Semi-supervised Learning Using an Unsupervised Atlas*

Nikolaos Pitelis, Chris Russell, and Lourdes Agapito

University College London, London, United Kingdom
{n.pitelis,c.russell,l.agapito}@cs.ucl.ac.uk

Abstract. In many machine learning problems, high-dimensional datasets often lie on or near manifolds of locally low-rank. This knowledge can be exploited to avoid the "curse of dimensionality" when learning a classifier. Explicit manifold learning formulations such as LLE are rarely used for this purpose, and instead classifiers may make use of methods such as local coordinate coding or auto-encoders to implicitly characterise the manifold.

We propose novel manifold-based kernels for semi-supervised and supervised learning. We show how smooth classifiers can be learnt from existing descriptions of manifolds that characterise the manifold as a set of piecewise affine charts, or an atlas. We experimentally validate the importance of this smoothness vs. the more natural piecewise smooth classifiers, and we show a significant improvement over competing methods on standard datasets. In the semi-supervised learning setting our experiments show how using unlabelled data to learn the detailed shape of the underlying manifold substantially improves the accuracy of a classifier trained on limited labelled data.

1 Introduction

A fundamental challenge of machine learning lies in finding embeddings in high dimensional spaces that capture meaningful measures of distance. Bellman [3] coined the term curse of dimensionality to describe the problems that arise as the volume of the space grows exponentially with the number of dimensions and this in turn necessitates an exponentially larger number of observations to cover the space. However, in most applications, data is not uniformly distributed over the whole space, but instead lies on a locally low-dimensional topological structure. This key geometric intuition drives the use of manifolds in machine learning. By finding a compact representation which preserves the relevant topological structure of the data, manifold learning techniques avoid many of the statistical and computational difficulties that arise from high-dimensionality and provide meaningful low-dimensional representations.

In this work, we primarily target semi-supervised learning. We show how unsupervised knowledge of the data manifold can be exploited to learn Support

* This research was funded by the European Research Council under the ERC Starting Grant agreement 204871-HUMANIS.

T. Calders et al. (Eds.): ECML PKDD 2014, Part II, LNCS 8725, pp. 565–580, 2014.

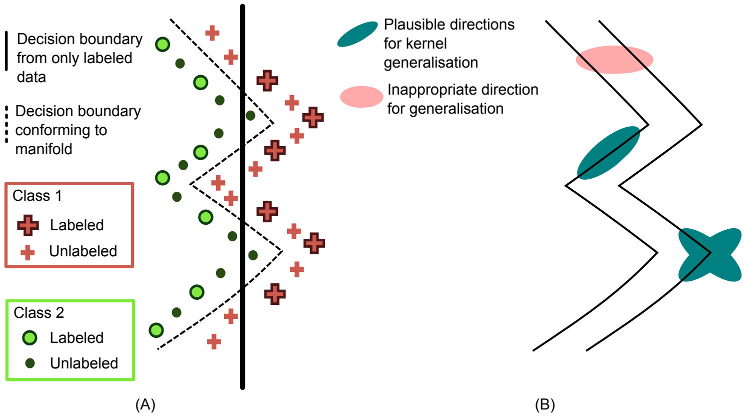

Decision boundary from only labeled data
Decision boundary conforming to manifold
Class 1
Labeled
Unlabeled
Class 2
Labeled
Unlabeled
Plausible directions for kernel generalisation
Inappropriate direction for generalisation
(A)
(B)

Fig. 1. Desirable properties of a learning algorithm, with respect to a manifold. *(A)* Knowledge of the underlying manifold structure of data can improve classification accuracy. Here unlabelled data can be used to discover the underlying shape of the manifold improving classification accuracy. *(B)* Given a manifold (**black**) an ideal classifier should generalise strongly in directions tangent to the manifold **(blue)** and generalise poorly with respect to directions orthogonal to the manifold **(pink, top)**.

Vector Machine (SVM) kernels [24] that work in a set of low-dimensional charts associated with the manifold, avoiding the curse of dimensionality and exhibiting good generalisation to unseen data. Our formulation of manifold kernels is based on the mathematical definition of a manifold as an atlas [14]. Although our implementation makes use of the recent manifold learning technique of [18], it does not rely on the particular atlas using this method. In principle, given a soft-cost function for associating points with charts, it can be applied to any atlas either known a priori or discovered using a manifold learning technique that characterises the manifold as a set of parameterised charts that can be used to back-project points, such as [19].

Although manifold learning has shown much promise in finding embeddings that capture the intrinsic local low dimensionality of data, in practice the majority of such approaches have difficulty with the presence of noise and are unable to characterise closed manifolds such as the surface of a ball. [18] showed how any manifold, either closed or otherwise, could be approximated by an atlas of piecewise affine charts, and experimentally demonstrated their method's robustness to noise. Unfortunately, a good approximation of a smooth manifold as a piecewise affine manifold may require the use of a large number of charts (cf. figure 2). In addition, as a path on the manifold discontinuously jumps from the co-ordinate system of one chart to another, the use of many charts limits the generality of classifiers that can be learned from raw chart co-ordinates and encourages over-fitting.

In response to these difficulties, we present a new class of chart-based Mercer kernels suitable for use with SVMs that smoothly vary in the transition from one

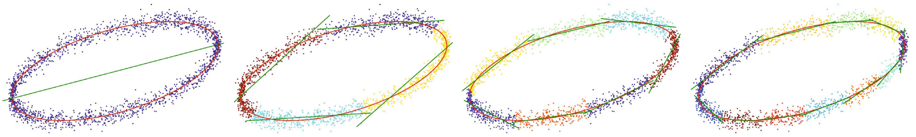


Fig. 2. Approximations of a closed manifold of varying coarseness found using [18] with different minimum description length priors. The underlying manifold is shown in red, and local affine charts approximating the manifold are in green. The coloured dots shown are points sampled with Gaussian noise from the manifold, and their colour indicates which chart they belong to. As the approximation is refined the local affine approximations come closer to tangent planes of the manifold. However, more charts are required for these tighter approximations, and as such, classifiers trained directly on the raw charts that approximate the manifold well (e.g. rightmost) will have poor generalisation. To avoid this trade-off between generalisation, and a good local characterisation of the manifold, section 4 proposes kernels which smoothly vary along a path transitioning from one chart to another.

chart to another[1]. We experimentally verify our formulation and show that it outperforms a variety of existing methods including standard SVMs using: Linear and RBF kernels; standard manifold unwrapping followed by nearest neighbour (NN) and SVM classification using techniques such as [9, 18, 21, 30]; various forms of local co-ordinate coding (LCC) [13, 25, 32]; multi-class kernel-based classifiers [4, 8, 23]; and RBF kernels on the raw chart co-ordinates. We also present asymptotic speed-ups for our kernel computation, and show how a sparse approximation of it can be typically calculated in $O(n\sqrt{n})$ rather than the more usual $O(n^2)$ associated with Mercer kernels.

In using manifold learning as a preprocessing step before classification, we are conforming to the three tenets of manifold learning set out in [19]. Namely:

1. **The semi-supervised learning hypothesis:** The distribution of unlabelled data is informative and should be used to guide supervised classifiers.
2. **The unsupervised manifold learning hypothesis:** High-dimensional datasets often lie near locally low-rank manifolds.
3. **The manifold learning hypothesis for classification:** Data from different classes typically lies in different regions of the manifold and are often separated from one another by regions containing few samples.

Taken together these tenets give us an intuitive picture of supervised learning shown in figure 1. Note that the *strong generalisation* of a classifier in a particular direction, simply means that we expect the classifier response to vary slowly as we move in that direction, while *weak generalisation* refers to the fact that the classifier response may fluctuate quickly in that direction. As an SVM trained classifier is simply a weighted sum of kernel responses, such generalisation in a classifier can be encouraged by making the kernel responses behave in this manner.

[1] See [16] for an extensive discussion of the relationship between smoothness and the generalisation of classifiers.

As a complete method ours is a two-step approach:

1. **Unsupervised learning of the underlying manifold:** We approximate the manifold of data on the original space by fitting an atlas of low-dimensional overlapping affine charts.
2. **Supervised training of an SVM:** We propose a new family of Mercer Kernels for SVM-based supervised learning that make use of a soft assignment of datapoints to the underlying low-dimensional affine charts to generating the kernels.

Our contribution. As with earlier approaches that fuse manifold learning with supervised classification [2, 22], our two-stage approach has a natural application in semi-supervised learning. Unlabelled data can be used to generate a more detailed description of the manifold, which can then improve the trained classifier (see figure 1). In the experimental section, we provide an extensive evaluation that shows how these unsupervised manifolds can be used to substantially improve the generalisation of classifiers trained from limited data, where we outperform three other competing approaches: Eigenfunction [10], PVM [28], and AnchorGraphReg [15].

A further contribution of our work lies in the transition from learning on a single chart, found with a standard method like LLE, to learning on multiple charts. Most manifolds, such as the surface of a ball, cannot be expressed as a single chart without either folding or tearing the manifold. Learning kernels on manifolds that cannot be expressed as a single chart is currently a topic of interest. For instance [11] extended kernel-based algorithms to the Riemannian manifold of Symmetric Positive Definite (SPD) matrices. However, while they restricted both the type of manifold (SPD matrices), and the types of kernel considered, our work shows how any kernel defined over a local Euclidean space can be transformed into a kernel over any atlas.

2 Prior Work

While preprocessing a dataset with explicit manifold learning techniques such as [21, 29, 30], that explicitly find a single global mapping of the data lying in a high dimensional $\mathbb{R}^D$ to a lower dimensional $\mathbb{R}^d$, is an obvious way of avoiding the curse of dimensionality, with the exception of [30], such approaches have seen little use in practice. As argued by [18], this may well be because finding a single global mapping by aligning patches that capture local information, is an unnecessarily hard problem that should be avoided wherever possible. Such mappings are unable to capture the intrinsic structure of closed manifolds such as the surface of a ball, and as such methods typically try to preserve various metric properties of the local neighbourhood, they are vulnerable to noise, and a misestimation of the local neighbourhood can propagate throughout the manifold leading to degenerate solutions.

As an alternative to unwrapping a manifold, there has been much interest in local co-ordinate systems to characterise low dimensional subspaces. [19] made use of a variant of auto-encoders to characterise a manifold as a set of charts that were then fine tuned to improve classification accuracy. Local co-ordinate coding [27] and the quadratic variant local tangent based coding [26] approximate non-linear functions by interpolating between anchor points assumed to lie on a low-dimensional manifold.

The works [13] and [32] learnt a linear SVM over a set of full rank linear co-ordinates that smoothly vary from one cluster centre to another. While inspired by local coordinate coding, neither [13] nor [32] make the same manifold assumptions. Instead, they explicitly make use of a weighted concatenation of coordinate systems each of which spans the entire space, rather than focusing on local low-dimensional subspaces as in manifold learning.

Our work differs from previous approaches that fuse manifold learning with SVMs [2, 17, 22] both in the types of manifold that can be expressed –the previous approaches are based on Laplacian eigenmaps [1] that have difficulty with closed manifolds– and in the form taken. These previous methods alter their regularisation to penalise changes in classification response on the manifold, while we reshape kernels to generalise more in the direction of the manifold. As such our different approaches can be seen as complimentary descriptions of manifold constraints.

3 Learning a Manifold as an Atlas

The recent work of [18] formulated manifold learning as a problem of finding an atlas $\mathcal{A}$, defined as a set of overlapping charts $\mathcal{A} = \{c_1, c_2, \dots c_n\}$, over points $\mathcal{X}$, such that each chart corresponds to an affine subspace of the original space $\mathbb{R}^D$ that accurately describe the local structure. This parametrization of the local transforms as affine spaces allows the efficient use of PCA to find local embeddings, without restricting the overall expressiveness of the atlas. Manifold learning is then formulated as a hybrid continuous/discrete optimisation that simultaneously estimates the affine mappings of charts and solving for a discrete labelling problem, that governs the assignment of points to charts. This objective takes the form of the minimisation of the cost:

$$C(\mathbf{z}) = \sum_{x \in \mathcal{X}} \left(\sum_{i \in \mathbf{z}_x} E_i(x) \right) + \lambda \mathrm{MDL}(\mathbf{z}), \tag{1}$$

Where $\mathbf{z}_x$ is the set of charts associated with point x, and E_i is defined as in (4). Both subproblems, assigning points to charts and choosing the affine mappings, minimise the same cost –the reconstruction error associated with mapping points from their location in a chart back into the embedding space– subject to the spatial constraint that every point must belong to the *interior of one chart* – that is that each point and all its neighbours in a k-NNgraph should belong to the same chart[2]. Sparse solutions are encouraged by adding a *minimum description length* (MDL) prior[12] term to the energy that penalises the total number of

[2] Note that some points belong to more than one chart.

active charts used in an assignment. In practice, Atlas[18] is initialised by an excess of chart proposals in the form of random affine subspaces and alternates between assigning points to charts using the graph-cut [6] based optimisation of [20] and refitting the chart subspaces with PCA. Figure 2 illustrates the approximation of a closed manifold with an Atlas of locally affine subspaces using different MDL priors.

This manifold learning technique offers a set of attractive properties that we take advantage of in our chart-based approach to learning with SVMs. First, since the set of charts that characterise the atlas overlap, points may belong to more than one chart. Therefore, overlapping charts must explain some of the same data in the areas of overlap that connect neighbouring subspaces which results in implicit smoothness in the transition from one subspace to another. Furthermore, this method allows us to learn closed manifolds since it finds charts corresponding to affine subspaces on the original space $\mathbb{R}^D$ and does not require unwrapping into a lower dimensional space. In addition, this method is intrinsically adaptive in that the size of the region assigned to each chart is selected automatically in response to the amount of noise, the curvature of the manifold, and the sparsity of the data.

3.1 Formulation

More formally, each chart c_i contains a subset of points $X_i \subseteq \mathcal{X}$. We use $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_X\}$ to describe the labelling, where $\mathbf{z}_x$ refers to the assignment of charts to point x (the set of charts that point x belongs to).

We define the d-dimensional affine subspace associated with each chart c_i in terms of its mean μ_i, and an orthonormal matrix $\mathbf{C}_i$ which describes its principal directions of variance. Using x to refer to a datapoint in a feature space $\mathbb{R}^D$, we use $P_i^\perp(x) : \mathbb{R}^D \to \mathbb{R}^d$ to refer to a projection from the original feature space into a low rank subspace defined by chart c_i of the form:

$$P_i^\perp(x) = \mathbf{C}_i(x - \mu_i), \tag{2}$$

where μ_i is an offset corresponding to the mean of a subset of points used to define chart c_i, and $\mathbf{C}_i$ is the orthonormal matrix composed of the top d eigenvectors of the covariance matrix of the points X_i that belong to the chart, that projects from the embedding space into chart c_i.

We refer to the back-projection of point x into a low rank subspace of the original space as $P_i(x) : \mathbb{R}^d \to \mathbb{R}^D$

$$P_i(x) = \mathbf{C}_i^T P_i^\perp(x) + \mu_i, \tag{3}$$

and define the reconstruction error for point x belonging to chart c_i as the squared distance between a point and the back-projection of the closest vector on the chart c_i

$$E_i(x) = ||x - P_i(x)||_2^2. \tag{4}$$

4 Chart-Based Kernels

4.1 Definition

We associate with each chart c_i a unique Mercer kernel K^i defined over the projected space $\mathbb{R}^d$, and we define each element $K_{x,y}$ of the square kernel matrix K as:

$$K_{x,y} = \sum_{c_i \in \mathcal{A}} \exp\left(-\frac{E_i(x) + E_i(y)}{\gamma^2}\right) K^i_{x,y}, \tag{5}$$

where $E_i(x)$ is defined as in (4). This kernel can be understood as a natural softening of the obvious hard-assignment kernel H

$$H_{x,y} = \sum_{c_i \in \mathcal{A}} \varDelta(x \in c_i)\varDelta(y \in c_i) K^i_{x,y}, \tag{6}$$

where $\varDelta(\cdot)$ is the indicator function that takes value 1 if $\cdot$ is true and 0 otherwise. This says that the inner product between two points is the same as a standard kernel defined over chart c_i if both points belong to c_i, and 0 otherwise.

In practice we consider two forms of local kernels K^i. Local linear kernels of the form

$$K^i_{x,y} = P_i^{\perp}(x) \cdot P_i^{\perp}(y) \tag{7}$$

and local Radial Basis Function (RBF) kernels of the form

$$K^i_{x,y} = \exp\left(-\frac{||P_i^{\perp}(x) - P_i^{\perp}(y)||_2^2}{\sigma^2}\right). \tag{8}$$

In the experimental section we compare against the hard-assignment kernel H and show the importance of our softening of the kernel response.

4.2 All such Kernels are Mercer

The proof follows directly by construction. We make use of two equivalent definitions of Mercer kernels. Namely: a kernel matrix is Mercer if and only if *(i)* it can be defined as a matrix of inner products over a Hilbert space; and equivalently a kernel is Mercer if and only if *(ii)* it is a positive semi-definite matrix.

We initially consider one of the kernels K^i. It follows from *(i)* that there must be some mapping $\phi_i(\cdot)$ from $\mathbb{R}^d$ to a Hilbert space such that

$$K^i_{x,y} = \langle \phi_i(x), \phi_i(y) \rangle. \tag{9}$$

We define

$$w^i_x = \exp\left(-\frac{||x - P_i(x)||_2^2}{\gamma^2}\right) \tag{10}$$

and linearly rescale the elements of the Hilbert space $\phi(x)$, by their weights w^i_x to induce a new kernel matrix $\bar{K}^i$

$$\bar{K}^i_{x,y} = \langle w^i_x \phi_i(x), w^i_y \phi_i(y) \rangle = w^i_x w^i_y K^i_{x,y}. \tag{11}$$

By *(i)* this kernel is Mercer. It follows that it is positive semi-definite, and consequently, the sum of all weighted kernels

$$K = \sum_{c_i \in \mathcal{A}} \bar{K}^i \tag{12}$$

is also positive semi-definite and therefore Mercer.□

While the weights w were chosen by analogy with the definition of a radial basis function, the proof holds for all choices of weight, and all choices of kernel. In the experimental section, we evaluate manifold variants of linear, local quadratic, and RBF kernels.

4.3 Efficient Approximation of the Kernel

In practice the majority of weights, w_x^i are close to zero and both w_y^i and $K_{x,y}^i$ are small[3]. As such, if w_x^i is small then the entire row $K_{x,\_}^i$ and column $K_{\_,x}^i$ can be safely set to 0, without altering the classification accuracy. To take advantage of this, we only compute explicitly the inner products $\langle \phi_i(x), \phi_i(y) \rangle$ if c_i is one of the closest subspaces for both x and y. This is equivalent to setting $w_x^i = 0$ if c_i is not one of the closest subspaces to x, and so the kernel remains Mercer.

We assume that the parameters of Atlas are chosen in such a way that for n datapoints Atlas will find $O(\sqrt{n})$ charts, each containing less than $O(\sqrt{n})$ points[4]. Then for each point, we must compute the distance to every subspace - which takes time $O(n\sqrt{n})$, and then for each subspace compute the local inner-products of all points assigned to it which again takes time $O(\sqrt{n}(\sqrt{n})^2) = O(n\sqrt{n})$ in total. Should these assumptions be violated the algorithm degrades naturally, with an overall run-time of $O(nm + \sum_i n_i^2)$, where m is the total number of charts, and n_i the number of points assigned to chart c_i.

Even with these modifications the asymptotic complexity of training an SVM using a cutting plane algorithm is $O(n^3)$. However for the datasets we consider the primary bottleneck lies in computation of the kernel matrix,and as such, a reduction in the complexity of computing the kernel has significant impact on run-time. See table 5 for a detailed breakdown of the run-time of the different components of our method vs. global RBF kernel. In practice, for all reported experiments we use the 10 closest subspaces in our approximation.

4.4 Integration with Efficient Primal Solvers

The restricted case in which $K_{x,y}^i = P_i^{\perp}(x) \cdot P_i^{\perp}(y)$ deserves special attention. In this case, we can solve the problem efficiently in the primal by taking as a

[3] $w_y^i K_{x,y}^i \leq 1$ in the case of an RBF kernel.

[4] These are sensible assumptions, and not just chosen to make asymptotic improvements possible. As the number of charts steadily increases Atlas will be able to approximate better any underlying manifold, while the fact that the number of charts grows sub-linearly means that Atlas should exhibit increasing robustness to sampling error.

feature vector for point x the concatenation of weighted projections $w_x^i P_i^\perp(x)$, and this allows the use of efficient schemes such as averaging stochastic gradient descent [5] that can exploit the sparsity of the training. As each feature vector is sparse with $O(\sqrt{n})$ non-zero components, computation of the inner products and sparse updates of the weight vector are $O(\sqrt{n})$ operations, making the overall run-time associated with a fixed number of passes over the training set $O(n\sqrt{n})$.

As a point of effectiveness, the linear kernel performs significantly better if we allow the SVM to learn a bias for each chart separately. As such, we train a standard linear SVM over the sparse feature vector

$$f_{\mathbf{x}} = \bigoplus_{c_i \in \mathcal{A}} w_x^i [1,\, P_i^\perp(x)]. \tag{13}$$

where $\bigoplus$ is the concatenation operator.

In the experimental section, we also explore the use of local quadratic kernels, both those without cross terms, in which the sparse feature vector takes the form

$$f_{\mathbf{x}} = \bigoplus_{c_i \in \mathcal{A}} w_x^i [1,\, P_i^\perp(x),\, P_i^\perp(x)^2] \tag{14}$$

with $P_i^\perp(x)^2$ being the *elementwise* square of $P_i^\perp(x)$, and those with cross-terms:

$$f_{\mathbf{x}} = \bigoplus_{c_i \in \mathcal{A}} w_x^i [1,\, P_i^\perp(x),\, l(P_i^\perp(x) \otimes P_i^\perp(x))] \tag{15}$$

where $l(P_i^\perp(x) \otimes P_i^\perp(x))$ is the vectorization of the lower triangular (inclusive of diagonals) component of the outer product matrix $P_i^\perp(x) \otimes P_i^\perp(x)$.

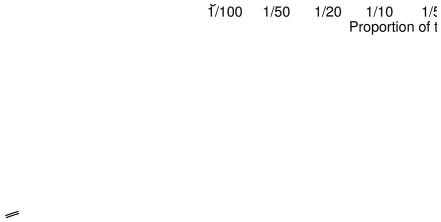
While in high-dimensional feature spaces, the use of quadratic features is largely unnecessary and incurs a substantial additional computation cost[5], in the local low-dimensional spaces of the manifold, the use of quadratic features incurs little overhead, and offers a noticeable improvement in discriminative performance.

5 Experiments

Semi-Supervised Learning: To illustrate the effectiveness of our approach in a semi-supervised situation, where the amount of labelled data is sparse relative to the total amount of data, we evaluate on MNIST by holding back the labels of a proportion of the training data. We generate a single Atlas over all training and test data, of local dimensionality 30, and calculate the classification error averaged over 20 trials varying the amount of labelled training data used from $\frac{1}{100}^{\text{th}}$ of the original training data (600 training samples) to $\frac{1}{2}$ of the data (30,000 training samples). As can be seen in figure 1, with sparse training data, AtlasRBF

[5] For example on MNIST, the raw feature vectors lie in a 784 dimensional space, while the quadratic features including cross terms lie in a 307,720 dimensional space.

Table 1. Classification performance on MNIST varying the proportion of labelled data. For all experiments, we use the same Atlas of local dimensionality 30, $\lambda = 100$, and $k = 2$, containing 835 charts. Zoom electronically to see standard deviation.



Training set ratio	1/100	1/50	1/20	1/10	1/5	1/4	1/3	1/2	1/1
Linear SVM	12.48 ± 0.33	10.65 ± 0.31	8.86 ± 0.13	7.85 ± 0.10	7.02 ± 0.12	6.83 ± 0.07	6.55 ± 0.10	6.20 ± 0.09	5.52
AtlasLin (eq. 13)	11.05 ± 0.60	7.15 ± 0.36	4.58 ± 0.16	3.41 ± 0.07	2.65 ± 0.08	2.44 ± 0.05	2.24 ± 0.06	1.94 ± 0.08	1.56
RBF SVM	8.95 ± 0.33	6.75 ± 0.24	4.73 ± 0.11	3.59 ± 0.07	2.73 ± 0.06	2.47 ± 0.06	2.20 ± 0.06	1.86 ± 0.05	1.41
AtlasRBF (eq. 12)	$\mathbf{4.13 \pm 0.20}$	$\mathbf{3.50 \pm 0.13}$	$\mathbf{2.87 \pm 0.06}$	$\mathbf{2.45 \pm 0.06}$	$\mathbf{2.12 \pm 0.05}$	$\mathbf{1.99 \pm 0.05}$	$\mathbf{1.87 \pm 0.05}$	$\mathbf{1.67 \pm 0.04}$	**1.31**

drastically outperforms other methods –achieving significantly less than half the error of an RBF kernel at maximum sparsity (4.13% vs. 8.95% error)– while the performance of the efficient linear Atlas kernel approximately tracks that of the standard RBF kernel. In the limit, with full training data effectively covering the testing data, the performance of AtlasRBF and the RBF kernel almost converges, with AtlasRBF retaining a small edge (see table 1).

Table 2. Comparison with semi-supervised approaches. With 100 labelled points, the extreme sparsity of the training data required a simpler Atlas with fewer charts. For this, we set $\lambda = 1000$, resulting in an Atlas with 207 charts. The parameters γ, σ are the same as the experiments in tables 1 and 4.

Method	100 labelled points	1000 labelled points
RBF SVM	22.70 ± 1.35	7.58 ± 0.29
EigenFunction	21.35 ± 2.08	11.91 ± 0.62
PVM(hinge loss)	18.55 ± 1.59	7.21 ± 0.19
AnchorGraphReg	9.40 ± 1.07	6.17 ± 0.15
AtlasRBF	$\mathbf{8.10 \pm 0.95}$	$\mathbf{3.68 \pm 0.12}$

The majority of semi-supervised approaches can not be used on datasets as large as MNIST (see discussion in [15]). As such, we also follow the protocol of [15] and compare our generalisation performance trained with 100 and 1000 training samples against three other scalable approaches: Eigenfunction [10], PVM [28], and AnchorGraphReg [15], alongside RBF SVMs.

Supervised Learning. To validate our approach we tested our algorithm on standard classification datasets MNIST, USPS, SEMEION, and LETTER. In all cases we compare the results from our Atlas-based kernel SVMs with Linear SVMs and RBF-kernel SVMs on the original data. In addition, for MNIST, USPS and LETTER we show comparisons with state-of-the-art approaches that use different variants of local co-ordinate coding [13, 25, 32], as well as large margin multi-class kernel-based classifiers [4, 8, 23] (see Table 3). We use SEMEION to compare against the most recent manifold learning approaches followed by nearest neighbour classifier.

Datasets. The MNIST, USPS, and SEMEION datasets consist of grayscale images of handwritten digits '0' – '9'. Both USPS and SEMEION contain images of resolution 16×16 encoded as 256 dimensional binary feature vectors. USPS contains 7291 training and 2007 testing images while on SEMEION, following [30], we create 100 random splits of the data with 796 training and 797 testing images in each set and report average error. The MNIST dataset is substantially larger, with $60,000$ training and $10,000$ test Grey-scale images.

Our choice of datasets was driven by the desire to compare the performance of our approach against as many alternative methods as possible. The 4 datasets we selected are popular datasets, used by many authors and allow us to give scores from a wide variety of related methods and show that our approach provides improved performance.

Implementation. We first perform manifold learning using the *Atlas* algorithm [18] to approximate of the underlying manifold as an atlas of piecewise affine overlapping charts before running our efficient SVM learning approach using linear, quadratic, and RBF chart-based kernels.

[18] takes three parameters as an input: the local dimensionality d common for all charts, a weight $\lambda \in \{10^0, 10^1, \dots, 10^5\}$ governing the strength of the MDL prior, the number of nearest neighbours $k \in \{2, 4, \dots, 10\}$ and d the local dimensionality of the manifold. For LETTER a 16-dimensional dataset we take $d \in [5, 10]$, and for all other datasets, we search $d \in \{5, 10, 15, 20, 30\}$. LLE and LTSA also need the local dimensionality and the number of neighbours as an input, and we search over the same range of values as Atlas. For SMCE we finely tune its parameter λ so that its local dimensionality varies over the same range as other methods. For all SVM kernel methods $\sigma^{-1}, \gamma^{-1} \in \{2^{-1}2^{-2}, \dots, 2^{-7}\}$, except on MNIST where a finer search of $\sigma^{-1} \in \{0.03, 0.031, \dots 0.04\}$ was required to replicate the performance of an RBF kernel reported in `http://yann.lecun.com/exdb/mnist/`.

The parameter σ_r of a raw RBF kernel can be understood as a compromise between the two parameters γ and σ used in AtlasRBF in that it should be chosen to be somewhere close to γ preventing generalisation off the manifold, but also close to σ to allow generalisation on the manifold. Empirically, for the parameters selected, this is always the case: On AtlasRBF $\gamma > \sigma$, and the raw RBF $\sigma_r \in [\gamma, \sigma]$. For example on USPS $\gamma = 2^3$, $\sigma = 2^7$, while $\sigma_r = 2^5$.

In our experiments we used two SVM solvers: The primal linear solver *SvmAsgd* [5] combined with a *one-versus-all* merging of binary SVMs; *Lib*SVM [7] allows the use of a precomputed custom kernel such as our chart-based RBF kernel merged using the built-in implementation of the *one-versus-one* merging SVMs

Table 3. Supervised classification with efficient primal SVM solvers or 1-NN. Our chart-based linear and quadratic kernel SVMs outperform all single-chart manifold learning methods followed by SVMs as well as Atlas followed by 1-NN on all datasets. Variants of local coordinate coding with SVMs also perform worse than our method on LETTER and MNIST, while [32] has slightly lower error on USPS. Scores for Linear SVMs and manifold learning methods are from our experiments and scores for other methods are as reported elsewhere. SMCE failed to converge on LETTER. All manifold learning methods except Atlas required more than 30GB of ram on MNIST and failed to complete.

Nearest Neighbour and Efficient Primal Formulations

	USPS	LETTER	SEMEION	MNIST
	error (%)	error (%)	error (%)	error (%)
Local coordinate based SVMs				
LL-SVM[13]	5.78	5.32	–	1.85
Linear SVM + G-OCC [32]	4.14	6.85	–	1.72
Linear SVM + C-OCC [32]	**3.94**	7.35	–	1.61
Linear SVM + LLC (512 anchor points) [25]	5.78	9.02	–	3.69
Linear SVM + LLC (4096 anchor points) [25]	4.38	4.12	–	2.28
Linear SVM + Tangent LLC (4096 points) [26]	–	–	–	1.64
Manifold Learning + Linear SVMs[5]				
Linear SVM on original space	8.42	35.75	7.40	5.55
SMCE[9]	6.88	–	9.04	–
LLE[21]	12.61	74.50	12.47	–
LTSA[31]	9.37	69.10	46.06	–
Manifold Learning + 1-NN classifier				
1-NN on original space	4.98	4.35	10.92	5.34
SMCE[9]	7.47	–	9.26	–
LLE[21]	6.83	19.03	9.41	–
wLTSA [30]	8.77	40.65	10.12	–
Atlas [18]	5.38	17.28	8.27	5.13
Primal Atlas SVMs (SvmAsgd)				
AtlasLinear - Hard Assignment (see eq. 6)	5.58	16.65	8.44	3.71
AtlasLinear (see eq. 13)	4.68	**3.13**	6.19	1.78
AtlasQuad (see eq. 14)	4.04	3.63	6.02	1.76
AtlasQuadCross (see eq. 15)	4.09	3.33	**5.48**	**1.46**

Comparison with standard Manifold learning. Looking at the results of tables 3 and 4, several themes can be seen. In general, the fusion of stock manifold learning techniques [9, 21, 30] with either linear or kernel SVMs is of limited value, and is perhaps more likely to hurt SVM scores than to improve them. In contrast, our Atlas kernels show substantial improvement over any baseline SVM approach (the only exception being the use of an RBF kernel on the already low dimensional dataset letter). Every type of our Atlas based kernels out-performs every use of stock manifold learning methods, both when used in conjuncture with a linear or kernel SVM, or as a nearest neighbour classifier.

Table 3 shows a comparison of the efficient methods on USPS, LETTER, MNIST, and SEMEION. On three of the four datasets, our approach, and particularly AtlasQuadCross, significantly outperforms all other methods. Note that, the local coordinate methods do not report scores on SEMEION. However, our efficient primal approach obtains substantially better scores than a standard RBF kernel (see table 4). In particular, on the LETTER dataset our approach to learning on an atlas halves the classification error of [32] and substantially improves on the classification error obtained with the coordinate coding approach of [13, 25].

Table 4. Classification performance on USPS, LETTER, SEMEION, and MNIST datasets. Our chart-based RBF kernel outperforms all other multi-class kernel based SVMs as well as all single-chart manifold learning methods followed by RBF SVMs. *Lib*SVM with an RBF kernel on the raw data achieved the best performance on LETTER, AtlasRBF is best on all other datasets. The comparison between AtlasRBF with soft and hard assignment shows the impact of our novel kernels. SMCE failed to converge on LETTER.

Kernel methods using cutting-plane type approaches

	USPS	LETTER	SEMEION	MNIST
Method	error (%)	error (%)	error (%)	error (%)
Global SVMs				
MCVSVM[8]	4.24	2.42	–	1.44
SVM$_{struct}$ [23]	4.38	2.40	–	1.40
LaRank [4]	4.25	2.80	–	1.41
LibSVM on raw data[7]	4.53	**2.05**	6.41	1.41
Manifold Learning + RBF SVMs[7]				
SMCE[9]	6.18	–	8.68	–
LLE[21]	4.78	5.38	6.93	–
LTSA[31]	7.03	44.63	9.17	–
Atlas-Kernel SVMs (LibSVM)				
AtlasRBF - Hard Assignment (see eq. 6)	4.63	4.95	7.15	3.13
AtlasRBF (see eq. 12)	**3.68**	2.33	**5.14**	**1.31**

Table 4 shows that in comparison with RBF SVMs and the multi-class kernel-based SVMs of [4, 8, 23], we achieve substantial improvement in classification performance on USPS. Our AtlasRBF kernel outperforms all methods with the exception of the global RBF kernel SVM on the LETTER dataset. As LETTER is 16-dimensional, it does not allow for the advantages of the manifold learning methods to be fully employed, it is perhaps unsurprising that manifold learning is not only unnecessary, but also slightly detrimental, as we see higher errors for the LCC-based methods. As [18] allows the learning of a manifold of arbitrary dimension, we could learn the trivial 16-dimensional manifold, composed of a single chart, and where the projection matrix $P^{\perp}(x)$ is the identity function. In such cases our performance is identical to that of the RBF kernel. As such a result is uninformative, we instead cap the local manifold dimensionality at 10, when reporting our result. Our approach still achieves the second best performance and outperforms all other multi-class kernel-based methods.

Tables 3 and 4 clearly show the importance of forcing the classifier to vary smoothly, when generalising to the testing set. While the smooth AtlasRBF kernel consistently outperforms related work, the hard assignment kernels (6) of section 4.1 show that training a single kernel in each chart without soft assignment is noticeably worse than existing approaches. Table 5 shows the response of our Atlas-based approaches vs. global linear and RBF kernels to increasing levels of Gaussian noise. Our approach appears to behave better with respect to noise with lower classification errors. The parameters used are the same as in tables 3 and 4.

Chart Characterisation. Our results can also be used to validate the manifold learning tenets of [19]. Particularly table 4, where the improved results come from forcing an RBF kernel to conform to the manifold [6], clearly show unlabelled data is important and that a learned manifold can improve the performance of classifiers. Empirically, tenet 2 (that a manifold can be fitted to the data) also holds and explains the success of our approach. Tenet 3 states that different classes should lie on different areas of the manifold. This can be tested by seeing if different classes belong to different charts of the Atlas.

Although [18] learns the set of overlapping affine charts in a totally unsupervised manner, tenet 3 suggests that points which share similar statistical properties and are more likely to lie on the same subspace or chart would also share the same label information. In fact, on USPS, most of the 18 charts learned contain points from a single dominant class, where for the median chart 96% of the points assigned to it come from the same class. However, some charts contain two or three prevalent classes and around 10% of the data label differs from that of its interior chart. On MNIST 262 out of 835 charts contain data from the same class and 180 contain more than 10% points whose label differs from the dominant class. Similarly, for the median chart 98% come from the dominant class. In total 6.9% of the data does not belong to the dominant class of the interior of the chart it is assigned to. Along with providing empiric validation of tenet 3, the fact that 5-10% of the data does not reflect the dominant label of the chart provides some insight in the difference in performance between NN, linear and RBF kernels, and implicitly bounds the maximal error of any classifier trained on this Atlas.

Table 5. Extended analysis on USPS

(a) Run-time of various components. The top row shows the run-time of components, while the bottom row shows the accumulated time.

	Init.	Atlas	Kernel	SVM train	SVM test
AtlasLinear	44.44	+10.29	+0.78	+0.78	+0.16
	44.44	54.73	55.51	56.29	56.45
AtlasRBF	44.44	+10.29	+28.99	+9.68	+2.53
	44.44	54.73	83.72	93.40	95.93
LibSVM	-	-	-	99.93	+95.47
	-	-	-	99.93	195.40

(b) Classification on USPS with increasing Gaussian noise.

Noise	2%	5%	10%	15%	20%	30%
Linear SVM	8.72	8.82	9.07	9.82	10.21	11.36
AtlasLinear	5.08	5.83	6.03	5.93	6.78	11.46
RBF SVM	4.58	4.58	5.53	5.58	**6.33**	8.67
AtlasRBF	**4.04**	**4.14**	**4.48**	**5.08**	6.34	**7.57**

[6] In contrast, table 3 shows that the charts found can be used to raise the data into a high-dimensional space, where linear SVMs perform better.

6 Conclusion

We have presented a novel approach to supervised and semi-supervised learning via training a manifold on unlabelled data. We have shown superior performance to both RBF kernels and local co-ordinate based methods on standard datasets, and to manifold learning based nearest neighbour. As such it provides additional empiric validation of the tenets of manifold learning first proposed in [19]. Our method provides a principled way for Support Vector Machines to make use of unlabelled data in learning a kernel, and we intend to further explore the benefits of this.

References

[1] Belkin, M., Niyogi, P.: Laplacian eigenmaps and spectral techniques for embedding and clustering. Advances in Neural Information Processing Systems 14, 585–591 (2001)

[2] Belkin, M., Niyogi, P., Sindhwani, V.: On manifold regularization. AISTATS (2005)

[3] Bellman, R.: Dynamic Programming. Dover Publications (March 1957)

[4] Bordes, A., Bottou, L., Gallinari, P., Weston, J.: Solving multiclass support vector machines with larank. In: Proceedings of the 24th International Conference on Machine Learning, pp. 89–96. ACM (2007)

[5] Bottou, L.: Large-scale machine learning with stochastic gradient descent. In: Lechevallier, Y., Saporta, G. (eds.) Proceedings of the 19th International Conference on Computational Statistics (COMPSTAT 2010), pp. 177–187. Springer, Paris (2010), http://leon.bottou.org/papers/bottou-2010

[6] Boykov, Y., Kolmogorov, V.: An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision. PAMI 26(9), 1124–1137 (2004)

[7] Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 27:1–27:27 (2011), software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

[8] Crammer, K., Singer, Y.: On the algorithmic implementation of multiclass kernel-based vector machines. J. Mach. Learn. Res. 2, 265–292 (2002), http://dl.acm.org/citation.cfm?id=944790.944813

[9] Elhamifar, E., Vidal, R.: Sparse manifold clustering and embedding. In: Advances in Neural Information Processing Systems, pp. 55–63 (2011)

[10] Fergus, R., Weiss, Y., Torralba, A.: Semi-supervised learning in gigantic image collections. In: Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C.K.I., Culotta, A. (eds.) Advances in Neural Information Processing Systems 22, pp. 522–530 (2009)

[11] Jayasumana, S., Hartley, R., Salzmann, M., Li, H., Harandi, M.: Kernel methods on the riemannian manifold of symmetric positive definite matrices. In: CVPR IEEE (2013)

[12] Ladický, L., Russell, C., Kohli, P., Torr, P.H.: Inference methods for crfs with co-occurrence statistics. International Journal of Computer Vision 103(2), 213–225 (2013)

[13] Ladicky, L., Torr, P.: Locally linear support vector machines. In: Proceedings of the 28th International Conference on Machine Learning (ICML 2011), pp. 985–992 (2011)

[14] Lee, J.M.: Introduction to smooth manifolds, vol. 218. Springer (2012)

[15] Liu, W., He, J., Chang, S.F.: Large graph construction for scalable semi-supervised learning. In: Fürnkranz, J., Joachims, T. (eds.) Proceedings of the 27th ICML (ICML 2010), pp. 679–686. Omni Press, Haifa (2010), http://www.icml2010.org/papers/16.pdf

[16] von Luxburg, U., Bousquet, O.: Distance–based classification with lipschitz functions. The Journal of Machine Learning Research 5, 669–695 (2004)

[17] Melacci, S., Belkin, M.: Laplacian support vector machines trained in the primal. Journal of Machine Learning Research 12, 1149–1184 (2011)

[18] Pitelis, N., Russell, C., Agapito, L.: Learning a manifold as an atlas. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR (2013)

[19] Rifai, S., Dauphin, Y., Vincent, P., Bengio, Y., Muller, X.: The manifold tangent classifier. Advances in Neural Information Processing Systems 24, 2294–2302 (2011)

[20] Russell, C., Fayad, J., Agapito, L.: Energy based multiple model fitting for non-rigid structure from motion. In: 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3009–3016. IEEE (2011)

[21] Saul, L., Roweis, S.: Think globally, fit locally: unsupervised learning of low dimensional manifolds. The Journal of Machine Learning Research 4, 119–155 (2003)

[22] Sindhwani, V., Niyogi, P.: Linear manifold regularization for large scale semi-supervised learning. In: Proc. of the 22nd ICML Workshop on Learning with Partially Classified Training Data (2005)

[23] Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y., Singer, Y.: Large margin methods for structured and interdependent output variables. Journal of Machine Learning Research 6(2), 1453 (2006)

[24] Vapnik, V.: The Nature of Statistical Learning Theory. Springer (1995)

[25] Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., Gong, Y.: Locality-constrained linear coding for image classification. In: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3360–3367. IEEE (2010)

[26] Yu, K., Zhang, T.: Improved local coordinate coding using local tangents. In: Proc. of the Intl. Conf. on Machine Learning, ICML (2010)

[27] Yu, K., Zhang, T., Gong, Y.: Nonlinear learning using local coordinate coding. Advances in Neural Information Processing Systems 22, 2223–2231 (2009)

[28] Zhang, K., Kwok, J.T., Parvin, B.: Prototype vector machine for large scale semi-supervised learning. In: Proceedings of the 26th Annual ICML, ICML 2009, pp. 1233–1240. ACM, New York (2009), http://doi.acm.org/10.1145/1553374.1553531

[29] Zhang, T., Tao, D., Li, X., Yang, J.: Patch alignment for dimensionality reduction. IEEE Transactions on Knowledge and Data Engineering 21(9), 1299–1313 (2009)

[30] Zhang, Z., Wang, J., Zha, H.: Adaptive manifold learning. IEEE Transactions on Pattern Analysis and Machine Intelligence 34(2), 253–265 (2012)

[31] Zhang, Z., Zha, H.: Principal manifolds and nonlinear dimension reduction via local tangent space alignment. SIAM Journal of Scientific Computing 26, 313–338 (2002)

[32] Zhang, Z., Ladicky, L., Torr, P., Saffari, A.: Learning anchor planes for classification. In: Advances in Neural Information Processing Systems, pp. 1611–1619 (2011)