ARTICLE **Communicated by Chris Rozell**

Learning Data Manifolds with a Cutting Plane Method

SueYeon Chung
schung@fas.harvard.edu
Center for Brain Science, Harvard University, Cambridge, MA 02138, U.S.A.; Edmond and Lily Safra Center for Brain Sciences, Hebrew University of Jerusalem, Jerusalem 9190401, Israel; and Department of Brian and Cognitive Sciences, MIT, Cambridge, MA 02138, U.S.A.

Uri Cohen
uri.co@mail.huji.ac.il
Edmond and Lily Safra Center for Brain Sciences, Hebrew University of Jerusalem, Jerusalem 9190401, Israel

Haim Sompolinsky
haim@fiz.huji.ac.il
Center for Brain Science, Harvard University, Cambridge, MA 02138, U.S.A., and Edmond and Lily Safra Center for Brain Sciences, Hebrew University of Jerusalem, Jerusalem 9190401, Israel

Daniel D. Lee
daniel.d.lee@cornell.edu
Center for Brain Science, Harvard University, Cambridge, MA 02138, U.S.A., and Department of Electrical and Computer Engineering, Cornell Tech, New York, NY 10044, U.S.A.

We consider the problem of classifying data manifolds where each manifold represents invariances that are parameterized by continuous degrees of freedom. Conventional data augmentation methods rely on sampling large numbers of training examples from these manifolds. Instead, we propose an iterative algorithm, M_{CP}, based on a cutting plane approach that efficiently solves a quadratic semi-infinite programming problem to find the maximum margin solution. We provide a proof of convergence as well as a polynomial bound on the number of iterations required for a desired tolerance in the objective function. The efficiency and performance of M_{CP} are demonstrated in high-dimensional simulations and on image manifolds generated from the ImageNet data set. Our results indicate that M_{CP} is able to rapidly learn good classifiers and shows superior generalization performance compared with conventional maximum margin methods using data augmentation methods.

Neural Computation **30, 2593–2615** (2018)
doi:10.1162/neco_a_01119

1 Introduction

Handling object variability is a major challenge for machine learning systems. For example, in visual recognition tasks, changes in pose, lighting, identity, or background can result in large variability in the appearance of objects (Hinton, Dayan, & Revow, 1997). Techniques to deal with this variability have been the focus of much recent work, especially with convolutional neural networks consisting of many layers. The manifold hypothesis states that natural data variability can be modeled as lower-dimensional manifolds embedded in higher-dimensional feature representations (Bengio, Courville, & Vincent, 2013). A deep neural network can then be understood as disentangling or flattening the data manifolds so that they can be more easily read out in the final layer (Brahma, Wu, & She, 2016). Manifold representations of stimuli have also been utilized in neuroscience, where different brain areas are believed to untangle and reformat their representations (Riesenhuber & Poggio, 1999; Serre, Wolf, & Poggio, 2005; Hung, Kreiman, Poggio, & DiCarlo, 2005; DiCarlo & Cox, 2007; Pagan, Urban, Wohl, & Rust, 2013).

This article addresses the problem of classifying data manifolds that contain invariances with a number of continuous degrees of freedom. These invariances may be modeled using prior knowledge, manifold learning algorithms (Tenenbaum, 1998; Roweis & Saul, 2000; Tenenbaum, De Silva, & Langford, 2000; Belkin & Niyogi, 2003; Belkin, Niyogi, & Sindhwani, 2006; Canas, Poggio, & Rosasco, 2012), or generative neural networks via adversarial training (Goodfellow et al., 2014). Based on knowledge of these structures, other work has considered building group-theoretic invariant representations (Anselmi et al., 2013) or constructing invariant metrics (Simard, Le Cun, & Denker, 1994). Most approaches today rely on data augmentation by explicitly generating virtual examples from these manifolds (Niyogi, Girosi, & Poggio, 1998; Schölkopf, Burges, & Vapnik, 1996). Unfortunately, the number of samples needed to successfully learn the underlying manifolds may increase the original training set by more than a thousandfold (Krizhevsky, Sutskever, & Hinton, 2012).

We propose a new method, the manifold cutting plane algorithm (M_{CP}), which uses knowledge of the manifolds to efficiently learn a maximum margin classifier. Figure 1 illustrates the problem in its simplest form, binary classification of manifolds with a linear hyperplane, with extensions to this basic model discussed later. Given a number of manifolds embedded in a feature space, the M_{CP} algorithm learns a weight vector $\vec{w}$ that separates positively labeled manifolds from negatively labeled manifolds with the maximum margin. Although the manifolds consist of uncountable sets of points, the M_{CP} algorithm is able to find a good solution in a provably finite number of iterations and training examples.

Support vector machines (SVM) can learn a maximum margin classifier given a finite set of training examples (Vapnik, 1998); however, with

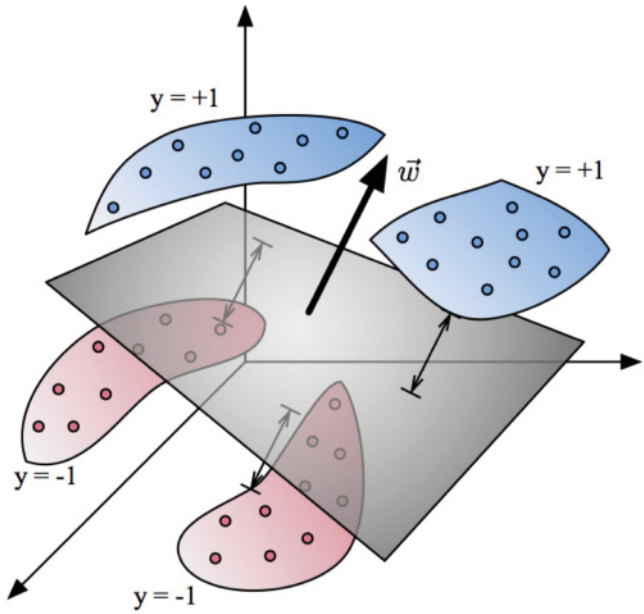

Figure 1: The maximum margin binary classification problem for a set of manifolds. The optimal linear hyperplane is parameterized by the weight vector $\vec{w}$, which separates positively labeled manifolds from negatively labeled manifolds. Conventional data augmentation techniques resort to sampling a large number of points from each manifold to train a classifier.

conventional data augmentation methods, the number of training examples increases exponentially, rendering the standard SVM algorithm intractable. Methods such as shrinkage and chunking to reduce the complexity of SVM have been studied in the context of dealing with large-scale data sets (Smola & Schölkopf, 1998), but the resultant kernel matrix may still be very large. Other methods that subsample the kernel matrix (Lee & Mangasarian, 2001) or reduce the number of training samples (Wang, Neskovic, & Cooper, 2005; Smola & Schölkopf, 1998) are infeasible when the input data come from manifolds with an uncountable set of examples.

Our M_{CP} algorithm directly handles the uncountable set of points in the manifolds by solving a quadratic semi-infinite programming problem (QSIP). M_{CP} is based on a cutting plane method that iteratively refines a finite set of training examples to solve the underlying QSIP (Fang, Lin, & Wu, 2001; Kortanek & No, 1993; Liu, Teo, & Wu, 2004). The cutting plane method was also previously shown to efficiently handle learning problems with a finite number of examples but an exponentially large number of constraints (Joachims, 2006) by adding constraints successively. We provide a

novel analysis of the convergence of M_{CP} with both hard and soft margins. When the problem is realizable, the convergence bound explicitly depends on the margin value, whereas with a soft margin and slack variables, the bound depends linearly on the number of manifolds.

The article is organized as follows. We first consider the hard margin problem and analyze the simplest form of the M_{CP} algorithm. Next, we introduce slack variables in M_{CP}, one for each manifold, and analyze its convergence with additional auxiliary variables. We then demonstrate the application of M_{CP} to both high-dimensional synthetic data manifolds and feature representations of images undergoing a variety of warpings. We compare its performance in both efficiency and generalization error with conventional SVMs using data augmentation techniques. Finally, we discuss some natural extensions and potential future work on M_{CP} and its applications.

2 Manifolds Cutting Plane Algorithm with Hard Margin

In this section, we first consider the problem of classifying a set of manifolds when they are linearly separable. This allows us to introduce the simplest version of the M_{CP} algorithm, along with the appropriate definitions and QSIP formulation. We analyze the convergence of the simple algorithm and prove an upper bound on the number of errors the algorithm can make in this setting.

2.1 Hard-Margin QSIP. Formally, we are given a set of P manifolds $M_p \subset \mathbb{R}^N$, $p = 1, \ldots, P$ with binary labels $y_p = \pm 1$ (all points in the same manifold share the same label). Each manifold M_p is defined by $\vec{x} = f_p(\vec{s}) \in M_p$ where $\vec{s} \in S_p$, S_p is a compact, convex subset of $\mathbb{R}^D$ representing the parameterization of the invariances and $f_p(\vec{s}) : \mathbb{R}^D \to \mathbb{R}^N$ is a continuous function of $\vec{s} \in S_p$ so that the manifolds are bounded: $\forall \vec{s}, \ \|f_p(\vec{s})\| < L$ by some L. In other words, the set of points in the pth manifold is $M_p = \left\{ f_p(\vec{s}) : \ \vec{s} \in S_p \right\} = f_p(S_p)$. We would like to solve the following semi-infinite quadratic programming problem for the weight vector $\vec{w} \in \mathbb{R}^N$:

$$SVM^{simple} : \underset{\vec{w}}{\operatorname{argmin}} \frac{1}{2}\|\vec{w}\|^2 \text{ s.t. } \forall p, \forall \vec{x} \in M_p : \ y_p \langle \vec{w}, \vec{x} \rangle \geq 1. \tag{2.1}$$

This is the primal formulation of the problem, where maximizing the margin $\kappa = \frac{1}{||\vec{w}||}$ is equivalent to minimizing the squared norm $\frac{1}{2}||\vec{w}||^2$. We denote the maximum margin attainable by κ^* and the optimal solution as $\vec{w}^*$. For simplicity, we do not explicitly include the bias term here. A nonzero bias can be modeled by adding a feature of a constant value as a component to all the $\vec{x}$. Note that the dual formulation of this QSIP is more complicated, involving optimization of nonnegative measures over the manifolds. In

order to solve the hard-margin QSIP, we propose the following simple M_{CP} algorithm.

2.2 M_{CP}^{simple} **Algorithm.** The M_{CP}^{simple} algorithm is an iterative algorithm to find the optimal $\vec{w}$ in equation 2.1, based on a cutting-plane method for solving the QSIP. The general idea behind M_{CP}^{simple} is to start with a finite number of training examples, find the maximum margin solution for that training set, augment the training set by finding a point on the manifolds that violates the constraints, and iterating this process until a tolerance criterion is reached.

At each stage k of the algorithm, there is a finite set of training points and associated labels. The training set at the kth iteration is denoted by the set $T_k = \{(\vec{x}_i \in M_{p_i}, y_{p_i})\}$ with $i = 1, \ldots, |T_k|$ examples. For the ith pattern in T_k, p_i is the index of the manifold and y_{p_i} its associated label.

On this set of examples, we solve the following finite quadratic programming problem,

$$SVM_{T_k} : \underset{\vec{w}}{\operatorname{argmin}} \frac{1}{2}\|\vec{w}\|^2 \text{ s.t. } \forall \vec{x}_i \in T_k : \; y_{p_i}\langle \vec{w}, \vec{x}_i\rangle \geq 1, \tag{2.2}$$

to obtain the optimal weights $\vec{w}^{(k)}$ on the training set T_k. We then find a constraint-violating point $\vec{x}_{k+1} \in M_{p_{k+1}}$ from one of the manifolds such that

$$\{p_{k+1}, \; \vec{x}_{k+1} \in M_{p_{k+1}}\} : \quad y_{p+1}\langle \vec{w}^{(k)}, \vec{x}_{p_{k+1}}\rangle < 1 - \delta \tag{2.3}$$

with a required tolerance $\delta > 0$. If there is no such point, the M_{CP}^{simple} algorithm terminates. If such a point exists, it is added to the training set, defining the new set $T_{k+1} = T_k \cup \{(\vec{x}_{k+1}, y_{p_{k+1}})\}$. The algorithm then proceeds at the next iteration to solve $SVM_{T_{k+1}}$ to obtain $\vec{w}^{(k+1)}$. For $k = 1$, the set T_1 is initialized with at least one point from each manifold. The graphic illustration of M_{CP}^{simple} in the simple case of $\delta = 0$, for the kth iteration to $k + 1$th iteration is shown in Figure 2. The pseudocode for M_{CP}^{simple} is shown in algorithm 1.

In step 3 of the M_{CP}^{simple} algorithm, a point among the manifolds that violates the margin constraint needs to be found. The use of a separation oracle is common in other cutting plane algorithms such as those used for structural SVM's (Joachims, 2006) or linear mixed-integer programming (Marchand, Martin, Weismantel, & Wolsey, 2002). In our case, this requires determining the feasibility of $y_p\langle \vec{w}^{(k)}, M_p(\vec{s})\rangle < 1 - \delta$ over the D-dimensional convex parameter set $\vec{s} \in S_p$. When the manifold mapping is convex, feasibility in some cases can be determined analytically. More generally, it can be solved using modern convex techniques such as with projection-based methods (Bauschke & Borwein, 1996). For nonconvex mappings, we note that only the convex hulls of the manifolds, conv (M_p),

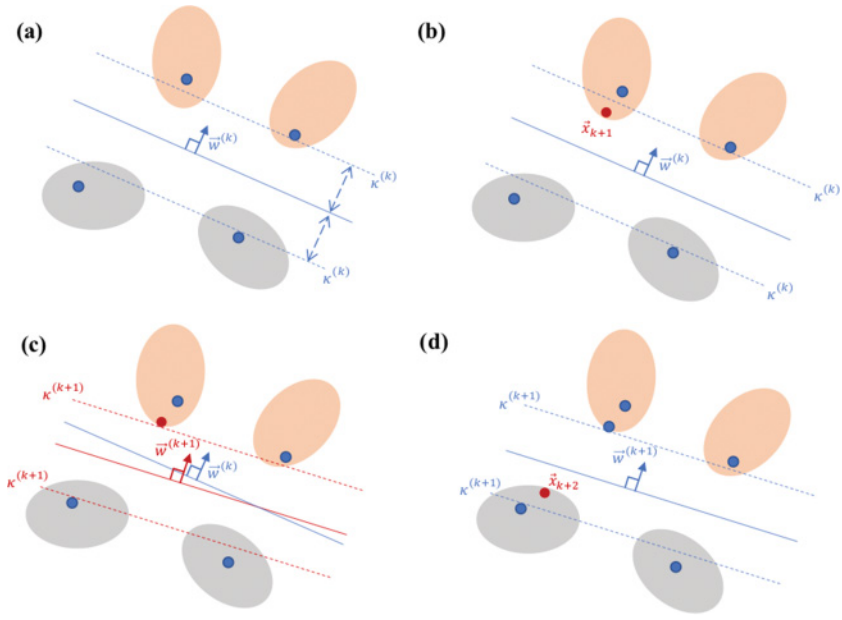

Figure 2: An illustration of M_{CP}^{simple} algorithm for classification of ellipsoids for the simple case of $\delta = 0$. (a) On the kth iteration, optimal $\vec{w}^{(k)}$ with margin $\kappa^{(k)}$ is found with a set of training examples added so far, T_k (blue points). (b) A new point, $\vec{x}_{k+1}$, with a distance to $\vec{w}^{(k)}$ smaller than margin $\kappa^{(k)}$ is found and added to the training set. (c) A new optimal SVM solution $\vec{w}^{(k+1)}$ with margin $\kappa^{(k+1)}$ is found with a set of training examples added so far, T_{k+1}. (d) A new point, $\vec{x}_{k+2}$, with a distance to $\vec{w}^{(k+1)}$ smaller than margin $\kappa^{(k+1)}$ is found and added to the training set. Panels a to d are repeated until the convergence criterion is satisfied.

Algorithm 1: Pseudocode for the M_{CP}^{simple} Algorithm.

Input: δ (tolerance), P manifolds and labels $\{M_p,\ y_p = \pm 1\}$, $p = 1, ..., P$.

1. Initialize $k = 1$, and the set $T_1 = \{(\vec{x}_i \in M_{p_i},\ y_{p_i})\}$ with at least one sample from each manifold M_p.

2. Solve for $\vec{w}^{(k)}$ in SVM_{T_k}: $\min \frac{1}{2} \|\vec{w}\|^2$ s.t. $y^{p_i} \langle \vec{w}, \vec{x}_i \rangle \geq 1$ for all $(\vec{x}_i, y^{p_i}) \in T_k$.

3. Find a point $\vec{x}_{k+1} \in M_{p_{k+1}}$ among the manifolds $\{p_{k+1} = 1, ..., P\}$ with a margin s.t. $y_{p_{k+1}} \left\langle \vec{w}^{(k)}, \vec{x}_{k+1} \right\rangle < 1 - \delta$.

4. If there is no such point, then stop. Else, augment the point set: $T_{k+1} = T_k \cup \left\{ \left(\vec{x}_{k+1}, y_{p_{k+1}} \right) \right\}$.

5. $k \leftarrow k + 1$ and go to 2.

are relevant to the classification problem. Thus, in those situations, the separation oracle need only return a feasible point in the intersection of a manifold convex hull with the half-space determined by $\vec{w}^{(k)}$ and δ. These intersection points can be computed using search techniques in the D-dimensional parameter set using gradient information, finite differences, or via convex relaxation techniques. We describe some specific methods of how separating points can be found in our experiments below.

2.3 Convergence of M_{CP}^{simple}. The M_{CP}^{simple} algorithm will converge asymptotically to an optimal solution when it exists. Here we show that the $\vec{w}^{(k)}$ asymptotically converges to an optimal $\vec{w}^{\star}$. Denote the change in the weight vector in the kth iteration as $\Delta\vec{w}^{(k)} = \vec{w}^{(k+1)} - \vec{w}^{(k)}$. We present a set of lemmas and theorems leading up to the bounds on the number of iterations for convergence and the estimation of the objective function.

Lemma 1. *The change in the weights satisfies* $\langle \Delta\vec{w}^{(k)}, \vec{w}^{(k)} \rangle \geq 0$.

Proof. Define $\vec{w}(\lambda) = \vec{w}^{(k)} + \lambda\Delta\vec{w}^{(k)}$. Then for all $0 \leq \lambda \leq 1$, $\vec{w}(\lambda)$ satisfies the constraints on the point set T_k: $y_{p_i}\langle \vec{w}(\lambda), \vec{x}_i \rangle \geq 1$ for all $\left(\vec{x}_i, y_{p_i}\right) \in T_k$. However, if $\langle \Delta\vec{w}^{(k)}, \vec{w}^{(k)} \rangle < 0$, there exists a $0 < \lambda' < 1$ such that $\|\vec{w}(\lambda')\|^2 < \|\vec{w}^{(k)}\|^2$, contradicting the fact that $\vec{w}^{(k)}$ is a solution to SVM_{T_k}. □

Next, we show that the norm $\|\vec{w}^{(k)}\|^2$ must monotonically increase by a finite amount at each iteration.

Theorem 1. *In the k_{th} iteration of M_{CP}^{simple} algorithm, the increase in the norm of $\vec{w}^{(k)}$ is lower-bounded by $\|\vec{w}^{(k+1)}\|^2 \geq \|\vec{w}^{(k)}\|^2 + \frac{\delta_k^2}{L^2}$, where $\delta_k = 1 - y_{p_{k+1}}\langle \vec{w}^{(k)}, \vec{x}_{k+1} \rangle$ and $\|\vec{x}_{k+1}\| \leq L$.*

Proof. First, note that $\delta_k > \delta \geq 0$; otherwise, the algorithm stops. We have $\|\vec{w}^{(k+1)}\|^2 = \|\vec{w}^{(k)}\|^2 + \|\Delta\vec{w}^{(k)}\|^2 + 2\langle \vec{w}^{(k)}, \Delta\vec{w}^{(k)} \rangle \geq \|\vec{w}^{(k)}\|^2 + \|\Delta\vec{w}^{(k)}\|^2$ (see lemma 1). Consider the point added to set $T_{k+1} = T_k \cup \left(\vec{x}_{k+1}, y_{p_{k+1}}\right)$. At this point, $y_{p_{k+1}}\langle \vec{w}^{(k+1)}, \vec{x}_{k+1} \rangle \geq 1$, $y_{p_{k+1}}\langle w^{(k)}, \vec{x}_{k+1} \rangle = 1 - \delta_k$, hence $y_{p_{k+1}}\langle \Delta\vec{w}^{(k)}, \vec{x}_{k+1} \rangle \geq \delta_k$.

Then, from the Cauchy-Schwartz inequality,

$$\|\Delta\vec{w}^{(k)}\|^2 \geq \frac{\delta_k^2}{\|\vec{x}_{k+1}\|^2} > \frac{\delta_k^2}{L^2} > \frac{\delta^2}{L^2}. \tag{2.4}$$

Since the solution $\vec{w}^{\star}$ satisfies the constraints for T_k, $\|\vec{w}^{(k)}\| \leq \frac{1}{\kappa^*}$. Thus, the sequence of iterations monotonically increases norms and is upper-bounded by $\frac{1}{\kappa^{\star}}$. Due to convexity, there is a single global optimum, and the M_{CP}^{simple} algorithm is guaranteed to converge. □

As a corollary, we see that this procedure is guaranteed to find a realizable solution if it exists in a finite number of steps.

Corollary 1. *The M_{CP}^{simple} algorithm converges to a zero error classifier in fewer than $\frac{L^2}{(\kappa^\star)^2}$ iterations, where $\kappa^\star$ is the optimal margin and L bounds the norm of the points on the manifolds.*

Proof. When there is an error, we have $\delta_k > 1$, and $\|\vec{w}^{(k+1)}\|^2 \geq \|\vec{w}^{(k)}\|^2 + \frac{1}{L^2}$ (see equation 2.4). This implies the total number of possible errors is upper-bounded by $\frac{L^2}{(\kappa^\star)^2}$. □

With a finite tolerance $\delta > 0$, we obtain a bound on the number of iterations required for convergence:

Corollary 2. *The M_{CP}^{simple} algorithm for a given tolerance $\delta > 0$ terminates in fewer than $\frac{L^2}{(\kappa^\star \delta)^2}$ iterations where $\kappa^\star$ is the optimal margin and L bounds the norm of the points on the manifolds.*

Proof. Again, $\|\vec{w}^k\|^2 \leq \|\vec{w}^\star\|^2 = \frac{1}{(\kappa^\star)^2}$, and each iteration increases the squared norm by at least $\frac{\delta^2}{L^2}$. □

We can also bracket the error in the objective function after M_{CP}^{simple} terminates:

Corollary 3. *With tolerance $\delta > 0$, after M_{CP}^{simple} terminates with solution $\vec{w}_{M_{CP}}$, the optimal value $\|\vec{w}^\star\|$ of SVM^{simple} is bracketed by $\|\vec{w}_{M_{CP}}\|^2 \leq \|\vec{w}^\star\|^2 \leq \frac{1}{(1-\delta)^2}\|\vec{w}_{M_{CP}}\|^2$.*

Proof. The lower bound on $\|\vec{w}^\star\|^2$ is as before. Since M_{CP}^{simple} has terminated, setting $\vec{w}' = \frac{1}{(1-\delta)}\vec{w}_{M_{CP}}$ would make $\vec{w}'$ feasible for SVM^{simple}, resulting in the upper bound on $\|\vec{w}^\star\|^2$. □

3 M_{CP} with Slack Variables

In many classification problems, the manifolds may not be linearly separable due to their dimensionality, size, or correlation structure. In these situations, M_{CP}^{simple} will not be able to find a feasible solution, and there is no solution that has zero error over all the manifold points. To handle these problems, the classic approach is to introduce slack variables on each point (SVM^{slack}) to control generalization error. For continuous manifolds, we define generalization performance as the classification error over distributions of points sampled from the manifolds M_p with corresponding

labeled outputs. The naive implementation of slack variables requires defining appropriate measures over entire manifolds and solving the optimization problem over infinite sets of slack variables. This is not computationally tractable, and we formulate a more efficient alternative version of QSIP with slack variables next.

3.1 QSIP with Manifold Slacks. In this work, we propose using only one slack variable per manifold for classification problems with nonseparable manifolds. This formulation demands that all the points on each manifold $\vec{x} \in M_p$ obey an inequality constraint with one manifold slack variable, $y_p\langle \vec{w}, \vec{x}\rangle + \xi_p \geq 1$. As we will see, solving for this constraint is tractable, and the algorithm has good convergence guarantees.

However, a single slack requirement for each manifold by itself may not be sufficient for good generalization performance. In particular, with only these constraints, the resulting solution can potentially misclassify entire manifolds. Our empirical studies show that generalization performance can be improved if we also demand that some representative points $\vec{x}_p \in M_p$ on each manifold also obey the margin constraint, $y_p\langle \vec{w}, \vec{x}_p\rangle \geq 1$, so that these representative points are correctly classified. In this work, we implement this intuition by specifying appropriate center points $\vec{x}_p^c$ for each manifold M_p. The center point can be the center of mass of the manifold, the exemplar used to generate the manifolds (Krizhevsky et al., 2012), or a particular representative point. In this article, we assume that it is possible for the center points to strictly obey the margin inequalities associated with their manifold labels. Otherwise, we could potentially introduce additional slack variables and corresponding regularization parameters for these center points in the optimization. Formally, the QSIP optimization problem is summarized below, where the objective function is minimized over the weight vector $\vec{w} \in \mathbb{R}^N$ and slack variables $\vec{\xi} \in \mathbb{R}^P$:

$$SVM^{slack}_{manifold} : \underset{\vec{w},\vec{\xi}}{\operatorname{argmin}} F(\vec{w}, \vec{\xi}) = \frac{1}{2}\|\vec{w}\|^2 + C\sum_{p=1}^{P} \xi_p,$$

$$\text{s.t. } \forall p, \forall \vec{x} \in M_p : \; y_p\langle \vec{w}, \vec{x}\rangle + \xi_p \geq 1 (\text{manifolds}),$$

$$\forall p : \; y_p\langle \vec{w}, \vec{x}_p^c\rangle \geq 1 \text{ (centers)}, \xi_p \geq 0$$

3.2 M^{slack}_{CP} Algorithm. With these definitions, we introduce our M^{slack}_{CP} algorithm (see algorithm 2) with slack variables. The proposed M^{slack}_{CP} algorithm modifies the cutting plane approach to solve a semi-infinite, semidefinite quadratic program. Each iteration involves a finite set, $T_k = \{(\vec{x}_i \in M_{p_i}, y_{p_i})\}$ with $i = 1, \ldots, |T_k|$ examples, that is used to define the following soft margin SVM:

Algorithm 2: Pseudocode for the M_{CP}^{slack} Algorithm.

Input: δ (tolerance), P manifolds and labels $\{M_p, y_p = \pm 1\}$, and centers $\vec{x}_p^c$

1. Initialize $k = 1$, and the set $T_1 = \{(\vec{x}_i \in M_{p_i},\ y_{p_i})\}$ with at least one sample from each manifold M_p.

2. Solve for $\vec{w}^{(k)}$,$\vec{\xi}^{(k)}$: $\min \frac{1}{2}\|\vec{w}\|^2 + C\sum_{p=1}^{P}\xi_p$ s.t. $y_{p_\mu}\langle \vec{w},\vec{x}^\mu\rangle + \xi_{p_\mu} \geq 1$ for all $\left(\vec{x}^\mu, y_{p_\mu}\right) \in T_k$ and $y_p\langle \vec{w},\vec{x}_p^c\rangle \geq 1$ for all p.

3. Find a point $\vec{x}_{k+1} \in M_{p_{k+1}}$ among the manifolds $\{p = 1, ..., P\}$ with slack violation larger than the tolerance δ: $y_{p_{k+1}}\left\langle \vec{w}^{(k)}, \vec{x}_{k+1}\right\rangle + \xi_{p_{k+1}} < 1 - \delta$

4. If there is no such point, then stop. Else, augment the point set: $T_{k+1} = T_k \cup \left\{\left(\vec{x}_{k+1}, y_{p_{k+1}}\right)\right\}$.

5. $k \leftarrow k + 1$, and go to 2.

$$SVM_{T_k}^{slack}: \underset{\vec{w},\vec{\xi}}{\operatorname{argmin}} \frac{1}{2}\|\vec{w}\|^2 + C\sum_{p=1}^{P}\xi_p$$

$$\text{s.t. } \forall(\vec{x}_i, y_{p_i}) \in T_k:\ y_{p_i}\langle \vec{w}, \vec{x}_i\rangle + \xi_{p_i} \geq 1;\ \forall p:\ y_p\langle \vec{w}, \vec{x}_p^c\rangle \geq 1\ (\textit{centers}),\ \xi_p \geq 0$$

to obtain the weights $\vec{w}^{(k)}$ and slacks $\vec{\xi}^{(k)}$ at each iteration. We then find a point $\vec{x}_{k+1} \in M_{p_{k+1}}$ from one of the manifolds so that

$$y_{p_{k+1}}\langle \vec{w}^{(k)}, \vec{x}_{k+1}\rangle + \xi_{p_{k+1}}^{(k)} = 1 - \delta_k, \tag{3.1}$$

where $\delta_k > \delta$. If there is no such a point, the M_{CP}^{slack} algorithm terminates. Otherwise, the point $\vec{x}_{k+1}$ is added as a training example to the set $T_{k+1} = T_k \cup \{(\vec{x}_{k+1}, y_{p_{k+1}})\}$, and the algorithm proceeds to solve $SVM_{T_{k+1}}^{slack}$ to obtain $\vec{w}^{(k+1)}$ and $\vec{\xi}^{(k+1)}$.

3.3 Convergence of M_{CP}^{slack}. Here we show that the objective function $F\left(\vec{w}, \vec{\xi}\right) = \frac{1}{2}\|\vec{w}\|^2 + C\sum_{p=1}^{P}\xi_p$ is guaranteed to increase by a finite amount with each iteration. This result is similar to Tsochantaridis, Joachims, Hofmann, and Altun (2005), but here we present proofs in the primal domain over an infinite number of examples.

Lemma 2. *The changes in the weights and slacks satisfy*

$$\langle \Delta\vec{w}^{(k)}, \vec{w}^{(k)}\rangle + C\sum_{p} \Delta\vec{\xi}_p^{(k)} \geq 0, \tag{3.2}$$

where $\Delta\vec{w}^{(k)} = \vec{w}^{(k+1)} - \vec{w}^{(k)}$ *and* $\Delta\vec{\xi}^{(k)} = \vec{\xi}^{(k+1)} - \vec{\xi}^{(k)}$.

Proof. Define $\vec{w}(\lambda) = \vec{w}^{(k)} + \lambda\Delta\vec{w}^{(k)}$ and $\vec{\xi}(\lambda) = \vec{\xi}^{(k)} + \lambda\Delta\vec{\xi}^{(k)}$. Then for all $0 \leq \lambda \leq 1$, $\vec{w}(\lambda)$ and $\vec{\xi}(\lambda)$ satisfy the constraints for $SVM^{slack}_{T_k}$. The resulting change in the objective function is given by

$$F\big(\vec{w}(\lambda), \vec{\xi}(\lambda)\big) - F\big(\vec{w}^{(k)}, \vec{\xi}^{(k)}\big) = \lambda\left[\langle\Delta\vec{w}^{(k)}, \vec{w}^{(k)}\rangle + C\sum_p \Delta\xi_p^{(k)}\right] + \frac{1}{2}\lambda^2\|\Delta\vec{w}^{(k)}\|^2. \tag{3.3}$$

If equation 3.2 is not satisfied, there is some $0 < \lambda' < 1$ such that $F\big(\vec{w}(\lambda'), \vec{\xi}(\lambda')\big) < F\big(\vec{w}^{(k)}, \vec{\xi}^{(k)}\big)$, which contradicts the fact that $\vec{w}^{(k)}$ and $\vec{\xi}^{(k)}$ are a solution to SVM_{T_k}. □

We derive that the added point at each iteration must be a support vector for the next weight:

Lemma 3. *In each iteration of M^{slack}_{CP} algorithm, the added point $\big(\vec{x}_{k+1}, y_{p_{k+1}}\big)$ must be a support vector for the new weights and slacks, s.t.:*

$$y_{p_{k+1}}\langle\vec{w}^{(k+1)}, \vec{x}_{k+1}\rangle + \xi^{(k+1)}_{p_{k+1}} = 1, \tag{3.4}$$

$$y_{p_{k+1}}\langle\Delta\vec{w}^{(k)}, \vec{x}_{k+1}\rangle + \Delta\xi^{(k)}_{p_{k+1}} = \delta_k. \tag{3.5}$$

Proof. Suppose $y_{p_{k+1}}\langle\vec{w}^{(k+1)}, \vec{x}_{k+1}\rangle + \xi^{(k+1)}_{p_{k+1}} = 1 + \epsilon$ for some $\epsilon > 0$. Then we can choose $\lambda' = \frac{\delta_k}{\delta_k + \epsilon} < 1$ so that $\vec{w}(\lambda') = \vec{w}^{(k)} + \lambda'\Delta\vec{w}^{(k)}$ and $\vec{\xi}(\lambda') = \vec{\xi}^{(k)} + \lambda'\Delta\vec{\xi}^{(k)}$ satisfy the constraints for $SVM^{slack}_{T_{k+1}}$. But from lemma 2, we have $F\big(\vec{w}(\lambda'), \vec{\xi}(\lambda')\big) < F\big(\vec{w}^{(k+1)}, \vec{\xi}^{(k+1)}\big)$, which contradicts the fact that $\vec{w}^{(k+1)}$ and $\vec{\xi}^{(k+1)}$ are a solution to $SVM_{T_{k+1}}$. Thus, $\epsilon = 0$, and the point $\big(\vec{x}_{k+1}, y_{p_{k+1}}\big)$ must be a support vector for $SVM_{T_{k+1}}$. Equation 3.5 results from subtracting equation 3.1 from 3.4. □

We also derive a bound on the following quadratic function over non-negative values:

Lemma 4. *Given $K > 0$, $\delta > 0$, $\forall x \geq 0$*

$$\frac{1}{2}(x - \delta)^2 + Kx \geq min\left(\frac{1}{2}\delta^2, \frac{1}{2}K\delta\right). \tag{3.6}$$

Proof. The minimum value occurs when $x^\star = [\delta - K]_+$. When $K \geq \delta$, then $x^\star = 0$, and the minimum is $\frac{1}{2}\delta^2$. When $K < \delta$, the minimum occurs at $K\big(\delta - \frac{1}{2}K\big) \geq \frac{1}{2}K\delta$. Thus, the lower bound is the smaller of these two values. □

Using the lemmas above, the lower bound on the change in the objective function can be found:

Theorem 2. *In each iteration k of the M^{slack}_{CP} algorithm, the increase in the objective function for $SVM^{slack}_{manifold}$, defined as $\Delta F^{(k)} = F\left(\vec{w}^{(k+1)}, \vec{\xi}^{(k+1)}\right) - F\left(\vec{w}^{(k)}, \vec{\xi}^{(k)}\right)$, is lower-bounded by*

$$\Delta F^{(k)} \geq min\left(\frac{1}{8}\frac{\delta_k^2}{L^2},\ \frac{1}{2}C\delta_k\right). \tag{3.7}$$

Proof. We first note that the change in objective function is strictly increasing:

$$F\left(\vec{w}^{(k+1)}, \vec{\xi}^{(k+1)}\right) - F\left(\vec{w}^{(k)}, \vec{\xi}^{(k)}\right) = \left[\langle \Delta\vec{w}^{(k)}, \vec{w}^{(k)}\rangle + C\sum_p \Delta\xi_p^{(k)}\right] + \frac{1}{2}\|\Delta\vec{w}^{(k)}\|^2 > 0. \tag{3.8}$$

This can be seen immediately from lemma 2 when $\Delta\vec{w}^{(k)} \neq 0$. On the other hand, if $\Delta\vec{w}^{(k)} = 0$, we know that $\Delta\xi_{p_k}^{(k)} = \delta_k$ from lemma 3 and $\Delta\xi_{p\neq p_k}^{(k)} = 0$ since $\vec{\xi}^{(k)}$ is the solution for SVM_{T_k}. So for $\Delta\vec{w}^{(k)} = 0$, $F\left(\vec{w}^{(k+1)}, \vec{\xi}^{(k+1)}\right) - F\left(\vec{w}^{(k)}, \vec{\xi}^{(k)}\right) = C\delta_k$. To compute a general lower bound on the increase in the objective function, we proceed as follows.

The added point $\vec{x}_k$ comes from a particular manifold M_{p_k}. If $\Delta\xi_{p_k}^{(k)} \leq 0$, from lemma 3, we have $y_{p_k}\langle\Delta\vec{w}^{(k)}, \vec{x}_k\rangle \geq \delta_k$. Then by the Cauchy-Schwarz inequality, $\|\Delta\vec{w}^{(k)}\|^2 \geq \frac{\delta_k^2}{L^2}$, which yields $F\left(\vec{w}^{(k+1)}, \vec{\xi}^{(k+1)}\right) - F\left(\vec{w}^{(k)}, \vec{\xi}^{(k)}\right) \geq \frac{1}{2}\frac{\delta_k^2}{L^2}$.

We next analyze $\Delta\xi_{p_k}^{(k)} > 0$ and consider the finite set of points $\left(\vec{x}_\nu,\ p_k\right) \in T_k$ that come from the p_k manifold. There must be at least one such point in T_k by initialization of the algorithm. Each of these points obeys the constraints:

$$y_{p_k}\langle\vec{w}^{(k)}, \vec{x}_\nu\rangle + \xi_{p_k}^{(k)} = 1 + \epsilon_\nu^{(k)}, \tag{3.9}$$

$$y_{p_k}\langle\vec{w}^{(k+1)}, \vec{x}_\nu\rangle + \xi_{p_k}^{(k+1)} = 1 + \epsilon_\nu^{(k+1)}, \tag{3.10}$$

$$\epsilon_\nu^{(k)},\ \epsilon_\nu^{(k+1)} \geq 0. \tag{3.11}$$

We consider the minimum value of the thresholds: $\eta = \min_\nu \epsilon_\nu^{(k)}$. We have two possibilities: η is positive so that none of the points are support vectors for $SVM^{slack}_{T_k}$, or $\eta = 0$ so that at least one support vector lies in M_{p_k}.

Case $\eta > 0$. In this case, we define a linear set of slack variables:

$$\xi_p(\lambda) = \begin{cases} \xi_p^{(k)} + \lambda\Delta\xi_p^{(k)} & p \neq p_k \\ \xi_{p_k}^{(k)} & p = p_k \end{cases} \tag{3.12}$$

and weights $\vec{w}(\lambda) = \vec{w}^{(k)} + \lambda\Delta\vec{w}^{(k)}$. Then for $0 \le \lambda \le \min\left(\frac{\eta_{p_k}}{\Delta\xi_{p_k}^{(k)}}, 1\right)$, $\vec{w}(\lambda)$ and $\vec{\xi}(\lambda)$ will satisfy the constraints for SVM_{T_k}. Following similar reasoning in lemma 2, this implies

$$\langle \Delta\vec{w}^{(k)}, \vec{w}^{(k)} \rangle + C \sum_{p \neq p_k} \Delta\vec{\xi}_p^{(k)} \ge 0. \tag{3.13}$$

Then in this case, we have

$$F\left(\vec{w}^{(k+1)}, \vec{\xi}^{(k+1)}\right) - F\left(\vec{w}^{(k)}, \vec{\xi}^{(k)}\right) \tag{3.14}$$

$$= \left[\langle \Delta\vec{w}^{(k)}, \vec{w}^{(k)} \rangle + C \sum_{p} \Delta\xi_p^{(k)} \right] + \frac{1}{2} \|\Delta\vec{w}^{(k)}\|^2 \tag{3.15}$$

$$\ge C\Delta\xi_{p_k}^{(k)} + \frac{1}{2} \|\Delta\vec{w}^{(k)}\|^2 \tag{3.16}$$

$$\ge \frac{1}{2} \frac{\left(\delta_k - \Delta\xi_{p_k}^{(k)}\right)^2}{L^2} + C\Delta\xi_{p_k}^{(k)} \tag{3.17}$$

$$\ge \min\left(\frac{1}{2L^2}\delta_k^2, \ \frac{1}{2}C\delta_k \right) \tag{3.18}$$

by applying equation 3.13 in 3.16, lemma 3 and Cauchy-Schwarz in equation 3.17, and lemma 4 in equation 3.18.

Case $\eta = 0$. In this case, we consider $\varepsilon = \min_{\epsilon_v^{(k)}=0} \epsilon_v^{(k+1)} \ge 0$, that is, the minimal increase among the support vectors. We then define

$$\xi_p(\lambda) = \begin{cases} \xi_p^{(k)} + \lambda\Delta\xi_p^{(k)} & p \neq p_k \\ \xi_{p_k}^{(k)} + \lambda\left(\Delta\xi_p^{(k)} - \varepsilon\right) & p = p_k \end{cases} \tag{3.19}$$

and weights $\vec{w}(\lambda) = \vec{w}^{(k)} + \lambda\Delta\vec{w}^{(k)}$. There will then be a finite range of $0 \le \lambda \le \lambda_{\min}$ for which $\vec{\xi}(\lambda)$ and $\vec{w}(\lambda)$ satisfy the constraints for SVM_{T_k} so that

$$\langle \Delta\vec{w}^{(k)}, \vec{w}^{(k)} \rangle + C \sum_{p \neq p_k} \Delta\vec{\xi}_p^{(k)} + C\left(\Delta\xi_{p_k}^{(k)} - \varepsilon\right) \ge 0 \tag{3.20}$$

$$\langle \Delta\vec{w}^{(k)}, \vec{w}^{(k)} \rangle + C \sum_{p} \Delta\vec{\xi}_p^{(k)} \ge C\varepsilon \tag{3.21}$$

We also have a support vector $\left(\vec{x}_v, \ p_k\right) \in T_k$ so that

$$y_{p_k} \langle \vec{w}^{(k+1)}, \vec{x}_v \rangle + \xi_{p_k}^{(k+1)} = 1 + \varepsilon, \tag{3.22}$$

$$y_{p_k}\langle \Delta\vec{w}^{(k)}, \vec{x}_v\rangle + \Delta\xi_{p_k}^{(k)} = \varepsilon. \tag{3.23}$$

Using lemma 3 and Cauchy-Schwarz, we get

$$y_{p_k}\langle \Delta\vec{w}^{(k)}, \vec{x}_k - \vec{x}^v\rangle = \delta_k - \varepsilon \tag{3.24}$$

$$\|\Delta\vec{w}^{(k)}\|^2 \geq \frac{1}{4L^2}\big(\delta_k - \varepsilon\big)^2. \tag{3.25}$$

Thus, we have

$$F\big(\vec{w}^{(k+1)}, \vec{\xi}^{(k+1)}\big) - F\big(\vec{w}^{(k)}, \vec{\xi}^{(k)}\big) \tag{3.26}$$

$$= \left[\langle \Delta\vec{w}^{(k)}, \vec{w}^{(k)}\rangle + C\sum_p \Delta\xi_p^{(k)}\right] + \frac{1}{2}\|\Delta\vec{w}^{(k)}\|^2 \tag{3.27}$$

$$\geq C\varepsilon + \frac{1}{8L^2}\big(\delta_k - \varepsilon\big)^2 \tag{3.28}$$

$$\geq \min\left(\frac{1}{8L^2}\delta_k^2,\ \frac{1}{2}C\delta_k\right) \tag{3.29}$$

by applying equations 3.21 and 3.25 to obtain the final bound. □

Since the M_{CP}^{slack} algorithm is guaranteed to increase the objective by a finite amount, it will terminate in a finite number of iterations if we require $\delta_k > \delta$ for some positive $\delta > 0$.

Corollary 4. *The M_{CP}^{slack} algorithm for a given $\delta > 0$ will terminate after at most $P \cdot max\big(\frac{8CL^2}{\delta^2}, \frac{2}{\delta}\big)$ iterations, where P is the number of manifolds and L bounds the norm of the points on the manifolds.*

Proof. $\vec{w} = 0$ and $\xi_p = 1$ is a feasible solution for $SVM_{manifold}^{slack}$. Therefore, the optimal objective function is upper-bounded by $F\big(\vec{w} = 0, \vec{\xi} = 1\big) = PC$. The upper bound on the number of iterations is then provided by theorem 2. □

We can also bound the error in the objective function after M_{CP}^{slack} terminates:

Corollary 5. *With $\delta > 0$, after M_{CP}^{slack} terminates with solution $\vec{w}_{M_{CP}}$, slack $\vec{\xi}_{M_{CP}}$, and value $F_{M_{CP}} = F\big(\vec{w}_{M_{CP}}, \vec{\xi}_{M_{CP}}\big)$, the optimal value $F^\star$ of $SVM_{manifold}^{slack}$ is bracketed by*

$$F_{M_{CP}} \leq F^\star \leq F_{M_{CP}} + PC\delta. \tag{3.30}$$

Proof. The lower bound on $F^\star$ is apparent since $SVM_{manifold}^{slack}$ includes $SVM_{T_k}^{slack}$ constraints for all k. Setting the slacks $\xi_p = \xi_{M_{CP},p} + \delta$ will make the solution feasible for $SVM_{manifold}^{slack}$ resulting in the upper bound. □

4 Experiments

4.1 L_q Ellipsoidal Manifolds. As an illustration of our method, we have generated D-dimensional L_q-norm ellipsoids with random radii, centers, and directions. The points on each manifold M_p are parameterized as $M_p = \{\vec{x} \,|\, \vec{x} = \vec{x}_p^c + \sum_{i=1}^{D} R_i s_i \vec{u}_i^p \in \mathbb{R}^N\}$, where the center $\vec{x}_p^c$ and basis vectors $\vec{u}_i^p$ are random gaussian and R_i, the ellipsoidal radii, are sampled from Unif$[0.5R_0,\ 1.5R_0]$ with mean R_0. Similar to the classification task shown in Figure 1, we consider the following task: given P of D-dimensional ellipsoids with the above radii distributions embedded in N dimension, where $\frac{1}{2}P$ of the ellipsoids are labeled positive and the rest negative, find a linear hyperplane $\vec{w}$ that shatters the given ellipsoids according to the labels, with maximum margin.

We compare the performance of M_{CP} to the conventional Point SVM (SVM^{simple}, SVM^{slack}) with training samples drawn uniformly from the surface of the L_q ellipsoids. The test example points for M_{CP} and point SVM are also drawn from the uniform distribution over the surface of each L_q ellipsoid and given the same label as the binary label assigned to each ellipsoid. Performance is measured by generalization error on the task of classifying new test points from positively labeled manifolds from negatively labeled ones as a function of the total number of training samples used during learning.

For these manifolds, the separation oracle of M_{CP} returns points that minimize $y^p\big[\vec{w} \cdot \big(\vec{x}_p^c + \sum_{i=1}^{D} R_i s_i \vec{u}_i^p\big)\big]$ over the set $\|\vec{s}\|_q \le 1$. For norms with $q \ge 1$, the solution can be expressed as $s_i = -\frac{(h_i^p)^{1/(q-1)}}{\{\sum (h_i^p)^{q/(q-1)}\}^{1/q}}$ where $h_i^p = y^p \vec{w} \cdot \vec{u}_i^{\,p}$. For M_{CP}^{slack}, we used an additional single margin constraint per manifold given by the center of the ellipsoids $\vec{x}_p^c$.

We compare the generalization performance of the M_{CP} algorithm and point-wise SVM on the L_2 ellipsoids (see Figure 3) and L_{50} ellipsoids (see Figure 4). As a linearly separable example, we use $P = 10$ L_2 ellipsoids embedded in N dimension, where with the parameters we use, $D = 40$ and $R_0 = 20$, the classification problem is known to be below critical manifold capacity for $N = 500$ and $P = 10$ and therefore linearly separable according to the recent theory of linear classification of general manifolds (Chung, Lee, & Sompolinsky, 2018). Then we test the generalization error performance of M_{CP}^{simple} and SVM^{simple} for different numbers of samples used for training (see Figure 3a). To illustrate a linearly nonseparable example, we use the same set of L_2 ellipsoids ($D = 40$ and $R_0 = 20$), except with an increased number of L_2 ellipsoids, $P = 12$, and a decreased ambient dimension $N = 475$, such that the task is above the critical manifold capacity and linearly nonseparable. Then we compare the generalization error performance of M_{CP}^{slack} simulations with point-SVM^{slack} (Figure 3b). Similarly, for L_{50} ellipsoids, N, P, D were chosen such that the problem is slightly below the capacity for a

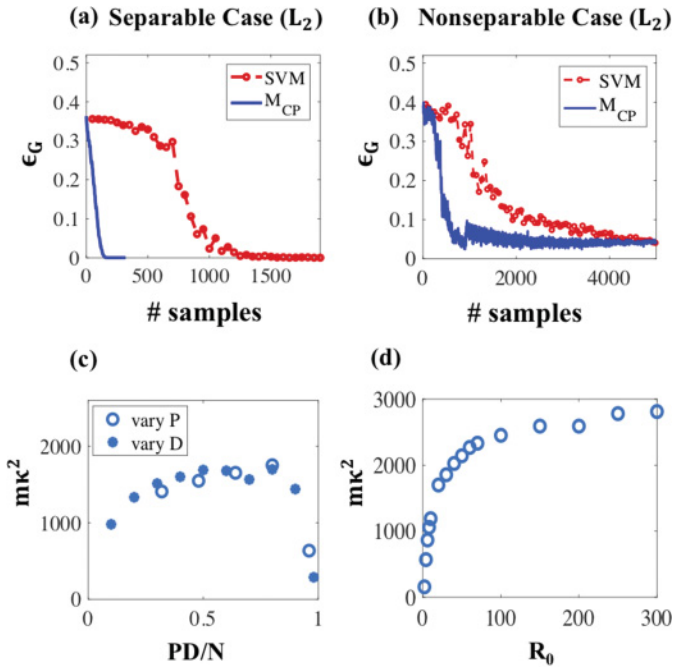

Figure 3: M_{CP} solution for L_2 ellipsoids. (a, b) Generalization error (ϵ_G) shown as a function of the total number of training samples (solid blue line) compared with conventional point SVM (dashed line). $D = 40$, $R_0 = 20$ (mean elliptic radii), and (a) $N = 500$, $P = 10$, just below the critical capacity is used for M_{CP}^{simple} and (b) $N = 475$, $P = 12$ for M_{CP}^{slack} with slack coefficient $C_{opt} = 1$. ϵ_G is computed from 500 points per manifold. (c, d) The effect of manifold parameters on task complexity, illustrated by $m\kappa^2$, where m is the number of samples required to reach 0 error solution in the separable regime and κ is the hard margin of the problem. (c) $m\kappa^2$ as a function of PD/N where P is varied, while $N = 500$, $R_0 = 20$, $D = 40$ (circles), and where D is varied while $N = 500$, $R_0 = 20$, $P = 10$ (x). (d) $m\kappa^2$ as a function of R_0 while $N = 500$, $P = 10$, $D = 40$.

separable case and above the capacity for a nonseparable case (the details are in the captions of Figure 4). The results illustrate that M_{CP} achieves a low generalization error very efficiently compared to a conventional maximum margin classifier using sampled training examples.

Another important question is how the classification task difficulty interacts with the number of samples required for convergence of M_{CP}, as well as the manifold geometries. The recent theory of manifold classification (Chung, Lee, & Sompolinsky, 2016, 2018) illustrates the relationship between the critical manifold linear classification capacity $\alpha = P/N$, the task

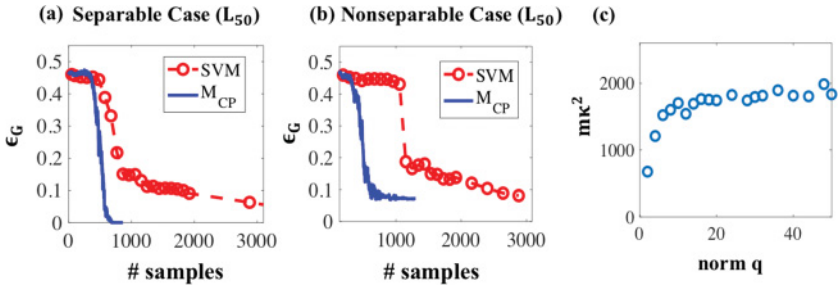

Figure 4: M_{CP} solution for L_q ellipsoids. (a, b) Generalization error (ϵ_G) of L_{50} ellipsoids shown as a function of the total number of training samples (solid blue line) compared with conventional point SVM (dashed line). $P = 48$, $D = 10$, $R_0 = 20$ (mean elliptic radii) and (a) $P/N = 0.096$ ($N = 500$) just below the critical capacity is used for M_{CP}^{simple} and (b) $P/N = 0.098$ ($N = 490$) just above the critical capacity is used for M_{CP}^{slack} with slack coefficient $C_{opt} = 100$. ϵ_G is computed from 500 points per manifold. (c) The effect of manifold parameters on task complexity, illustrated by $m\kappa^2$, where m is the number of samples required to reach 0 error solution in the separable regime and κ is the hard margin of the problem $m\kappa^2$ as a function of q, which determines the norm of the L_q ellipsoids, while $N = 500$, $R = 5$, $D = 40$, $P = 10$.

margin κ, and the manifold geometries such as dimension and size. To see the effect of the manifold geometries, we tested the dependence between the L_2 ellipsoid properties above, P,D, R_0, q (for L_q), and the number of required training examples m and the task margin κ within the linearly separable regime. First, we varied the number of the ellipsoids P, while all the other task parameters are fixed at $N = 500$, $D = 40$, $R_0 = 20$ (see Figure 3c), and varied the dimension of the ellipsoids D, while all the other task parameters are fixed at $N = 500$, $P = 10$, $R_0 = 20$ (see Figure 3c). Then we find a regime where there is a trade-off between the number of examples required for convergence, m, and the task margin, κ. The task margin κ is related to the task difficulty, because the harder the classification task is, the smaller the margin κ is. When we vary P and D, the trade-off between m and κ is manifested by an approximate plateau in $\sqrt{m\kappa^2}$, except when $PD/N \sim 1$ (close to the capacity where $\kappa \to 0$). The task difficulty also depends on the size of the manifolds, and when we increase the ellipsoid scale, R_0, $\sqrt{m\kappa^2}$ shows very similar behavior as the increase in P or D, showing a rapid increase when the task is easy and finding an approximate plateau for the larger R_0, except that even in the large R_0 regime, there is rapid drop in $\sqrt{m\kappa}$ because even in the $R_0 \to \infty$, the hyperplane can orthogonalize the solution and have a nonzero asymptotic margin (see Figure 3d). The variation in the norm q for L_q ellipsoids shows a qualitatively similar behavior as the variation in R_0, reflecting the fact that

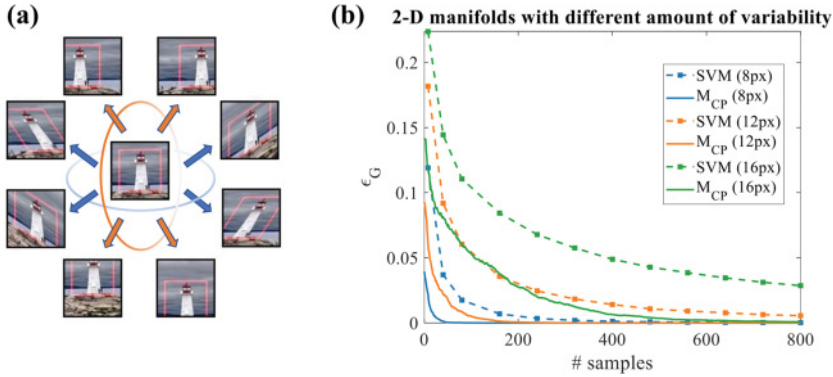

Figure 5: Image-based object manifolds. (a) Basic affine transformation: a template image (middle) with an object-bounding box (pink) surrounded by changes along four axes defined by basic affine transformation—horizontal and vertical translation, horizontal and vertical shear—all with maximal displacement of 16 pixels (px). Those represent the "corners" of the object manifold, which includes all combinations of such transformations with limits on the resulting object displacement. (b) Generalization error of the M_{CP} solution (with hard margins) for image-based object manifolds as a function of the number of training samples (solid line) compared with that of conventional point SVM (with hard margins; dashed lines and squares) for manifolds with different amounts of variation (color coded). Results obtained for $2-D$ object manifolds (pixel representations with horizontal and vertical translation) and generalization error are averaged over different choices for labels.

even when $q \to \infty$, the hyperplane can find an orthogonalizing solution and have a nonzero, asymptotic margin κ (see Figure 4c).

4.2 ImageNet Object Manifolds. We also apply the M_{CP} algorithm to a more realistic class of object manifolds. Here, each object manifold is constructed from a set of affine warpings applied to a template image from the ImageNet data set (Deng et al., 2009). Figure 5a illustrates image changes along multiple axes of variation in such a manifold, which contains all combinations of changes along multiple axes. Those manifolds are parameterized by the intrinsic transformation dimensionality (i.e., the number of axes of variation) and the generated image variability (defined as the maximal displacement of the object corners, in pixels). In general, those affine transformations have 6 degrees of freedom; here, we demonstrate results for $2-D$ manifolds, utilizing only translation transformations, and $4-D$ manifolds, utilizing both translation and shear transformations as in Figure 5a. The resulting P manifolds are split into dichotomies, with $\frac{1}{2}P$ manifolds considered as positive instances and the remaining $\frac{1}{2}P$ manifolds as negative

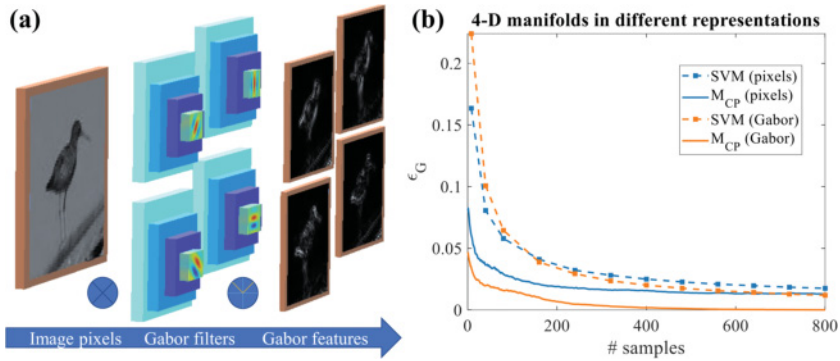

Figure 6: Change in representation of object manifolds. (a) Illustration of the transformation from pixels to Gabor representation: a grayscale image of an object (left) is processed by applying an array of Gabor functions with four orientations and four scales (middle), followed by full-wave rectification and results with different feature maps (right). (b) Generalization error of the M_{CP} solution (with slack variables) for image-based object manifolds as a function of the number of training samples (solid line) compared with that of conventional point SVM (with slack variables; dashed lines and squares) for the pixel and Gabor representations (color coded). Results obtained for $4 - D$ object manifolds (object translation and shear transformations) and generalization error are averaged over different choices for labels.

instances. M_{CP} is then used to learn a binary classifier for the manifolds, and we obtain the parameter C through cross-validation. We note that M_{CP} can easily be extended to multiclass problems, but only binary classification problems were used in this experiment.

We trained on $P = 8$ object manifolds and used $N = 500$ features throughout. At each iteration of M_{CP}, a constraint-violating point on the manifolds is found using a local search among $K = 5$ neighboring samples in the affine warping space. Figure 5b shows how quickly M_{CP} converges to a low generalization error when classifying manifolds with different amounts of variation.

Next, we compared the generalization performance of M_{CP} to conventional SVM on sampled points from two different feature representations of the images: in the original pixel space (as in the previous experiment) and in a V1-like representation of the same images created by applying full-wave rectification after filtering by arrays of Gabor functions (Serre, Wolf, Bileschi, Riesenhuber, & Poggio, 2007), as illustrated in Figure 6a. The test examples are created by transforming the template image with parameters drawn uniformly in the intrinsic transformation-parameter space and given the same label as the binary label assigned to each object manifold.

For pixel representation, the images' grayscale value is used, and Gabor representations are created by applying a set of Gabor filters (of four orientations and four spatial scales) around multiple image locations, from which we randomly sampled features to match the number of pixels.

Figure 6b shows the results of M_{CP} on those object manifolds, showing how as manifolds are transformed from pixel to Gabor features, the problem changes from nonseparable to separable. We note that the maximal number P of manifolds that are separable depends on the feature representation. Thus, M_{CP} can also be used to investigate the benefits of alternative features representations, such as those found in different areas of the visual hierarchy in the brain or in the layers of a deep neural network.

5 Discussion

We described and analyzed a novel algorithm, M_{CP}, based on the cutting plane method for finding the maximum margin solution for classifying manifolds. We proved the convergence of M_{CP} and provided bounds on the number of iterations required. Our analysis shows that in the separable case, the algorithm finds a solution in a finite number of iterations that completely segregates the manifolds even though they consist of uncountable sets of points.

The situation is more complex when the manifolds are not separable. In that case, M_{CP} finds an approximate solution in a finite number of iterations that obey slack constraints on a per manifold basis. We consider the generalization performance of the M_{CP} solution on points randomly sampled from the manifolds and find that the solution generalizes well across all the manifold input points even though it has only been provided with a finite set of samples during the training iterations. The number of slack variables that are nonzero indicates the fraction of manifolds containing inseparable points, providing an upper bound on the generalization error.

M_{CP} is reminiscent of cutting-plane methods for structured SVMs in that its convergence does not explicitly depend on the presence of a large number of constraints. However, for M_{CP}, the semi-infinate nature of the optimization problem arises even for binary classification due to the continuous manifold structure of the inputs, not from constraints on the outputs as it does for structured SVMs. A possible future extension of M_{CP} would be to handle structured output labels on manifold inputs.

Our experiments with both synthetic data manifolds and image manifolds demonstrate the efficiency of M_{CP} and superior performance in terms of generalization error compared to conventional SVMs using many virtual examples. Although the theoretical convergence bounds depend only on the margin, our numerical examples show that the empirical number of training samples required for M_{CP} depends on both the task margin and the manifold geometric properties. In particular, we have demonstrated how

the algorithm performs when the dimensionalities, sizes, and shapes of the manifolds are varied. This illustrates the complex interplay in learning dynamics when the underlying manifold geometries are considered.

There is a natural extension of M_{CP} to nonlinear classifiers via the kernel trick, as all our operations involve dot products between the weight vector $\vec{w}$ and manifold points $M_p(\vec{s})$. At each iteration, the algorithm would solve the dual version of the SVM_{T_k} problem, which is readily kernelized. More theoretical work is needed to analyze infinite-dimensional kernels when the manifold optimization problem no longer is a QSIP but becomes a fully (doubly) infinite quadratic programming problem.

Beyond binary classification, variations of M_{CP} can also be used to solve other machine learning problems, including multiclass classification, ranking, ordinal regression, and one-class learning. We can also use M_{CP} to evaluate the computational benefits of manifold representations at successive layers of deep networks in both machine learning and brain sensory hierarchies. We anticipate using M_{CP} to help construct novel hierarchical architectures that can incrementally reformat the manifold representations through the layers for better overall performance in machine learning tasks, improving our understanding of how neural architectures can learn to process high-dimensional real-world signal ensembles and cope with a large variability due to continuous modulation of the underlying physical parameters.

Acknowledgments

The work is partially supported by the Gatsby Charitable Foundation, the Swartz Foundation, the Simons Foundation (SCGB grant 325207), the NIH, the MAFAT Center for Deep Learning, and the Human Frontier Science Program (project RGP0015/2013). D. L. also acknowledges the support of the U.S. National Science Foundation, Army Research Laboratory, Office of Naval Research, Air Force Office of Scientific Research, and Department of Transportation.

References

Anselmi, F., Leibo, J. Z., Rosasco, L., Mutch, J., Tacchetti, A., & Poggio, T. (2013). *Unsupervised learning of invariant representations in hierarchical architectures*. arXiv:1311.4158.

Bauschke, H. H., & Borwein, J. M. (1996). On projection algorithms for solving convex feasibility problems. *SIAM Review*, *38*(3), 367–426.

Belkin, M., & Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, *15*(6), 1373–1396.

Belkin, M., Niyogi, P., & Sindhwani, V. (2006). Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, *7*, 2399–2434.

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1798–1828.

Brahma, P. P., Wu, D., & She, Y. (2016). Why deep learning works: A manifold disentanglement perspective. *IEEE Transactions on Neural Networks and Learning Systems*, *27*(10), 1997–2008.

Canas, G., Poggio, T., & Rosasco, L. (2012). Learning manifolds with k-means and k-flats. In F. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems, 25* (pp. 2465–2473). Red Hook, NY: Curran.

Chung, S., Lee, D. D., & Sompolinsky, H. (2016). Linear readout of object manifolds. *Physical Review E*, *93*(6), 060301.

Chung, S., Lee, D. D., & Sompolinsky, H. (2018). Classification and geometry of general perceptual manifolds. *Physical Review X*, 8:031003.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 248–255). Piscataway, NJ: IEEE.

DiCarlo, J. J., & Cox, D. D. (2007). Untangling invariant object recognition. *Trends in Cognitive Sciences*, *11*(8), 333–341.

Fang, S.-C., Lin, C.-J., & Wu, S.-Y. (2001). Solving quadratic semi-infinite programming problems by using relaxed cutting-plane scheme. *Journal of Computational and Applied Mathematics*, *129*(1), 89–104.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., . . . Bengio, Y. (2014). Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems*, *27* (pp. 2672–2680). Red Hook, NY: Curran.

Hinton, G. E., Dayan, P., & Revow, M. (1997). Modeling the manifolds of images of handwritten digits. *IEEE Transactions on Neural Networks*, *8*(1), 65–74.

Hung, C. P., Kreiman, G., Poggio, T., & DiCarlo, J. J. (2005). Fast readout of object identity from macaque inferior temporal cortex. *Science*, *310*(5749), 863–866.

Joachims, T. (2006). Training linear SVMS in linear time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 217–226). New York: ACM.

Kortanek, K. O., & No, H. (1993). A central cutting plane algorithm for convex semi-infinite programming problems. *SIAM Journal on Optimization*, *3*(4), 901–918.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems*, *25* (pp. 1097–1105). Red Hook, NY: Curran.

Lee, Y.-J., & Mangasarian, O. L. (2001). RSVM: Reduced support vector machines. In *Proceedings of the 2001 SIAM International Conference on Data Mining* (pp. 1–17). Philadelphia: SIAM.

Liu, Y., Teo, K. L., & Wu, S.-Y. (2004). A new quadratic semi-infinite programming algorithm based on dual parameterization. *Journal of Global Optimization*, *29*(4), 401–413.

Marchand, H., Martin, A., Weismantel, R., & Wolsey, L. (2002). Cutting planes in integer and mixed integer programming. *Discrete Applied Mathematics*, *123*(1), 397–446.

Niyogi, P., Girosi, F., & Poggio, T. (1998). Incorporating prior information in machine learning by creating virtual examples. *Proceedings of the IEEE*, *86*(11), 2196–2209.

Pagan, M., Urban, L. S., Wohl, M. P., & Rust, N. C. (2013). Signals in inferotemporal and perirhinal cortex suggest an untangling of visual target information. *Nature Neuroscience*, *16*(8), 1132–1139.

Riesenhuber, M., & Poggio, T. (1999). Hierarchical models of object recognition in cortex. *Nature Neuroscience*, *2*(11), 1019–1025.

Roweis, S. T., & Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, *290*(5500), 2323–2326.

Schölkopf, B., Burges, C., & Vapnik, V. (1996). Incorporating invariances in support vector learning machines. In *Proceedings of the International Conference on Artificial Neural Networks* (pp. 47–52). New York: Springer.

Serre, T., Wolf, L., Bileschi, S., Riesenhuber, M., & Poggio, T. (2007). Robust object recognition with cortex-like mechanisms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(3), 411–426.

Serre, T., Wolf, L., & Poggio, T. (2005). Object recognition with features inspired by visual cortex. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (vol. 2, pp. 994–1000). Piscataway, NJ: IEEE.

Simard, P. Y., Le Cun, Y., & Denker, J. S. (1994). Memory-based character recognition using a transformation invariant metric. In *Proceedings of the 12th IAPR International Conference on Computer Vision and Image Processing* (vol. 2, pp. 262–267). Piscataway, NJ: IEEE.

Smola, A. J., & Schölkopf, B. (1998). *Learning with kernels*. Citeseer.

Tenenbaum, J. B., De Silva, V., & Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, *290*(5500), 2319–2323.

Tenenbaum, J. B. (1998). Mapping a manifold of perceptual observations. In M. I. Jordan, M. J. Kearns, & S. A. Solla (Eds.), *Advances in neural information processing systems, 10* (pp. 682–688). Cambridge, MA: MIT Press.

Tsochantaridis, I., Joachims, T., Hofmann, T., & Altun, Y. (2005). Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, *6*, 1453–1484.

Vapnik, V. (1998). *Statistical learning theory*. New York: Wiley.

Wang, J., Neskovic, P., & Cooper, L. N. (2005). Training data selection for support vector machines. In *Proceedings of the International Conference on Natural Computation* (pp. 554–564). New York: Springer.

Received August 23, 2017; accepted February 13, 2018.