorithm has explicitly asked for the corresponding label, and the goal of the "active" data selection is to reach the same accuracy as standard "passive" algorithms—but with less labeled data points. In many practical tasks, the acquisition of unlabeled data can be automated, while the actual labeling must often be done by humans and is therefore time consuming and costly. In these cases, active learning methods—which usually trade labeling costs against the computational burden required for optimal data selection—can be a valuable alternative.

There are two approaches to active learning. So-called query filtering methods (Freund et al., 1997; Opper et al., 1992) operate on a given pool of unlabeled data and select—at every learning step—a "most informative" data point for subsequent labeling. So-called constructive methods lit-

erally "construct" an unlabeled datum and ask the user to provide a label. There is strong empirical evidence for many learning scenarios and for different selection procedures, that active learning methods can reduce the number of labeled training data which are needed to reach a predefined generalization performance (see Balcan et al., 2006; Fine et al., 2002; Freund et al., 1997; Tong and Koller, 2001; Warmuth et al., 2002). In addition, theoretical work has shown that active learning strategies can achieve an exponential reduction of the generalization error *with high probability* (see Balcan et al., 2006; Freund et al., 1997). In this contribution, we put the emphasis on a mathematically rigorous derivation of *hard* upper bounds for the generalization error. This is in contrast to other studies which give bounds in probability (see Freund et al., 1997) or discuss asymptotic behavior (see Bach, 2007).

We use a geometrical approach to active learning which is based on the concept of a version space (see Mitchell, 1982; Tong and Koller, 2001). Loosely speaking, given a set of predictors and a set of labeled training data, "version space" denotes the set of models whose predictions are consistent with the training data. If a version space can be defined, active learning strategies should evaluate data points which allow the learning machine to maximally reduce the "size" of its version space at every learning step. Some active learning algorithms try to halve the volume of the version space (see Tong and Koller, 2001). In contrast to this, our approach is to reduce the *maximal distance* between pairs of points that belong to the version space. We prefer distance over volume simply because it is impossible to compute the exact volume of version space in high dimensions. A detailed discussion of this problem can be found at the end of Section 7. For special types of data spaces, our method of maximal length reduction coincides with volume reduction. In this case, we observe a rapid (that is, exponential) progress in learning, as is explicitly shown in Section 4 and Proposition 9 of Section 6 for data which is arbitrarily distributed on an n-dimensional torus.

In the following we will consider the simple case of a separable, binary classification problem. Assuming the knowledge of the data density μ on the data space M, the generalization distance $d^G(c_1,c_2)$ of two classifiers $c_1,c_2 : M \to \mathbb{Z}_2 := \{0,1\}$ is given by the integral

$$d^G(c_1,c_2) := \frac{1}{\mathrm{Vol}(M)} \int_{D(c_1,c_2)} \omega,$$

where $D(c_1,c_2) := \{x \in M \mid c_1(x) \neq c_2(x)\}$ is the set of points which are classified differently by the two classifiers, and ω denotes the volume form of M (see Section 2). If we further assume the labeling of the data to be generated by an unknown classifier $c^* : M \to \mathbb{Z}_2$, the important question is: *Can we give tight upper bounds on the generalization error $d^G(c,c^*)$ of some classifier c, and can we reduce this bound during learning in an optimal way?* Inspired by this question, we introduce a constructive active learning algorithm which reduces such a bound by successive subdivisions of the version space on a hypersphere. This then allows us to compute exact and tight generalization error bounds for several classes of data densities. After deriving the bounds for the case of an uniform distribution on the hypersphere, we use the notion of Riemannian isometries to extend the algorithm and the error bounds to a set of selected data densities on other data spaces including $\mathbb{R}^n$. In a second step, we extend our results to product manifolds in order to obtain sharp error bounds for a larger set of data densities. Finally, we provide bounds for arbitrary densities on $\mathbb{R}^n$, hyperspheres and products thereof.

The article is organized as follows: After introducing the geometric setup of active learning for binary classification (Section 2), we formulate a learning method for data distributed on the unit sphere $S^n \subset \mathbb{R}^{n+1}$ (Section 3). Thereafter, we extend the basic algorithm to a broader class

of separable, binary classification problems using isometries induced by charts of the sphere and products of hyperspheres (Sections 4, 5). This will include perceptrons (that is, linear classifiers with bias on $\mathbb{R}^n$) as a special case. Upper bounds are provided for the generalization error for non-constant data densities for binary classification problems on the sphere (Section 6) and for linear classifiers without bias in $\mathbb{R}^n$ (Section 7). Finally, Section 8 provides an empirical evaluation of the geometric method. As our focus lies on a theoretical analysis of active learning algorithms, applications of the proposed algorithm to concrete problems have to be of second importance.

2. A Geometric Setup for Active Learning

In the sequel, we will apply some standard constructions from differential geometry. We refer to Appendix B for a quick introduction to the terminology.

Let M be an n-dimensional compact manifold, the *data space*, equipped with a Riemannian metric g. One might object to this type of data space, since the compactness assumption seems to rule out the most important instance of data space, the Euclidean space $\mathbb{R}^n$. However, this is not the case, because $\mathbb{R}^n$, or any submanifold therein, can be embedded into S^n, the n-sphere, by the inverse of stereographic projection (see Section 4). Recently, spherical data spaces have received some attention in machine learning (see Lebanon and Lafferty, 2004; Belkin and Niyogi, 2004; Minh et al., 2006).

We assume the existence of an *unknown binary classifier* $c^* : M \to \mathbb{Z}_2 := \{0,1\}$ and a given set of labeled data points $\{(x_1,y_1),\ldots,(x_I,y_I))\}$, $(x_i,y_i) \in M \times \mathbb{Z}_2$, with correct labels, that is, $c^*(x_i) = y_i$. Now, the *binary classification problem* asks for an approximation $c : M \to \mathbb{Z}_2$ which minimizes the *generalization error*, that is, the probability of misclassification of data points. This can be formalized as follows.

The Riemannian volume form ω which belongs to the metric g is given in local coordinates $x : M \supset U \to \mathbb{R}^n$ by

$$\omega = \sqrt{\det(g)}dx_1 \wedge \ldots \wedge dx_n, \tag{1}$$

where U is some open chart domain in M. We assume that the Riemannian volume form ω (see Equation 1) represents up to a scaling factor $\mathrm{Vol}(M) := \int_M \omega$ the p.d.f. of the data points $x \in M$. This allows us to interpret the probability of disagreement between two classifiers c_1, c_2 as a distance measure[1] on the set of all classifiers:

$$d^G(c_1,c_2) := \frac{1}{\mathrm{Vol}(M)} \int_{D(c_1,c_2)} \omega, \tag{2}$$

where $D(c_1,c_2) := \{x \in M \mid c_1(x) \neq c_2(x)\}$ is called the *disagreement area*. In these terms, the generalization error of c is given by $d^G(c,c^*)$.

3. Subdivisions of the Sphere

In order to be able to compute upper bounds on d^G, we need to impose restrictions on the data space M as well as on the set of classifiers.

1. Depending on the regularity conditions imposed on the classifiers, it might happen that $d^G(c_1,c_2) = 0$ while $c_1 \neq c_2$ on a set of measure zero.

We start with the following simple setup: Let $M = S^n = \{x \in \mathbb{R}^{n+1} \mid \langle x,x\rangle = 1\}$, the n-sphere with its canonical Riemannian metric and denote by C the set of *hemisphere classifiers*

$$c : S^n \to \mathbb{Z}_2, \quad c(x) := \begin{cases} 1 & : \quad \langle x,p\rangle \geq 0 \\ 0 & : \quad \langle x,p\rangle < 0 \end{cases} .$$

Here, $p \in S^n$ is the center of the hemisphere, and $\langle .,. \rangle$ denotes the Euclidean scalar product in $\mathbb{R}^{n+1}$. This setup implies that the data are uniformly distributed on the sphere. The use of closed hemispheres as classifiers is appropriate for spherical data, since hemispheres are the direct analog to half spaces in Euclidean geometry. A simple, yet crucial, observation is the duality $C = S^n$. By a slight abuse of notation, we use the symbol c to denote both, the classifier and the center of the hemisphere. Concerning the generalization distance of two classifiers we have the following proposition.

Proposition 1 *For hemispheres $c_1, c_2 \in C$*

$$d^G(c_1,c_2) = \frac{1}{\pi} d(c_1,c_2),$$

where $d(c_1,c_2) := \arccos(\langle c_1,c_2\rangle)$ is the geodesic distance on S^n.

Proof The disagreement area $D(c_1,c_2)$ consists of two congruent lunes on the sphere. The area of a lune is proportional to its opening angle $\alpha = d(c_1,c_2)$. Since the total volume of the unit sphere $V := \int_{S^n} \omega$ with respect to its canonical Riemannian metric and volume form is not equal to one, we have to normalize:

$$d^G(c_1,c_2) = \frac{1}{V} \int_{D(c_1,c_2)} \omega = \frac{1}{V} \left(\frac{\alpha}{\pi} V \right) = \frac{1}{\pi} d(c_1,c_2).$$

■

We assume the true classifier c^* to be some unknown hemisphere. If $(x,1)$ is a labeled data point, it follows that c^* is contained in the closed hemisphere around x. Thus, given a labeled set $\{(x_1,y_1),\ldots,(x_I,y_I))\}$, c^* is contained in the intersection V of the corresponding hemispheres. The *version space* of a labeled set (see Mitchell, 1982; Tong and Koller, 2001) is defined as the set of classifiers which are consistent with the given labeled data. In our case, the version space coincides with the intersection V. Geometrically, V is a convex spherical polytope, a high-dimensional generalization of a spherical polygon whose edges are segments of great circles and whose $(n-1)$-dimensional facets are segments of $(n-1)$-dimensional great spheres within S^n.

In theory, one can compute the vertices of the version space of any finite labeled set. An iterative algorithm would reduce this polytope by taking intersections with hemispheres corresponding to new data points. Unfortunately, during this process, the number of vertices of the polytope grows exponentially. Thus, the computational costs render its explicit computation impossible even for low dimensions.

One possibility to reduce the immense computational complexity of polytopes is to work with *spherical simplices*. This motivates the following active learning algorithm:

Definition 2 (Simplex algorithm)

1. *Specify some maximal edge length $\varepsilon > 0$ as termination criterion. Choose a random orthogonal matrix. Ask for the labels of the $n+1$ column vectors. This results in a set of admissible classifiers $S \subset S^n$ which is an equilateral simplex on S^n.*
2. *Select one of the edges of maximal length of the current simplex S. If its length is less than ε, go to step 7. Otherwise, compute its midpoint m.*
3. *Compute a unit normal u of the plane in $\mathbb{R}^{n+1}$ spanned by m together with all vertices of S which do not belong to the edge through m.*
4. *Ask for the label l of u. If $l = 0$, replace u by $-u$.*
5. *Replace the old simplex S by the part* in direction of *u, that is, the simplex which has as vertices m as well as all vertices of S with exception of that end point e of the chosen longest edge for which $\langle u, e \rangle < 0$.*
6. *Repeat with step 2.*
7. *Return the simplex' center of mass $c \in S$ as the learned classifier.*

Figure 1 illustrates one iteration of the simplex algorithm on the sphere S^2. The "random orthogonal

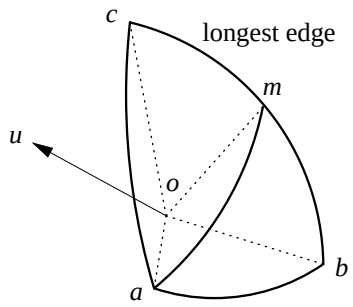

Figure 1: The drawing above shows one iteration of the simplex algorithm for the two-dimensional case, that is, for spherical triangles on the unit sphere S^2. The current version space is the spherical triangle (a,b,c). The longest edge (b,c) is about to be cut at its midpoint m. Together with the origin o, the vertex a and the point m define a plane in $\mathbb{R}^3$ one of whose unit normal vectors is $u \in S^2$. Depending on the label of u, the new triangle will be either (a,b,m) or (a,m,c).

matrix" that occurs in step one of the algorithm is meant to be drawn from the uniform distribution on the Lie group

$$O(n) = \{A \in GL(n) \mid AA^t = I\}$$

of orthogonal real matrices. A practical approach to the problem of generating such matrices can be found in Stewart (1980). The worst case generalization error after each iteration can be computed

by evaluating the maximum spherical distance of the chosen classifier to the vertices of the spherical simplex. To be explicit, the following statement holds:

Proposition 3 *If S is the current simplex from the simplex algorithm (see Definition 2) with vertices $v_1, \ldots, v_{n+1} \in S^n$, and $c \in S$ some classifier,*

$$d^G(c^*, c) \leq \max_{i,j} d(v_i, v_j) = \max_{i,j} \arccos\langle v_i, v_j \rangle.$$

This bound is tight and attainable if we allow any element of the version space to be the learned classifier. Moreover, if $c \in S$ denotes the center of mass, then

$$d^G(c^*, c) \leq \max_{i} d(c, v_i) = \max_{i} \arccos\langle c, v_i \rangle$$

is a tight and attainable upper bound for the generalization error.

Proof Within a simplex, the maximal distance of two points is realized by pairs of vertices. Now the first inequality follows from proposition 1. If all elements of the simplex are admissible classifiers, the bound is tight. The second inequality follows from the convexity of the simplex. ■

Clearly, the maximal edge length of the simplex S converges to zero. In Appendix A, we derive $O((n+1)^3)$ as a rough complexity estimate for one iteration of the simplex algorithm (see Definition 2).

Another question concerning the convergence of the algorithm is: How many iterations are needed until the maximum edge length of the initial simplex starts to drop? To this end, we have the following proposition whose proof is given in Appendix A.

Proposition 4 *Let S be the initial equilateral simplex from the simplex algorithm (see Definition 2). Let $k \in \mathbb{N}$ be the number of steps needed until the maximum of the edge lengths drops. Then*

$$n \leq k \leq \frac{n(n+1)}{2},$$

and these bounds are tight and attainable.

4. Extensions by Isometries

We will now extend our results to other data spaces by applying the concept of isometries. The easiest method to obtain isometries from the n-sphere to other data spaces is to consider charts of the sphere together with the induced metric. Being isometries, they preserve the geometry, and any generalization bounds derived for the sphere can be applied without modifications. Combining isometries with the product construction of Section 5, we end up with a large family of data densities on $\mathbb{R}^n$ to which our results are directly applicable. In Section 6, we will loosen our preconditions even further and consider the general case of arbitrary data densities.

We begin with the discussion of the stereographic chart of the n-sphere.

Stereographic chart. The stereographic projection

$$\sigma_N : S^n \setminus \{N\} \to \mathbb{R}^n,$$
$$(x_1, \ldots, x_{n+1}) \mapsto \frac{(x_1, \ldots, x_n)}{1 - x_{n+1}},$$

where $N = (0,\ldots,0,1)$ denotes the north pole, is an isometry of $S^n \setminus \{N\}$ to $(\mathbb{R}^n, g)$, where the Riemannian metric g is given by

$$g_x(v,w) = \frac{4}{(1+\|x\|^2)^2}\langle v,w\rangle.$$

See Figure 2 for an illustration of the two-dimensional case. Hence, we can *identify* $S^n \setminus \{N\}$ with

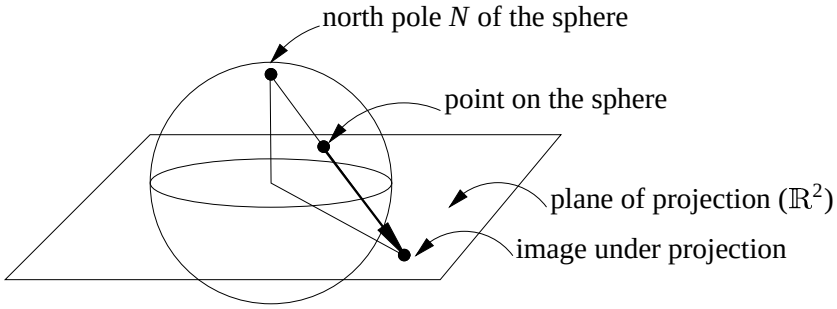

Figure 2: The stereographic projection $S^2 \setminus \{N\} \to \mathbb{R}^2$ is a diffeomorphism from the sphere with the north pole removed to the plane. It distorts lengths but preserves angles. If one considers a ray from the north pole to some point on the plane, the stereographic projection will map the intersection point of this ray with the sphere onto its intersection point with the plane.

$(\mathbb{R}^n, g)$, and the induced Riemannian volume form is

$$\omega_x = \frac{2^n}{(1+\|x\|^2)^n} dx_1 \wedge \ldots \wedge dx_n,$$

where $dx_1 \wedge \ldots \wedge dx_n$ denotes the Euclidean volume on $\mathbb{R}^n$ (the determinant). If the given data density on $\mathbb{R}^n$ is (up to a constant factor) equal to ω, the data can be considered to lie on S^n with constant density, and our error bounds hold. When viewed under stereographic projection, our spherical classifiers fall in three categories: If the boundary of the hemisphere, which is a great $(n-1)$-sphere within S^n, contains the north pole, its projection is a hyperplane through the origin in $\mathbb{R}^n$. The equatorial great sphere $\{x \in S^n \subset \mathbb{R}^{n+1} \mid x_{n+1} = 0\}$ is projected onto $S^{n-1} \subset \mathbb{R}^n$. All other great spheres become spheres intersecting $S^{n-1} \subset \mathbb{R}^n$ orthogonally. Hence, any data which is separable by these classes of hypersurfaces in $\mathbb{R}^n$ is separable by great spheres on S^n and vice versa.

Gnomonic chart. The gnomonic projection

$$\varphi : S^n \supset \{x_{n+1} > 0\} \to \mathbb{R}^n,$$
$$(x_1,\ldots,x_{n+1}) \mapsto \frac{(x_1,\ldots,x_n)}{x_{n+1}},$$

generates on $\mathbb{R}^n$ the metric

$$g = \frac{1}{(1+x_1^2+\ldots+x_n^2)^2} \begin{pmatrix} s_1 & & -x_i x_j \\ & \ddots & \\ -x_i x_j & & s_n \end{pmatrix},$$

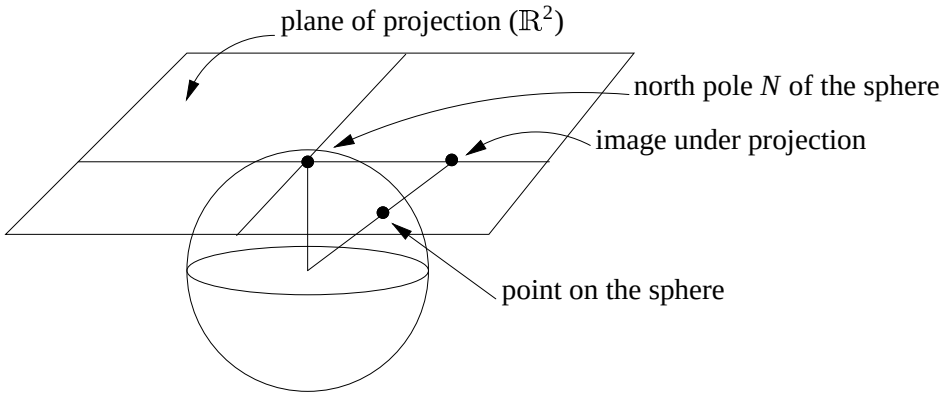

Figure 3: Illustration of the gnomonic projection for the case $S^2 \supset \{x_3 > 0\} \to \mathbb{R}^2$. In this case, we embed $\mathbb{R}^2$ at height one into $\mathbb{R}^3$, such that it touches the sphere at the north pole. Rays are sent from the center of the sphere. Their intersection points with the sphere are mapped to the corresponding intersection points with the plane. The gnomonic projection distorts lengths as well as angles, but maps great circles to straight lines.

where we have used the abbreviation $s_i := 1 + x_1^2 + \ldots + x_n^2 - x_i^2$. Figure 3 illustrates the gnomonic projection in two dimensions. As was the case with the stereographic chart, gnomonic projection is an isometry from the upper half-sphere to $(\mathbb{R}^n, g)$. Using Equation 1, we have for $x_2 = \ldots = x_n = 0$

$$\sqrt{\det(g)} = \frac{1}{(1+x_1^2)^{\frac{n+1}{2}}}.$$

Since the scaling function of the volume form ω has to be rotationally symmetric, it follows that

$$\omega_x = \frac{1}{(1+\|x\|^2)^{\frac{n+1}{2}}}\, dx_1 \wedge \ldots \wedge dx_n. \tag{3}$$

Note that our separating great spheres are projected to affine hyperplanes in $\mathbb{R}^n$. Therefore, the classical approach to the binary classification problem using *linear classifiers with bias* can be considered a special case of our spherical setup. More precisely, the strict error bounds derived for our algorithm apply to linear classifiers with bias on $\mathbb{R}^n$ if and only if the data density on $\mathbb{R}^n$ is given by Equation 3. For an analysis that applies to a greater variety of densities we refer to Section 6.

As a byproduct, formula 3 also clarifies the arguments given in the proof of Theorem 4 of Freund et al. (1997) which estimates the information gain of queries made by the Query by Committee algorithm. In the proof, the scaling factor of the volume form of the gnomonic chart is estimated using infinitesimals. The explicit formula for the volume form given above makes this more lucid.

5. Products of Spheres

We now extend the simplex algorithm (see Definition 2) to other data manifolds and other sets of classifiers using a simple product construction. The main purpose of this section is to obtain

building blocks for data manifolds and data densities which later can be combined to produce more sophisticated examples. We consider product data manifolds of the type

$$(M,g) = (S^{n_1},g^1) \times \ldots \times (S^{n_k},g^k), \tag{4}$$

where each factor (S^{n_j},g^j) is a unit sphere of dimension n_j with its standard metric g^j. For a point $x = (x_1,\ldots,x_k) \in M$ and tangent vectors $X = (X_1,\ldots,X_k), Y = (Y_1,\ldots,Y_k) \in T_xM$ we have

$$g_x(X,Y) = \sum_{j=1}^{k} g^j_{x_j}(X_j,Y_j).$$

The Riemannian volume form of the product is given in local coordinates by

$$\omega = \prod_{j=1}^{k} \sqrt{\det(g^j)} = \bigwedge_{j=1}^{k} \omega^j,$$

where ω^j denotes the volume form of the jth factor. On the product manifold M, we consider classifiers which are products of hemispheres, that is, $C = C^1 \times \ldots \times C^k$, where the $C^j = S^{n_j}$ are the individual sets of hemispheres defined in Section 3 and a classifier $c \in C$ is given by

$$c : M \to \mathbb{Z}_2, \quad c(x) := \begin{cases} 1 & : \quad \langle x, c_j \rangle \geq 0 \, \forall \, j \\ 0 & : \quad \text{otherwise} \end{cases} .$$

Due to the simplicity of the product structure, we arrive at the following formula for the generalization metric:

Proposition 5 *For products of hemispheres $c^1, c^2 \in C$,*

$$d^G(c^1,c^2) = \frac{1}{2^{k-1}} \left(1 - \frac{1}{\pi^k} \prod_{j=1}^{k} (\pi - d_j(c^1_j, c^2_j)), \right)$$

where d_j is the geodesic distance on S^{n_j}.

Proof Since each component of a classifier is a hemisphere,

$$\mathrm{Vol}(c^1_j \cap c^2_j) = \mathrm{Vol}(S^{n_j}) \frac{\pi - d_j(c^1_j, c^2_j)}{2\pi}.$$

Furthermore, the volume of a product of such hemispheres is given by

$$\mathrm{Vol}(c^1) = \prod_{j=1}^{k} \mathrm{Vol}(c^1_j) = \prod_{j=1}^{k} \frac{\mathrm{Vol}(S^{n_j})}{2} = \frac{\mathrm{Vol}(M)}{2^k}.$$

Inserting this into

$$\mathrm{Vol}(D(c^1,c^2)) = \mathrm{Vol}(c^1) + \mathrm{Vol}(c^2) - 2\mathrm{Vol}(c^1 \cap c^2)$$

where $D(c^1,c^2)$ denotes the disagreement area, yields the proposition. ■

This leads to the extended spherical simplex algorithm:

Definition 6 (Extended simplex algorithm)

1. *Specify some maximal edge length* $\varepsilon > 0$ *as termination criterion. For each factor* S^{n_j}*, create an initial simplex using a random orthogonal matrix whose columns are interpreted as a basis for* $\mathbb{R}^{n_j+1}$*. This results in a product S of equilateral simplices.*

2. *Find one of the edges of maximal length of each factor of the current simplex product S. If all the respective lengths are less than* ε*, go to step 7. Otherwise, compute the midpoints* $m_1, \ldots, m_k$*.*

3. *Compute the corresponding unit normals* u_j*.*

4. *Ask for the labels* l_j *of* u_j*. If* $l_j = 0$*, replace* u_j *by* $-u_j$*.*

5. *Replace the old simplex product S by the product of the parts* in direction of u_j*.*

6. *Repeat with step 2.*

7. *For each factor, compute its center of mass* c_j*, and return* $(c_1, \ldots, c_k) \in S$ *as the learned classifier.*

In parallel to the case of a single sphere, the minimization of maximal edge lengths forces convergence. Note, that if k denotes the number of factors in the product of spheres, then k training points are needed to carry out one iteration of the algorithm. The worst case generalization error after each step is bounded as follows.

Proposition 7 *If S is the current product simplex from the extended simplex algorithm (see Definition 6) with maximal edge lengths* $d_1, \ldots, d_k$ *of its factors, and* $c \in S$ *some classifier,*

$$d^G(c^*, c) \leq \frac{1}{2^{k-1}} \left(1 - \frac{1}{\pi^k} \prod_{j=1}^{k} (\pi - d_j) \right).$$

This bound is tight and attainable if we allow any element of the version space to be the learned classifier.

Proof In analogy to the case of a single sphere, this follows from proposition 5. ■

The complexity estimate for one iteration of the extended simplex algorithm is

$$O((n_1+1)^3 + \ldots + (n_k+1)^3).$$

Here, $n_1, \ldots, n_k$ denote the dimensions of the k individual factors of the product data space. This estimate can be deduced directly from the complexity analysis of the simplex algorithm given in Appendix A.

Products combined with isometries. We now apply the isometries discussed in Section 4 to product manifolds. For each factor in Equation 4, we may choose one of stereographic or gnomonic projection.

$$\begin{array}{rccc} M^{n_1+\ldots+n_k} = & S^{n_1} & \times\ldots\times & S^{n_k}, \\ & f_1\downarrow & \cdots & \downarrow f_k \\ \mathbb{R}^{n_1+\ldots+n_k} = & \mathbb{R}^{n_1} & \times\ldots\times & \mathbb{R}^{n_k}. \end{array}$$

This results in product densities on $\mathbb{R}^n = \mathbb{R}^{n_1+\ldots+n_k}$, represented by the volume form

$$\omega = \prod_{j=1}^{k} \omega^j$$

with factors ω^j given by either

$$\omega_x^j = \frac{2^{n_j}}{(1+\|x\|^2)^{n_j}} dx_1 \wedge \ldots \wedge dx_{n_j}$$

(stereographic projection) or

$$\omega_x^j = \frac{1}{(1+\|x\|^2)^{\frac{n_j+1}{2}}} dx_1 \wedge \ldots \wedge dx_{n_j}$$

(gnomonic projection). Similarly, the projected separating hypersurfaces are products of the individual projections. One could now go on to produce *many* more families of compatible densities by working with different charts, but instead we turn our attention to an important special case.

The *n*-torus. A particular case is the gnomonic projection of the *n*-torus

$$T^n = \underbrace{S^1 \times \ldots \times S^1}_{n \text{ factors}},$$

which yields a scaled version of the Cauchy distribution on $\mathbb{R}^n$:

$$\omega_x = \prod_{i=1}^{n} \frac{1}{1+x_i^2} dx_1 \wedge \ldots \wedge dx_n.$$

Here, we take S^1 to be the unit circle which results in T^n having total mass $(2\pi)^n$. Since there is one circle S^1 per axis, the projected classifiers are *axis parallel corners* in $\mathbb{R}^n$. One iteration of the algorithm consumes n labeled data points, because T^n is made up of n individual factors. At each step of the extended simplex algorithm, the version space is a *hypercube* of equal edge length l. Therefore, we do not need to compare edge lengths. One step results in halving all the edges, and the volume is divided by 2^n. Hence, if V_i denotes the volume after i iterations, we have

$$V_i = \frac{\pi}{2^{in}}.$$

In analogy, the maximal generalization error G_i of the center of mass classifier after i steps is given by

$$G_i = \frac{\sqrt{n}\pi}{2^i}.$$

Thus, we observe an exponential decrease in volume and in the tight upper bound for the generalization error.

6. Aspherical Data Manifolds

Up to now, our methods apply only to such data manifolds which are isometric to products of subsets of unit spheres. We now loosen this assumption by looking at all oriented Riemannian data manifolds M which admit an orientation preserving[2] diffeomorphism

$$\Phi : M \to S^n,$$

that is, Φ is a smooth bijective map which has a smooth inverse. The Riemannian volume form belonging to the metric g on M induces a volume form $\widetilde{\omega}$ on S^n which, in general, is not equal to the spherical volume ω. More precisely,

$$\widetilde{\omega} = f\omega$$

with some smooth positive scaling function $f : S^n \to \mathbb{R}^+$. Note that *all* volume forms (or smooth positive densities) on S^n can be written in this form. This results in the formula

$$d^G(c_1, c_2) := \frac{1}{\widetilde{\mathrm{Vol}}(S^n)} \int_{D(c_1,c_2)} f\omega$$

for the generalization metric. An illustation of a non-uniform density on the sphere is given in Figure 4 in Section 8 where the reader also finds empirical results for this case.

Concerning the set of admissible classifiers, let us keep the assumption that the Φ-image of the data is separable by hemisphere classifiers. If we know that the true classifier c^* lies in some subset $S \subset S^n$ the worst case generalization error of some classifier $c \in S$ is bounded from above by

$$d^G(c, c^*) \leq \sup_{\widetilde{c} \in S} d^G(\widetilde{c}, c^*).$$

Therefore, the simplex algorithm (see Definition 2) will still converge, as it reduces an upper bound of the generalization error. Its rate of convergence will depend on the properties of the density.

Nevertheless, we can force a simple upper bound by assuming the deviation of the induced volume form from spherical volume to be small:

$$\sup_{x \in S^n} |1 - f(x)| < \varepsilon.$$

This implies

Proposition 8 *Let ω be the canonical volume form of S^n. Denote by $d^{\widetilde{G}}$ the generalization distance induced by the scaled volume form $\widetilde{\omega} = f\omega$ where $f : S^n \to \mathbb{R}^+$ is some positive smooth scaling function. If $\sup_{x \in S^n} |1 - f(x)| < \varepsilon$ for some $\varepsilon > 0$ then*

$$d^{\widetilde{G}}(c_1, c_2) \leq \frac{(1+\varepsilon)\mathrm{Vol}(S^n)}{\pi \widetilde{\mathrm{Vol}}(S^n)} d(c_1, c_2),$$

where d is the canonical geodesic distance of S^n. In this formula, $\mathrm{Vol}(S^n) := \int_{S^n} \omega$ and $\widetilde{\mathrm{Vol}}(S^n) := \int_{S^n} \widetilde{\omega}$ denote the volumina of S^n with respect to ω and $\widetilde{\omega} := f\omega$.

2. A map between oriented Riemannian manifolds is orientation preserving if its functional determinant is positive.

Proof Using the definition of the generalization distance in Equation 2 and applying proposition 1, we compute

$$\begin{aligned} d^{\widetilde{G}}(c_1,c_2) &= \frac{1}{\widetilde{\mathrm{Vol}}(S^n)} \int_{D(c_1,c_2)} f\omega \\ &\leq \frac{1}{\widetilde{\mathrm{Vol}}(S^n)} \int_{D(c_1,c_2)} (1+\varepsilon)\omega \\ &= \frac{(1+\varepsilon)\mathrm{Vol}(S^n)}{\widetilde{\mathrm{Vol}}(S^n)} d^G(c_1,c_2) \\ &= \frac{(1+\varepsilon)\mathrm{Vol}(S^n)}{\pi\widetilde{\mathrm{Vol}}(S^n)} d(c_1,c_2). \end{aligned}$$

■

Inserting the above upper bound into proposition 3 we obtain

Proposition 9 *Let S be the current simplex from the simplex algorithm (see Definition 2) with vertices $v_1,\ldots,v_{n+1} \in S^n$, and $c \in S \subset S^n$ an arbitrary classifier, not necessarily the center of mass. If $c^* \in S^n$ denotes the unknown true classifier, the generalization error of c is bounded by*

$$d^{\widetilde{G}}(c,c^*) \leq \frac{(1+\varepsilon)\mathrm{Vol}(S^n)}{\pi\widetilde{\mathrm{Vol}}(S^n)} \max_{i,j} d(v_i,v_j),$$

where $d(v_i,v_j)$ denotes the spherical distance of the vertices.

The usefulness of the above proposition depends on how much the scaled volume form $f\omega$ deviates from the canonical spherical volume ω. In the case of the n-torus, the same arguments as given at the end of Section 4 yield an exponential decrease of the volume of the version space as well as of the upper bound for the generalization error—*regardless of the data density under consideration*. The only difference is the newly introduced constant ε which may affect the absolute rate but not the functional form of convergence.

7. Linear Classifiers without Bias

We now return to the case of linear classifiers on the Euclidean space $\mathbb{R}^n$ which commonly appear in the machine learning literature. The corresponding separating hypersurfaces are linear subspaces, that is, hyperplanes through the origin, of $\mathbb{R}^n$.

Consider the data space $M = \mathbb{R}^n$. Then, the set C of *linear classifiers without bias* consists of maps

$$c : \mathbb{R}^n \to \mathbb{Z}_2, \quad c(x) := \begin{cases} 1 & : \quad \langle x,p\rangle \geq 0 \\ 0 & : \quad \langle x,p\rangle < 0 \end{cases},$$

where $p \in S^n$ is the unit normal vector of an oriented $(n-1)$-dimensional plane through the origin. Therefore, we can identify C with the unit n-sphere, $C = S^n$. In the following, we use c to denote both, the classifier and its corresponding unit normal.

Consider some data density $f : \mathbb{R}^n \to \mathbb{R}$. Since the origin is always classified as $+1$ by all classifiers we may remove it from the data space, and the remaining data space is given by

$$M = \mathbb{R}^n \setminus \{0\}.$$

Using polar coordinates (s,r), we can write M as the product[3]

$$M = S^{n-1} \times \mathbb{R}^+, \quad s = \frac{x}{\|x\|}, r = \|x\|.$$

For a given $s \in S^{n-1}$, we will call the subset $F_s := \{(s,r) \mid r > 0\}$ the *fiber* over s. Since the data is assumed to be separable by at least one element of C any two points belonging to the same fiber have the same label.

The area $D(c_1,c_2)$ of disagreement between two classifiers c_1, c_2 is given by

$$D(c_1,c_2) = \{(s,r) \in M \mid \langle c_1, rs\rangle\langle c_2, rs\rangle < 0\} = \{(s,r) \in M \mid \langle c_1, s\rangle\langle c_2, s\rangle < 0\}.$$

The generalization distance is given by

$$d^G(c_1,c_2) = \int_{D(c_1,c_2)} f dx = \int_{\{s \in S^{n-1} \mid F_s \subset D(c_1,c_2)\}} \left(\int_{F_s} f(s,.) dr \right) ds, \tag{5}$$

where dx, dr, ds denote the canonical volume forms on $\mathbb{R}^n$, F_s, and S^{n-1}. It may happen that different fibers F_s have different mass in the sense that

$$S^{n-1} \to \mathbb{R}, \quad s \mapsto \int_{F_s} f(s,.) dr$$

is a non-constant function. If we rule out this case we end up with the following proposition:

Proposition 10 *The generalization distance of any two linear classifiers c_1, c_2 is given by*

$$d^G(c_1,c_2) = \frac{\lambda}{\pi} d(c_1,c_2),$$

where $d(c_1,c_2) = \arccos\langle c_1, c_2\rangle$ is the geodesic distance on S^n, if and only if the fiber mass is equal to a positive constant,

$$\int_{F_s} f(s,.) dr = \lambda > 0 \ \ \forall s \in S^n.$$

Proof This follows by applying proposition 1 to Equation 5. ■

The precondition of proposition 10 does *not* assume that the density f is rotationally invariant on $\mathbb{R}^{n+1}$. Instead, it assumes the *accumulated* density to be invariant on the sphere. Linear classification problems on non-constant densities which fulfill this condition map to classification problems with hemisphere classifiers for the uniform density on the sphere. Consequently, all results derived for the spherical simplex algorithm (see Definition 2) apply, including the hard upper bounds on the generalization error. In particular, we deduce the following result from Proposition 3

3. We use the convention $\mathbb{R}^+ := \{r \in \mathbb{R} \mid r > 0\}$.

Proposition 11 *If S is the current simplex from the simplex algorithm (see Definition 2) with vertices $v_1, \ldots, v_{n+1} \in S^n$, c^* denotes the unknown true classifier and $c \in S$ an arbitrary classifier,*

$$d^G(c^*, c) \leq \lambda \max_{i,j} d(v_i, v_j) = \lambda \max_{i,j} \arccos\langle v_i, v_j \rangle.$$

As in Proposition 10, $\lambda > 0$ denotes the fiber mass. This bound is tight and attainable if we allow any element of the version space to be the learned classifier. Moreover, if $c \in S$ denotes the center of mass, then

$$d^G(c^*, c) \leq \lambda \max_{i} d(c, v_i) = \lambda \max_{i} \arccos\langle c, v_i \rangle.$$

is a tight and attainable upper bound for the generalization error.

Relation to SVM methods. Proposition 10 also sheds new light on some active learning strategies that use Support Vector Machines (SVM). A SVM classifier can be interpreted as an approximation of the center of the largest inscribable hypersphere of the version space on S^{n-1} (see Herbrich, 2002). Let us denote this center by $p^* \in V \subset S^{n-1}$, where $V \subset S^{n-1}$ is the version space (a spherical polytope, see Section 3) on the hypersphere.

Tong and Koller (2001) argue that, despite its dependence on the particular shape of the version space, the center of the maximal inscribable hypersphere often lies close to "the center of the version space". Motivated by these insights, they propose the following pool-based strategy for selecting an unlabeled data point $x \in S^{n-1}$ to be labeled: Choose x such that the (spherical) distance from the $(n-2)$-dimensional great sphere

$$X := \{s \in S^{n-1} \mid \langle x, s \rangle = 0\}$$

to p^* is minimal. After the data point x is labeled, the version space will be cut into two pieces along the great sphere $X \subset S^{n-1}$. The goal of this strategy is to reduce the volume of the spherical polytope V as quickly as possible. Similar strategies can be found in Warmuth et al. (2002). Up to now, a closed formula for the volume of a n-spherical simplex *is not known*, not to speak of polytopes, hence, there is *no way* of computing the exact volume of a version space on a hypersphere. We refer to Milnor (1994) for a detailed discussion of this topic. Nevertheless, Monte-Carlo methods can be applied to obtain volume estimates.

Proposition 10 can now be applied. The SVM algorithm works with the Euclidean scalar product on $\mathbb{R}^n$, and therefore implicitly assumes the canonical Riemannian metric and volume form on the unit sphere. Proposition 10 tells us that the SVM approximation is theoretically justified if and only if the given data density induces (up to a constant factor) the uniform density on the sphere.

8. Experimental Results

The following results were obtained from a C++ implementation of the simplex algorithm (see Definition 2). The numerically most sensitive operation of the algorithm is the computation of a normal vector $u \in S^n$ to the hyperplane $H \subset \mathbb{R}^{n+1}$ whose intersection $I = H \cap S^n$ with S^n cuts the current simplex S into two pieces. In higher dimensions, say $n > 50$,

$$\int_{S^n \setminus \{x \in S^n \mid d(x,I) < \delta\}} \omega$$

becomes very small even for small values $\delta > 0$.[4] This means, nearly all the mass of S^n is concentrated within a thin tube of radius δ around I, which makes it hard to compute normal vectors. We avoid numerical problems by using the following procedure:

1. Select the basis of H given by the midpoint of the longest edge and all vertices of the simplex excluding the end points of the longest edge. Construct a corresponding orthonormal basis using the modified Gram-Schmidt algorithm (see Meyer, 2000).

2. Choose random points $x \in S^n$ until the projected length $\sqrt{\sum_i \langle x, h_i \rangle}$, where h_i are the orthonormal basis vectors of H, is less than a given constant $\varepsilon < 1$.

3. Use x to construct a unit vector which is orthogonal to all basis vectors h_i.

In order to test the performance of the simplex algorithm on spheres of different dimensions a series of numerical experiments was conducted, where the following quantities were measured.

Maximal edge length: If $(v_1, \ldots, v_{n+1})$ denote the vertices of the current simplex S of the simplex algorithm, the maximal edge length

$$\max_{i,j} d(v_1, v_j)$$

is an upper bound on the generalization error, regardless which point of the simplex is chosen as the learned classifier.

Maximal distance from center of mass: Let $c \in S$ denote the center of mass of S. Then the maximal distance between c and any other classifier from S is given by

$$\max_i d(c, v_i).$$

This yields a tight upper bound on the generalization error if we choose the center of mass as the learned classifier.

Approximate generalization error for the center of mass classifier c: We estimated the generalization error of c using the empirical average of the individual errors for 50,000 randomly selected "test" data points. Test data were sampled (i) from a uniform density on the sphere and (ii) from an aspherical density with two distinct "clusters" at opposite poles. The aspherical density was constructed by mapping the uniform distribution from an open parameter cuboid onto the sphere using n-spherical coordinates.

Figure 4 illustrates the relation between a sample drawn from this density on the sphere S^2 and its stereographic projection onto the plane $\mathbb{R}^2$. Densities of this kind are typical for binary classification problems. Any density with two peaks at $p_1, p_2 \in \mathbb{R}^n$ can be identified with a density on S^n with peaks at opposite poles: Firstly, we apply a translation to move the midpoint of the line $\overline{p_1 p_2}$ to the origin. Then we use a rotation to place p_1, p_2 on the x_1-axis. After a scaling, $p_1 = (-1, 0, \ldots, 0)$ and $p_2 = (+1, 0, \ldots, 0)$. Now inverse stereographic projection will map the peaks onto opposite poles.

4. For a comprehensive discussion of these effects we refer to Gromov (1999).

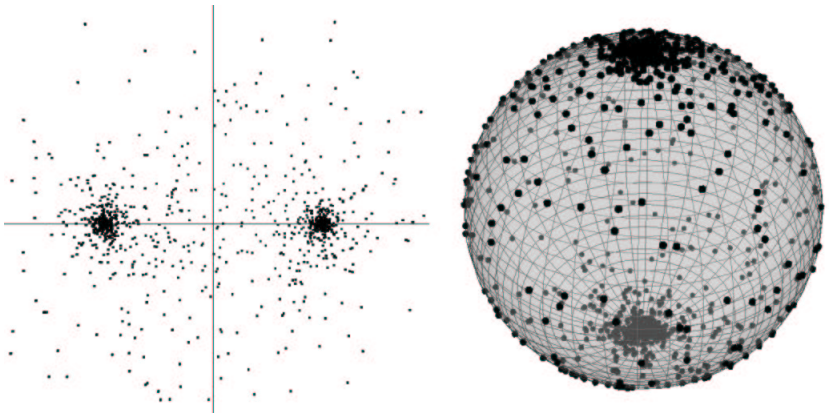

Figure 4: A data sample from a density with two peaks on $\mathbb{R}^2$ (left) and S^2 (right). The data points on the plane are the stereographic projections of the points on the sphere. On S^2, the two peaks are located at opposite poles.

The above quantities were computed for each step of the simplex algorithm. Averages and variances were calculated for 1,000 simulations, and averages were normalized to lie within the interval $[0,1]$. For every simulation, a true classifier was drawn from the uniform distribution on the sphere. Figures 5 and 6 show the resulting learning curves for the spheres $S^9 \subset \mathbb{R}^{10}$, $S^{29} \subset \mathbb{R}^{30}$, $S^{49} \subset \mathbb{R}^{50}$, and $S^{79} \subset \mathbb{R}^{80}$. The average maximal edge length as a function of the number of selected training data

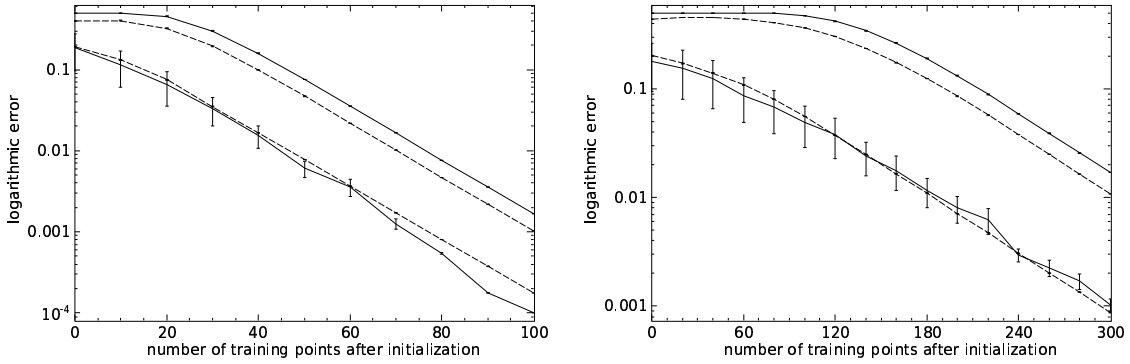

Figure 5: Learning curves on $S^9 \subset \mathbb{R}^{10}$ (left) and on $S^{29} \subset \mathbb{R}^{30}$ (right). The figures show the average maximal edge length (upper solid line), the average maximal distance from the simplex's center of mass (upper dashed line), and the average approximate generalization errors for the uniform (lower dashed line) and aspherical (lower solid line) data densities as a function of the number of selected training examples. Error bars indicate variances, however, only the approximate generalization error for the aspherical data density shows large fluctuations between simulation runs. Proposition 4 yields the bounds $9 \leq k_9 \leq 45$ and $29 \leq k_{29} \leq 435$ for the number k of steps needed before the maximal edge length starts to drop on S^9 and S^{29}.

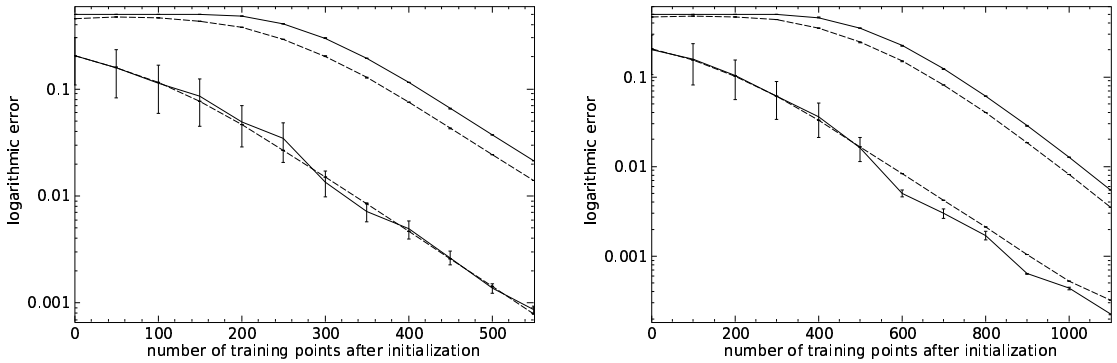

Figure 6: Learning curves on $S^{49} \subset \mathbb{R}^{50}$ (left) and $S^{79} \subset \mathbb{R}^{80}$ (right). For details see legend of Figure 5. Proposition 4 yields the bounds $49 \leq k_{49} \leq 1225$ and $79 \leq k_{79} \leq 3160$ for the number k of steps needed before the maximal edge length starts to drop on S^{49} and S^{79}.

shows an initial plateau, until the values begin to decrease in an approximately exponential fashion. The length of the plateau increases with the dimensionality of the sphere and is a direct result of Proposition 4. The average maximal distance from the center of mass rises initially (see Figure 7 for a magnified version of the initial segment of the learning curve), until a sudden drop occurs, again followed by a roughly exponential decrease. This can be explained as follows. The simplex algorithm is initialized with an equilateral simplex. During the first learning steps, the center of mass moves towards those vertices whose adjacent edges are cut already. This results in a slight increase of the maximal distance of the vertices from the center of mass. The simplex becomes a "thin pyramid" with small base, and the following drop in the plots then corresponds to a cut of a line connecting the apex to the base. The ratio between the edges connecting the apex to the base and the edges which are contained within the base is given by $\frac{2}{n-1}$, where n is the dimensionality of the sphere. Since this number tends to zero for $n \to \infty$ the sudden drop disappears in higher dimensions.

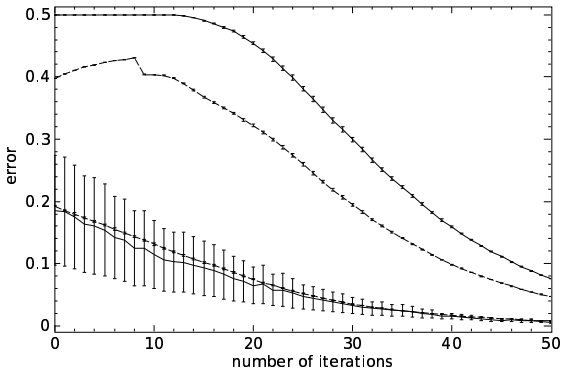

Figure 7: Initial learning curves on $S^9 \subset \mathbb{R}^{10}$ (see Figure 5, left), now plotted on a linear scale. For details see legend of Figure 5.

If the data is drawn from the spherical distribution, the approximate generalization error changes smoothly with the number of selected training data, and its variance is very small. For data distributed according to the aspherical (two cluster) density, the average approximate generalization error is similar, but the variance increases dramatically. Nevertheless, the numerical experiments show that the spherical simplex algorithm performs well even in the case of non-uniform densities.

Experimental results on product manifolds. So far, we restricted the experiments to single spheres instead of products, because the simulation of the extended simplex algorithm on

$$M = S^{n_1} \times \ldots \times S^{n_k},$$

is equivalent to the parallel execution of several copies of the basic algorithm. Nevertheless, it might be interesting to consider the special case of the n-torus T^n (see Section 5):

$$T^n = \underbrace{S^1 \times \ldots \times S^1}_{n \text{ factors}}.$$

The product structure of the torus reflects the fact that data is distributed independently on each factor. For the standard product density on T^n, volume and distances can be computed explicitly. Therefore, we consider only the non-uniform case. We consider a von Mises density (see Devroye, 1986) on the unit circle:

$$f : S^1 = \mathbb{R}/2\pi \to \mathbb{R}, \quad f(x) = \frac{\exp(\kappa \cos(x-\mu))}{2\pi I_0(\kappa)}.$$

with center $\mu \in [0, 2\pi]$ and width $\kappa \geq 0$. The symbol I_0 in the equation above denotes the modified Bessel function of the first kind of order zero. A technique for simulating the von Mises distribution can be found in Best and Fisher (1979). In order to obtain a density with two peaks on opposite poles of the circle, we superimpose two copies of f with $\mu = 0$ and $\mu = \pi$. This construction is applied to every factor S^1 of the torus $T^n = S^1 \times \ldots \times S^1$.

We implemented the extended simplex algorithm on the n-torus. Due to the product structure, numerical problems like those described at the beginning of Section 8 do not arise. For the case $n = 2$, the torus can be embedded into $\mathbb{R}^3$ using

$$S^1 \times S^1 \to \mathbb{R}^3, \quad (s,t) \mapsto \begin{pmatrix} (2+\cos t)\cos s \\ (2+\cos t)\sin s \\ \sin t \end{pmatrix}.$$

Using this mapping, we can visualize the iterations of the extended simplex algorithm on von Mises distributed data. Figure 8 shows a data sample as well as several iterations of the algorithm on the embedded torus. Figure 9 depicts the stereographic projection of a sample drawn from the von Mises distribution together with the projected classifier in $\mathbb{R}^2$.

Finally, we conducted experiments on the n-torus in order to obtain learning curves analogous to those on the n-sphere. As was shown in Section 5 distances and volumina on the n-torus can be computed explicitly provided the data density is uniform. Therefore, we focus on the approximal generalization error for data distributed according to the modified von Mises density described above. The approximation was done by evaluating the performance of the classifier on a data sample of 50,000 test points after each training step. The resulting values were averaged over 1,000

Figure 8: The extended simplex algorithm on the 2-torus. The large dot on the upper part of the torus represents the true classifier. Small dots depict positively (light gray) and negatively (dark) classified points drawn from the modified von Mises distribution. The meaning of the nested regions is the following (light to dark): positively classified area of the true classifier, version space after initialization (step one of the extended simplex algorithm), version space after first iteration, version space after second iteration.

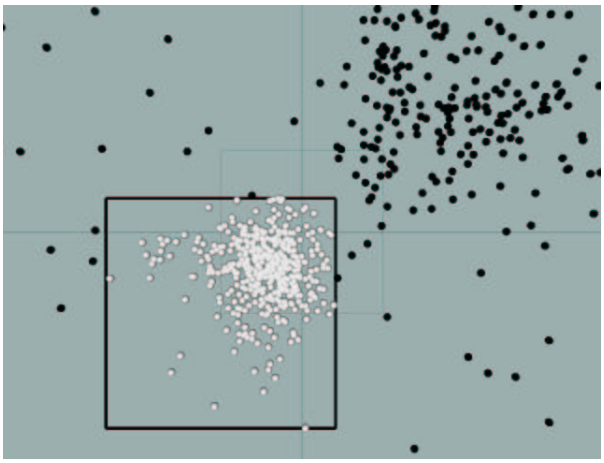

Figure 9: The image of a data sample from two superposed von Mises distributions on the two-dimensional torus under the stereographic projection (defined in Section 4) to $\mathbb{R}^2$. The black square represents the projected classifier. Light dots are classified positively, dark dots negatively.

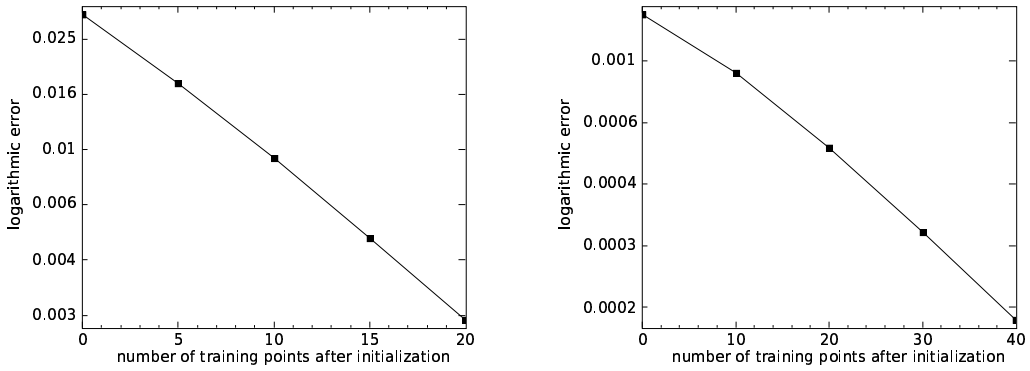

Figure 10: Learning curves on the 5-dimensional torus (left) and on the 10-dimensional torus (right). The figures show the average approximate generalization errors for the modified von Mises density as a function of the number of selected training examples. Since the variances are almost zero, they are not included in the diagram.

simulations. For every simulation, a true classifier was drawn from the uniform distribution on the torus. The resulting learning curves for dimensions $n = 5$ and $n = 10$ are shown in Figure 10.

Due to the product structure of the torus, volumina of rectangular subsets are given by the products of their side lengths. For higher dimensions, the volume of the initial version space becomes very small. Therefore, the initialization of the extended simplex algorithm yields a classifier whose error is by far smaller than the average generalization error of its spherical counterpart. This effect reflects the *statistical independence* of the data which makes the learning task a lot easier. For dimensions $n > 15$, the initialization of the algorithm alone provides a classifier with almost vanishing average generalization error. As the curves depicted in Figure 10 show the error decreases exponentially.

9. Conclusion

In this contribution we provided exact upper bounds for the generalization performace of binary classifiers. In order to do so, we used an active learning scheme for model selection, and we designed a constructive method which reduces such a bound by successive subdivisions of a version space.

The algorithm was first formulated for the generic case of a binary classification problem, where data lies on a n-dimensional hypersphere and where both classes are separable using $(n-1)$-dimensional great spheres as classifiers. We derived tight upper bounds for the case that the density of data is constant (cf. Proposition 3) as well as for cases, where at least an upper bound of the deviation from the constant density is known (cf. Proposition 9).

We then showed, using the concept of isometries, that abovementioned results are not restricted to hyperspherical data spaces. We showed that if a data space can be mapped onto (a subset of) a hypersphere using an isometry, the constructive active learning method can be applied and Propositions 3 and 9 remain valid and can be used to calculate the bound. In particular, the constructive algorithm can be applied to linear classification in the widely used Euclidean data space $\mathbb{R}^n$, and the corresponding bounds hold. A further extension to binary classification on products of spheres

is straightforward. As a simple example, we considered binary classification on products of circles and proved the exponential decrease of the upper bound for arbitrary densities. Using isometries we showed that this problem can be mapped, for example, onto a binary classification problem in $\mathbb{R}^n$ with axis-parallel hypercube classifiers for which the same exponential decrease holds.

The theoretical results were illustrated using a number of classification tasks using flat as well as non-constant densities, and the derived bounds were compared with the classification error on a test set as a standard method for assessing prediction quality. Since our focus lies on a theoretical analysis of active learning methods (the constructive methods being a vehicle of this analysis), an empirical evaluation and applications of the proposed algorithm to real world problems are of second importance here. Still, a few comments can be made. The computational complexity of the method is $O((n+1)^3)$ where n is the dimension of the hypersphere, hence the method works in practice. Empirically, it also provides good results for non-constant densities. The main current limitation, however, is the restriction of the method to separable classification problems.

Acknowledgments

This work was funded in part by the Deutsche Forschungsgemeinschaft (DFG grant Ob 102/10–2).

Appendix A.

The purpose of this appendix is to give a more detailed analysis of the complexity of the simplex algorithm (see Definition 2) as well as a proof of Proposition 4.

Complexity analysis. We first consider step two. The edge lengths

$$d(v_i, v_j) := \arccos(\langle v_i, v_j \rangle),$$

between vertices v_i, v_j of the simplex must be computed in order to determine which edge is to be cut next. To reduce the number of scalar products that actually need to be evaluated we keep a record of all edge lengths of the current simplex. After step 2, the current simplex is cut by a plane through the midpoint m of some edge (a,b). Assume b gets thrown out. Then all $\frac{n(n-1)}{2}$ edges of the facet opposite to b stay untouched. Further, the length of the new edge (a,m) is one half of the length of (a,b). It is left to compute the lengths of all other edges that contain m. Therefore, we need to compute

$$\frac{n(n+1)}{2} - \frac{n(n-1)}{2} - 1 = n-1$$

scalar products of vectors in $\mathbb{R}^{n+1}$ which gives us an additive term of order $O(n-1)$. In step three, one has to apply an orthonormalization procedure. The modified Gram-Schmidt algorithm (see Meyer, 2000) gives us another summand $O((n+1)^3)$. Since the computational complexity of the other steps is negligible we obtain $O((n+1)^3)$ as a rough complexity estimate for one iteration of the simplex algorithm (see Definition 2). Steps one and seven are performed only once. The initialization by choosing a random orthogonal matrix can be implemented by using an algorithm of Stewart (1980). The complexity of this algorithm is $O(n^2)$ plus the time needed for generating n pseudo-random vectors according to the standard normal distribution. The computation of the center of mass in step seven amounts to adding up all vertex vectors of the current simplex and normalizing their sum to length one. Hence, $O((n+1)^3)$ is a complexity estimate for the final step.

We now restate and prove Proposition 4 from Section 3:

Proposition 12 *Let S be the initial equilateral simplex from the simplex algorithm (see Definition 2). Let $k \in \mathbb{N}$ be the number of steps needed until the maximum of the edge lengths drops. Then*

$$n \leq k \leq \frac{n(n+1)}{2},$$

and these bounds are tight and attainable.

Proof The proof consists of four steps:

1. n is a lower bound: Assume $k \leq n-1$ and do k iterations of the algorithm. Since the degree of each vertex is n, each vertex of the initial simplex is end point of an edge of full length. Thus, if one of these initial vertices is contained in the new subsimplex, the subsimplex contains the adjacent edge of full length, too. The $k \leq n-1$ subdivisions have created at most $n-1$ new vertices. Thus, the new subsimplex contains at least two vertices of the initial simplex. Hence, its maximal edge length is still $\frac{\pi}{2}$.

2. The lower bound is tight: Choose some vertex e. Subdivide all n edges adjacent to e and keep the subsimplex containing the vertex e. All edges starting from e now have length $\frac{\pi}{4}$. The angle enclosed by any two edges at e is $\frac{\pi}{2}$. Now the spherical law of cosines tells us that all edges *not* adjacent to e have length $\frac{\pi}{3}$. This implies that the constructed subsimplex realizes the lower bound.

3. $\frac{n(n+1)}{2}$ is an upper bound: This is clear since $\frac{n(n+1)}{2}$ is the number of edges of the simplex.

4. The upper bound is tight: This is clear for $n = 1$.

The induction step $(n-1) \to n$ goes as follows: Use $\frac{n(n-1)}{2}$ steps to subdivide a facet F of the simplex. Then all edges contained in F are shortened, while the n edges connecting F with the opposite vertex e still have full length. Now subdivide the connecting edges, and always choose the subsimplex which contains e. In this case, e is the only common vertex belonging to the newly subdivided edge and the rest of the edges of full length. Hence, in each of these last n steps, only one edge length is reduced. An illustration of this case is shown in Figure 11. ■

Appendix B.

The purpose of this appendix is to introduce some differential geometric notions used in the main text. For a comprehensive treatise of Riemannian manifolds we refer to Gallot et al. (1990).

A *manifold* M is a generalization of Euclidean space $\mathbb{R}^n$. It is covered by *coordinate charts*, that is, bijective maps $u : U \to \mathbb{R}^n$, where $U \subset M$ is an open subset. The inverse of u is called a *parametrization*. For our work, the most important example of a manifold is the n-sphere $S^n = \{p \in \mathbb{R}^{n+1} \mid \|p\| = 1\}$. It can be covered by two charts, stereographic projection from the north and south pole. Another system of charts is given by the gnomonic projections. Both are discussed in detail in Section 4.

At each point $p \in M$, the manifold is approximated by its *tangent space* T_pM, which generalizes the tangent of a smooth curve. In the case of S^n, the space T_pS^n can be identified with the linear subspace

$$T_pS^n = \{X \in \mathbb{R}^{n+1} \mid \langle p, X \rangle = 0\}.$$

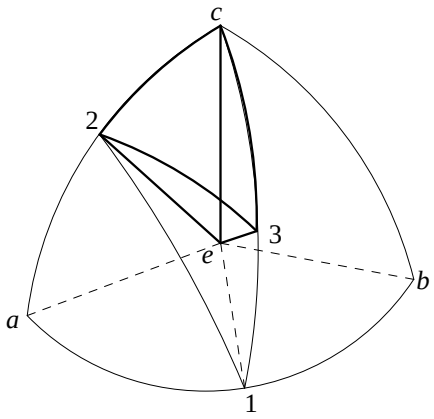

Figure 11: The figure shows a subdivision of a spherical simplex on $S^3 \subset \mathbb{R}^4$ under stereographic projection (see Section 4). The initial simplex, a tetrahedron, is (a,b,c,e). All of its edges have *spherical* length $\frac{\pi}{2}$. After three iterations, indicated by their midpoints $1,2,3$, the subsimplex with edges drawn in bold face still contains three edges (those starting from vertex e) of full length.

A *Riemannian metric* is a choice of a scalar product g_p for each tangent space T_pM. The pair (M,g) is called a *Riemannian manifold*. The canonical Riemannian metric of S^n is given by the restriction of the Euclidean scalar product on $\mathbb{R}^{n+1}$ to the subspace T_pS^n. Given some parametrization $f : \mathbb{R}^n \to U \subset S^n \subset \mathbb{R}^{n+1}$ of a subset U of the sphere, the matrix representation of g is computed by

$$g_{ij} = \langle \frac{\partial f}{\partial x_i}, \frac{\partial f}{\partial x_j} \rangle,$$

where $\langle , \rangle$ denotes the Euclidean scalar product on $\mathbb{R}^{n+1}$. We use this equality in Section 4 to compute the metric in stereographic and gnomonic coordinates.

Let M,N be manifolds of dimensions $\dim M = m$ and $\dim N = n$. Let $p \in M$ be some point in M. A map $f : M \to N$ is called *smooth at* p if there are charts $u : M \supset U \to \mathbb{R}^m$, $v : N \supset V \to \mathbb{R}^n$ with $p \in U$, $f(p) \in V$ such that the composition $\widetilde{f} = v \circ f \circ u^{-1} : \mathbb{R}^m \to \mathbb{R}^n$ is infinitely differentiable in the usual sense. We denote by $df : T_pM \to T_{f(p)}N$ the total differential of f at p. The map f is called *smooth* if it is smooth at all points of M.

A smooth bijective map $f : (M,g) \to (N,h)$ with smooth inverse is called a *diffeomorphism*. If f additionally preserves the metric,

$$g_p(X,Y) = h_{f(p)}(df(X), df(Y)),$$

we call f an *isometry*.

Given a metric g we can measure the length of a curve $\gamma : [a,b] \to M$ by integrating the norm of its tangent vector:

$$L(\gamma) = \int_a^b \|\dot{\gamma}(t)\| \, dt = \int_a^b \sqrt{g_{\gamma(t)}(\dot{\gamma}(t), \dot{\gamma}(t))} dt.$$

The *geodesic distance* $d(p,q)$ of two points $p,q \in M$ is defined to be the infimum of lengths of all curves joining p with q. The minimizing curves are called *geodesics*. In the majority of cases, there

is no explicit formula for $d(p,q)$. Nevertheless, in the case of S^n with its canonical metric it is given by $d(p,q) = \arccos(\langle p,q\rangle)$. Here, the geodesic distance is realized by segments of great circles.

For each Riemannian metric g, there exists a corresponding *Riemannian volume form* ω given in local coordinates $u = (u_1,\ldots,u_m) : M \supset U \to \mathbb{R}^m$ by

$$\omega = \sqrt{\det(g)}du_1 \wedge \ldots \wedge du_n.$$

This can be viewed as a scaled version of the determinant that depends on the base point. Using a coordinate chart u, the volume of a subset A of M is given by

$$\mathrm{Vol}_g(A) = \int_A \omega = \int_{u(A)} \sqrt{\det(g)}du,$$

where the integration on the right hand side is performed in $\mathbb{R}^n$.

References

F.R. Bach. Active learning for misspecified generalized linear models. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 65–72. MIT Press, Cambridge, MA, 2007.

M.-F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 65–72, New York, NY, USA, 2006. ACM Press.

M. Belkin and P. Niyogi. Semi-supervised learning on Riemannian manifolds. *Machine Learning*, 56(1–3):209–239, 2004.

D. J. Best and N. I. Fisher. Efficient simulation of the von Mises distribution. *Applied Statistics*, 28 (2):152–157, 1979.

L. Devroye. *Non-Uniform Random Variate Generation*. Springer-Verlag, 1986.

S. Fine, R. Gilad-Bachrach, and E. Shamir. Learning using query by committee, linear separation and random walks. *Theoretical Computer Science*, 284:25–51, 2002.

Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2–3):133–168, 1997.

S. Gallot, D. Hulin, and J. Lafontaine. *Riemannian Geometry*. Universitext. Springer, 1990.

M. Gromov. *Metric Structures for Riemannian and Non-Riemannian Spaces*. Progress in Mathematics. Birkhäuser, 1999.

R. Herbrich. *Learning Kernel Classifiers–Theory and Algorithms*. Adaptive Computation and Machine Learning. MIT Press, 2002.

G. Lebanon and J. Lafferty. Hyperplane margin classifiers on the multinomial manifold. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*. ACM Press, 2004.

C. Meyer. *Matrix Analysis and Applied Linear Algebra*. SIAM, 2000.

J. Milnor. The Schläfli differential inequality. In *Collected Papers (Volume 1)*. Publish or Perish, Houston, 1994.

H. Q. Minh, P. Niyogi, and Y. Yao. Mercer's theorem, feature maps, and smoothing. In *COLT*, pages 154–168, 2006.

T. M. Mitchell. Generalization as search. *Artificial Intelligence*, 18(2):203–226, 1982.

M. Opper, H. S. Seung, and H. Sompolinsky. Query by commitee. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 287–294, Pittsburgh, Pennsylvania, United States, 1992.

G. W. Stewart. The efficient generation of random orthogonal matrices with an application to condition estimators. *SIAM Journal on Numerical Analysis*, 17:403–409, 1980.

S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2001.

M. K. Warmuth, J. Liao, G. Rätsch, M. Mathieson, and C. Lemmen. Active learning in the drug discovery process. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 1449–1456. MIT Press, 2002.